UNIVERSITÀ
DEGLI STUDI
FIRENZE

DOTTORATO DI RICERCA IN

Tecnologie Elettroniche per l'Ingegneria dell'Informazione

indirizzo RF, MICROWAVES AND ELECTROMAGNETICS

CICLO XXVI

NUMERICAL TECHNIQUES FOR ANTENNA ARRAYS: MULTI-OBJECTIVE OPTIMIZATION AND METHOD OF MOMENTS ACCELERATION

Settore Scientifico Disciplinare ING-INF/02

Dottorando
Dott. RUGGERO TADDEI

Tutori
Prof. GIUSEPPE PELOSI

Dott. STEFANO SELLERI

Coordinatore
Prof. GIANFRANCO MANES

Anni dal 2011 al 2014

Contents

Preface

The approximate solution of Maxwell's equations exploiting Numerical Techniques is known as Computational ElectroMagnetics (CEM). CEM techniques have been available for close on four decades now, and they currently form an invaluable part of current RF, antenna and microwave engineering practice. CEM is a multi-disciplinary field: its core disciplines are electromagnetic theory and numerical methods, but for its practical implementations, geometric modelling, computer science and algorithms are important. The applications of CEM are many, and include antennas, biological EM effects, medical diagnosis and treatment, electronic packing and high-speed circuits, microwave devices, monolithic microwave integrated circuits, materials, avionics, communications, radars and imaging, surveillance and many others.

Before the advent of high-speed computers, it was advantageous to expend considerable effort, recurring to elaborate mathematics, to manipulate solutions analytically into a form which minimized the subsequent computational effort. By these methods, the solution of the electromagnetic field inside a complex device under design can be a very lengthy process. Indeed, it is often the case that no closed form solution is possible without making drastic simplifying assumptions: as a consequence of these assumptions, the ensuing solution cannot be completely reliable and defeats the purpose of the analysis, which is to deliver an accurate design. It is now often more convenient to use methods that are analytically simple, but require large amount of computation; furthermore, many problems of practical interest can be solved only by the use of such methods. Because of the great speed and storage capabilities of modern computers, almost any problem of linear analysis can be solved to some degree of accuracy. Although these schemes are known as *approximation* methods, the term is misleading, since it is possible to increase the accuracy as much as desired, at some additional computing cost. These digital approximate solutions may be less accurate than the closed form solutions from classical analysis for simple problems shapes such as circles and rectangles (which rarely exist in the real design environment). However, in real world problems, that involve complex geometries and electromagnetic configurations, numerical schemes yield far more accurate solutions than those possible by classical analysis, in view of the latter methods' above mentioned dependence in simplifying assumptions. However, the widespread adoption of computational methods to complement the traditional tools of analysis and measurement has to be considered with caution,

since in some situations codes are being applied by users unfamiliar with the basic formulations underlying them, and not infrequently to problems for which the codes were not designed.
The present work is focused on two different applications of numerical techniques for Computational Electromagnetics: numerical optimization and full-wave techniques. The thesis structure is the following:

- Part I is focused on the Multi-Objective Taguchi optimization Method (MO-TM) for electromagnetics. In Chap. 1 foundations of numerical optimization will be presented, while Taguchi's Optimization Method, in its single- and multi-objective implementation, will be illustrated in Chap. 2. Finally in Chap. 3 the algorithm will be applied to a series of test problems and real world engineering design situations.

- Part II is focused on a Method of Moments (MoM) acceleration via ASM-MBF for antenna arrays. Chap. 4 presents the Method of Moments, and the proposed implementation is in Chap. 5. Array Scanning Method-Macro Basis Function acceleration will be presented in Chap. 6 . Finally in Chap. 7 the code and its accelerated version will be applied to a series of computational benchmarks.

Part I

Multi-Objective Taguchi Optimization Method for Electromagnetics

Chapter 1

Numerical Optimization

Designing an electronic system is an engineering task which requires the satisfaction of a given set of specifications, or constraints, over the system behaviour. For example, in a waveguide bandpass filter, in-band ripple and out-band rejection are typical requirements, while the overall length of the device might be constrained to be smaller than a given length; in antenna design, the radiation pattern is often required to comply to a mask, or to several masks in multi-beam array design, while keeping the antenna geometry simple. There is a limited number of analytic synthesis techniques leading from specifications to design directly, and these are usually limited to very simple and particular problems, usually with limited applications. When an analytic synthesis technique is not available, the experience of the designer is fundamental in driving the whole process. If it is possible to express the compliance of a given system to the requirements in a mathematical form giving a measure of how much it is far from compliance, then it would be possible to reach a satisfactory the design by seeking for the minimum of such a function, which is usually referred to as *cost* or *objective function*. If it is more natural for a given problem to define a function to be maximized, rather than minimized, it is then more appropriate to refer to the function as to a *fitness function*. A whole branch of mathematics, *optimization*, is devoted to seeking for the minimum (or the maximum) of a cost (or of a fitness) function. One of the advantages of optimization techniques is that they are completely general methods, unrelated to the real system to be designed, and which can hence be applied to any design problem.

Optimization is something sought for since long: Fermat and Lagrange proposed calculus-based formulas for finding optimum points, while Newton, Gauss and Cauchy proposed iterative techniques for moving towards an optimal value. Being most of the optimization techniques iterative, they were a challenging task to perform without a computer. Indeed the first applications of optimization techniques to real problems are dated back to world war II [1], [2]. It was only after the advent of digital computers, which enabled at the same time the implementation of optimization algorithms and the accurate simulation of complex systems for an efficient evaluation of the cost function value associated to

a given design, that optimization began to be widely used in engineering.

The first techniques to be devised were deterministic and local optimizations. Deterministic implies that, given a starting search point, algorithms always follow the same path to the same optimal point; they are local, as the path leads to the closest minimum. The most popular of these deterministic techniques are gradient and conjugate based methods [3]–[5], where the minimum is sought by evaluation, or approximation, of the gradient of the function at the test point, and iteratively moving towards the direction in which the function decreases. Finding just the *local* minimum, i.e. the one close to the initial guess, is a poor strategy; a more interesting approach is finding the *global* minimum, i.e. the best possible solution in the search space. Obviously, global optimization is much harder than local, but its relevance is much greater. Global optimization techniques can be further divided into deterministic and stochastic techniques: global deterministic techniques [6], [7] do exist, but stochastic techniques are the most popular. Among them, evolutionary algorithms (EA) [8], [9], capable of mimicking nature behavior have proven to be very effective and efficient: a non-exhaustive, but detailed, list of algorithms includes Genetic Algorithms (GA) [10], Particle Swarm Optimization (PSO) [11], [12], simulated annealing [13], [14], tabu search [15]–[17], random search [18], [19], ant colony [20], invasive weed [21], [22], bacteria foraging [23] and others. Stochastic techniques, as opposed to deterministic ones, presents some sort of randomness somewhere in the algorithm: this brings the advantage of being able to have a relatively simple technique, yet able to perform an efficient search throughout the whole design variables' domain. An important point of stochastic technique is that, due to randomness, two consecutive runs of the algorithm over the same function to be optimized generally lead to a different optimal point; moreover, a possible drawback in stochastic algorithms is that they usually require a higher number of cost function evaluations with respect to deterministic ones.

A second broad division does not involve the optimization technique, but the problem at hand, that is the kind of cost function, either scalar or vectorial, which is appropriate to model it: in the scalar problem there is just one function, or objective, to be handled, and hence relative techniques are called single-objective (SO) optimizations. On the other hand, dealing with real life optimization problems, it is rather common to have to handle problems that intrinsically present more than one cost function. This is the case when several constraints are defined, and for each of them a cost function can be assigned, leading to a multi-objective (MO) optimization. Objectives to be optimized are typically conflicting with respect to each other. Indeed three cases can be recognized: objectives are totally conflicting, non-conflicting, or partially conflicting [24]. In the first case, the conflicting nature of the objectives implies that no global improvement can be made without violating any constraint, hence no optimization is even possible. On the other hand, in non-conflicting problems the various objectives are related so that any change leading to a better performance in one objective leads also to the improvement, or at least not to the worsening, of the other objectives. This class of multi-objective problems can be conveniently reduced to single-objective problem via scalarization (Section 1.1.1),

since a *single* optimal solution do exists for such a multi-objective problem. The most interesting, and indeed the most appropriate to model real-world engineering designs, is the class of problems with partially-conflicting objectives. In this class there is no single optimal solution, and indeed an *optimum* solution must be carefully defined (Section 1.1.2). It is always possible to define a single cost function as a weighted sum of the various cost functions also in this case, but this is a poor strategy since optimal weights values, that deeply affect the optimization process, are unknown a-priori. Furthermore, it is often more appealing to have a *set* of possible optimal solutions, among which the designer can select a good *trade-off* solution to represent the best possible compromise among all the objectives [25]–[27]. This latter option is the essence of multi-objective optimization, presented with details in [P1] together with a number of multi-optimization algorithms. MO basics will be given in Section 1.1, describing the two main approaches to this problem: scalarization and Pareto techniques, the latter being a true Multi-Objective optimization approach. Fig. 1.1 shows a possible classification of methods.

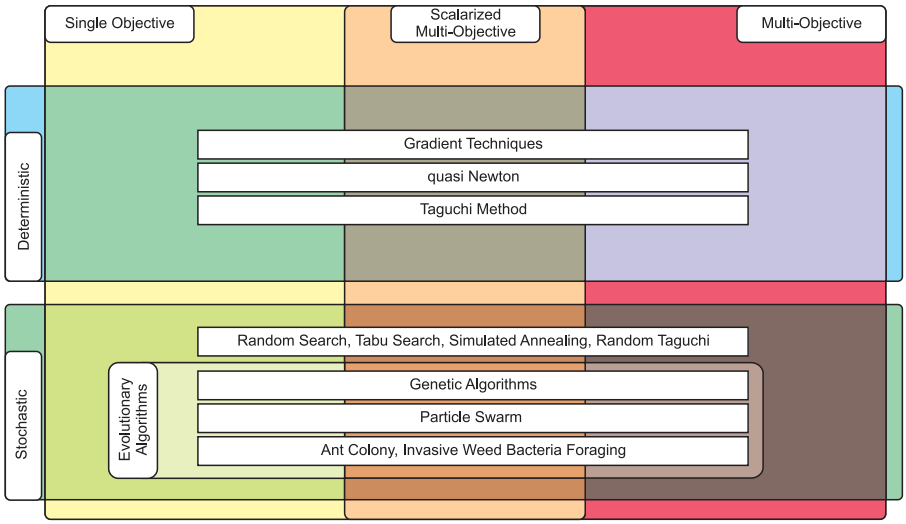


Figure 1.1: Classification of some of the most popular optimization methods. On the left the class of true single objective optimization, on the right true multi-objective optimization, in the middle scalarized multi-objective. On top deterministic method, on the bottom stochastic methods. Of course since the same method can be applied either to SO or to MO, methods can span several areas.

1.1 Multi-Objective Optimization

Multi-objective optimization (MOO) may be defined as a strategy to address multiple design constraints in practical engineering problems. A MOO design

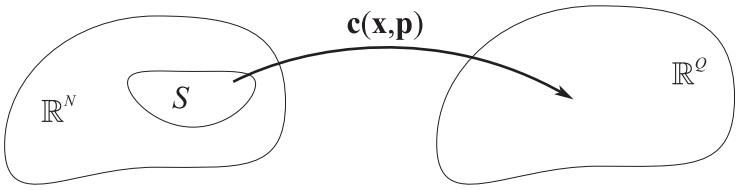


Figure 1.2: Basic MOO: find the global minimum of $c : \mathbb{R}^N \to \mathbb{R}^Q$

may be formulated as:

$$\min \mathbf{c}(\mathbf{x},\mathbf{p}) \quad \text{with} \begin{cases} \mathbf{x} \in S \\ x_n^{(LB)} \leq x_n \leq x_n^{(UB)}, \quad n = 1,\dots,N \\ \mathbf{g}(\mathbf{x},\mathbf{p}) \leq 0; \qquad \mathbf{h}(\mathbf{x},\mathbf{p}) = 0 \end{cases} \tag{1.1}$$

being $\mathbf{c}$ the multi-valued cost function $\mathbf{c}(\mathbf{x},\mathbf{p}) = [c_1(\mathbf{x},\mathbf{p}), c_2(\mathbf{x},\mathbf{p}), \dots, c_Q(\mathbf{x},\mathbf{p})]^\top$, a vector of $Q \geq 2$ real-valued cost functions, each dependent on the vector of N design variables $\mathbf{x} = [x_1, x_2, \dots, x_N]^\top$ and on a constant vector of P design parameters $\mathbf{p} = [p_1, p_2, \dots, p_P]^\top$ (Fig. 1.2). S is the domain of possible *feasible* designs. The individual design variables x_n are assumed continuous and can be changed independently within lower and upper bounds, $x_n^{(LB)}$ and $x_n^{(UB)}$, respectively. These bounds are themselves arranged in vectors $\mathbf{x}^{(LB)}$ and $\mathbf{x}^{(UB)}$. If S does not allow for full independence for the x_n, both a vector of K_i inequality constraints, $\mathbf{g}(\mathbf{x},\mathbf{p}) = [g_1(\mathbf{x},\mathbf{p}), g_2(\mathbf{x},\mathbf{p}), \dots, g_{K_i}(\mathbf{x},\mathbf{p})]^\top$, and K_e equality constraints, $\mathbf{h}(\mathbf{x},\mathbf{p}) = [h_1(\mathbf{x},\mathbf{p}), h_2(\mathbf{x},\mathbf{p}), \dots, h_{K_e}(\mathbf{x},\mathbf{p})]^\top$, have to be satisfied. As stated before, the minimum of the function is sought for. If the problem at hand naturally requires a maximum to be searched for, that is it is a fitness function rather than a cost one, the fitness function can be converted into a cost function by simply multiplying it by -1. The definition of *optimum* in a MOO case is in itself tricky; one of the first who tried to define it is Edgeworth [28] who defined an optimum for multi-criteria economic decision making by stating, for $N = Q = 2$, that the optimum point $[x_1, x_2]$ is such that, in whatever direction we take an infinitely small step, c_1 and c_2 do not increase together but that, while one increases, the other decreases. Pareto, contemporary of Edgeworth, defined what it is known as *Pareto Optimum* stating that "The optimum allocation of the resources of a society is not attained so long as it is possible to make at least one individual better off in his own estimation while keeping others as well off as before in their own estimation" [29].

Several approaches have been proposed to address this kind of optimization problems [27], [30]: they can be broadly divided in the two categories of scalarization methods and Pareto methods [31], [32]. These two categories will be analyzed in the following two sub-Sections.

1.1.1 Scalarization

Scalarization Methods approach the MO problem by reducing the vector cost function in (1.1) to a scalar one. The most common of these techniques is the weighted sum method (WSM); it projects the MO problem in a suitably defined equivalent SO one, whose fitness function is the dot product of the vector cost function $\mathbf{c}(\mathbf{x}, \mathbf{p})$ with a suitable real vector of positive weights $\mathbf{w} = [w_1, w_2, \ldots, w_Q]$. The resulting overall scalar cost is hence:

$$c_s(\mathbf{x}, \mathbf{p}) = \mathbf{w} \cdot \mathbf{c}(\mathbf{x}, \mathbf{p}) = \sum_{q=1}^{Q} w_q c_q(\mathbf{x}, \mathbf{p}) \tag{1.2}$$

Usually positive ($w_q > 0$) weighting coefficients are chosen in order to satisfy the normalization relationship

$$\sum_{q=1}^{Q} w_q = 1 \tag{1.3}$$

and by changing their values it is possible to control the relative importance of each design constraint, obtaining consequently different overall cost/fitness functions and sets of solutions. The key issue of the WSM approach is therefore to find out the best trade-off among weighting coefficients, and this requires a good knowledge of the relative importance of each of them with respect to the others. For this reason WSM often requires an extensive tuning of the weighting coefficients w_q, particularly for problems where objectives are unrelated. Another issue related to this approach is the inability to find Pareto-optimal solutions in non-convex regions, although a solution obtained with this approach is always Pareto-optimal. This approach however is a mere extension of the single-objective oriented strategy to multi-objective problems: therefore, in order to make the optimization process effective, it is necessary to have an in-deep knowledge of the problem to be optimized, so that the weighting coefficients are suitably chosen, and to run the single-objective optimizer a large number of times, to explore the different solutions: even then, good distribution of the results is not guaranteed.

1.1.2 The Pareto Front concept

A completely different approach is that provided by the so-called Pareto methods. These are techniques that do not offer a single optimal solution, but rather a set of optimal solutions, that is a set of non-dominated solutions (in the Pareto sense) capable of providing an approximation of the Pareto Front. A solution $\bar{\mathbf{x}}$ is *non dominated*, according to Pareto [29], if there is no other solution that has better values in all cost functions, i.e.:

$$c_q(\bar{\mathbf{x}}) < c_q(\mathbf{x}) \qquad \text{for at least one } q \in 0, 1, \ldots, Q \tag{1.4}$$

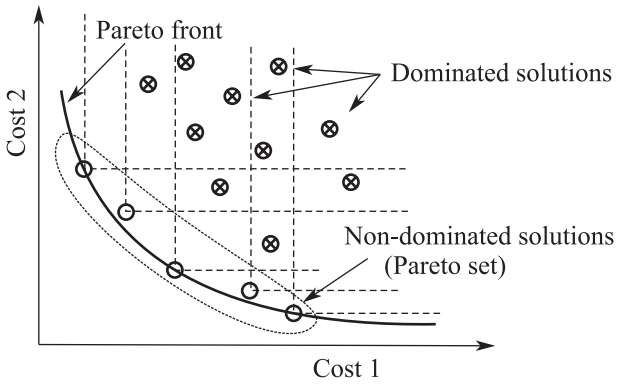


Figure 1.3: Cost function space and a finite set of solution, of which most are dominated (crossed circles) and some non-dominated (empty circles). Non-dominated solutions form the Pareto Set, which is an approximation of the true Pareto Front (note that solutions in the Pareto Set are not necessarily on the Pareto Front.)

It is possible to build, from all possible $\mathbf{x}$ in the design variables set, the set of non-dominated solutions, to determine the so-called *Pareto Front*. Of course an exhaustive search for the Pareto Front, which is often a continuous or piecewise-continuous subset of the cost function co-domain, is not feasible, so MO optimization rather consists on searching for a finite set of solutions which are as close as possible to the Pareto Front. Such a set is called a *Pareto Set* (Fig. 1.3).

A MO algorithm's job is to find a set of solutions, extract the non-dominated ones, and iteratively refine the search so as to have a Pareto Set which is at every step a better approximation of the true Pareto Front. Evolutionary Algorithms are well established as the best method at hand to seek for such an approximation of the Pareto Front; this is true both for the lack of true alternatives, and for their inherent parallelism and capability to exploit a recombination of known solutions to further explore the solution space. Since EA operate iteratively on a population of solutions, a selection operator is often necessary to reduce the current iteration population from its current number R to a smaller number S to which new individuals are added while exploring the cost function domain. While in a single objective optimization such operator is straightforward since the solution are ranked in terms of their cost function value and taking the best S ones is trivial, deciding which S out of R solutions, that are *all* on the Pareto Set, are to keep is a trickier matter, since they are all equally optimal in the Pareto sense. In this case a *crowding distance* selection is operated [33]. The crowding distance is a criterion based on comparison of the congestion around a solution and is bound to the dimensions of the largest cuboid which does not contain any other solution of the set (a rectangle in the two-costs case depicted in Fig. 1.4); the crowding distance is the average of the sides' length of that cuboid. It is apparent how, in Fig. 1.4, the crowding distance for solution A_1 is

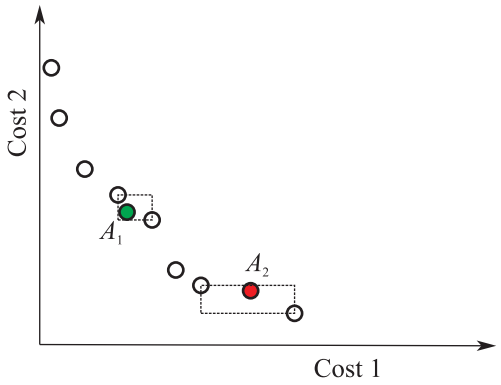


Figure 1.4: Crowding distance definition, and two examples.

much smaller than the crowding distance for solution A_2. A greater crowding distance is preferred in order to maintain the diversity of the solutions and a good distribution of the Pareto Set along the Pareto Front. Indeed terminal solutions, for which the cuboid is bounded by only one solution of the front, are assigned a very large crowding distance since they must be maintained in the selection process to ensure that the Pareto Front is spanned as much as possible. As a possible strategy, in [34] the MaxiMin procedure was proposed, in which the reduced set is built by first selecting the extreme points, i.e. those with the minimum (or maximum) value in each single cost, then adding points to the set one by one. Each new point is selected to maximize the distance from points that are already in the set; it works with distances over the selected individuals, in order to get a good spreading, while other approaches, like the crowding distance above, consider all individuals, without considering if they will be selected or not; in this way the search should have a better coverage of the cost space, leading to a more uniform front.

In summary, when solving a multi-objective problem, there are three main goals to achieve [35]:

1. Maximize the number of elements on the Pareto Set;
2. Minimize the distance of the Pareto Set produced by the algorithm with respect to the true Pareto Front;
3. Maximize the spread of solutions.

Knowledge of a good approximation of the Pareto Front allows the designer for a trade-off among a set of optimal solution at design time. This is much better than the previous scalar approach; mathematically, every Pareto optimal point is an equally acceptable solution of the multi-objective optimization problem, but from a practical point of view the designer needs only one solution, no matter whether the associated optimization problem is single or multi-objective, and so, a choice among these optimal solutions must be performed. Selecting one

out of the set of Pareto optimal solutions involves higher-level information which is often qualitative and experience-driven, hence it cannot be embedded in any objective function.

Chapter 2

Taguchi's Optimization Method

Taguchi's method (TM) is a statistical technique originally designed for quality control of manufactured goods [36], that has later become widespread in many different fields of engineering. The essential concept of Taguchi's quality method is the definition of a loss function for a particular production process, calculated by evaluating equivalent signal-to-noise-ratios (SNRs), defined as ratios between the magnitude of the mean of a process to its variation; this allows to define the best solutions of a problem as those leading to maximum values of SNRs, hence minimum values of the loss function.

Recently some efforts have been done to exploit TM capabilities in electromagnetics optimization [37]–[40]; some report a comparison between Taguchi's algorithm and well-known optimization techniques, like PSO [41]. Some others [42], [43] have hybridized TM with other optimization techniques (mainly PSO and GA), finding improvements in convergence speed and accuracy. In any case, while GA [44]–[46] and PSO [47]–[49] are well established, Taguchi method is still to be deeply investigated in electromagnetics. In particular existing articles in open literature have been focused on single-objective (SO) implementation of Taguchi's method; the proposed implementation is instead aimed at multi-objective (MO) optimization. Section 2.1 will introduce orthogonal arrays (OAs), a mathematical tool Taguchi's method is based on. In Section 2.2 Taguchi's method will be presented, in its classical single-objective version, to introduce basic notation and for the sake of completeness. In Section 2.3 the novel multi-objective algorithm derived from Taguchi's method will be described.

2.1 Orthogonal Arrays

TM relies on the concept of orthogonal arrays (OAs) [50]. OAs are a statistical tool proven to be useful in designing an optimization process with fewer

experimental runs with respect to a full factorial strategy. The definition of an OA is the following: let S be a set of s symbols; an orthogonal array on S with s levels and strength t is a matrix with N rows and k columns (where $t \leq k$) - usually referred to as $\mathrm{OA}(N, k, s, t)$ - where in every $N \times t$ sub-matrix, each t-uple appears the same number of times as a row [50]. Table 2.1 contains several examples of OA; for example, an $\mathrm{OA}(25, 6, 5, 2)$ on $S = \{0, 1, 2, 3, 4\}$. It has $N = 25$ rows (corresponding to 25 experiments) and $k = 6$ columns (6 parameters to optimize). Each parameter can assume one of the $s = 5$ different values in S. If a choice of any $t = 2$ columns is made, each pair (each t-uple) appears the same number of times as a row. In this OA, every combination of two columns appears exactly once as a row in each 25×2 sub-matrix, while every parameter appears exactly 5 times in every column. This characteristic of the OAs ensures a balanced and fair selection of parameters. The t parameter therefore indicates the maximum number of columns that lead to an equal number of occurrences as a row; e.g. with an OA with strength $t = 3$ every combination of one, two, or three input parameters has the same number of occurrences and hence is tested an equal number of times. In general, the strength t of the OA can be increased to consider interactions between more parameters; however, the larger the strength t is, the more rows (experiments) the OA has.

Table 2.1: Some examples of orthogonal arrays

$\mathrm{OA}(4, 3, 2, 2)$

Experiments	p_1	p_2	p_3
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	0

$\mathrm{OA}(8, 5, 2, 2)$

Experiments	p_1	p_2	p_3	p_4	p_5
1	0	0	0	0	0
2	1	0	0	1	1
3	0	1	0	1	0
4	0	0	1	0	1
5	1	1	0	0	1
6	1	0	1	1	0
7	0	1	1	1	1
8	1	1	1	0	0

$\mathrm{OA}(9, 4, 3, 2)$

Experiments	p_1	p_2	p_3	p_4
1	0	0	0	0
2	0	1	1	2
3	0	2	2	1
4	1	0	1	1
5	1	1	2	0
6	1	2	0	2
7	2	0	2	2
8	2	1	0	1
9	2	2	1	0

$\mathrm{OA}(25, 6, 5, 2)$

Experiments	p_1	p_2	p_3	p_4	p_5	p_6
1	0	0	0	0	0	0
2	0	1	1	2	3	4
3	0	2	2	3	4	1
4	0	3	3	4	1	2
5	0	4	4	1	2	3
6	1	0	1	1	1	1
7	1	1	2	4	0	3
8	1	2	4	0	3	2
9	1	3	0	3	2	4
10	1	4	3	2	4	0
11	2	0	2	2	2	2
12	2	1	4	3	1	0
13	2	2	3	1	0	4
14	2	3	1	0	4	3
15	2	4	0	4	3	1
16	3	0	3	3	3	3
17	3	1	0	1	4	2
18	3	2	1	4	2	0
19	3	3	4	2	0	1
20	3	4	2	0	1	4
21	4	0	4	4	4	4
22	4	1	3	0	2	1
23	4	2	0	2	1	3
24	4	3	2	1	3	0
25	4	4	1	3	0	2

Hence, every row represents a particular configuration of the k input parameters (the equivalent of an individual in a GA approach). By using OAs, a fractional factorial research is performed, in opposition to the full factorial strategy, which requires a complete search over the parameters space. In the present

example, with 5 possible levels for the 6 parameters, a full factorial strategy would require $5^6 = 15625$ experiments, while the fractional design here outlined only involves the evaluation of 25 configurations. It is possible to demonstrate that optimum results obtained from the fractional design are statistically close to that of the full one [50].

An interesting property of OAs is that if a sub-array $N \times k'$ ($k' < k$) is extracted from the $N \times k$ array $\mathrm{OA}(N, k, s, t)$, the result is still an OA, indicated as $\mathrm{OA}(N, k', s, t')$, where $t' = \min\{k', t\}$. This property is very important from a practical point of view, because OAs databases are available [51], and it is simpler to select an array from an existing database and to adapt it than building an *ad-hoc* one. As a final remark, numerous techniques are known for constructing OAs: Galois fields [50] turn out to be a powerful tool for the construction of OAs, and several methods are proposed using such fields and finite geometries. In addition, since there is a close relation between OAs and coding theory, many construction techniques for OAs are proposed based on error-correcting codes.

2.2 Single-Objective Taguchi's Method

In this Section Taguchi's iterative method is briefly presented. Single-objective Taguchi's method (SO-TM) is illustrated in Fig. 2.1 and comprises the following steps:

2.2.1 Initialization

The optimization procedure starts with the problem initialization, where a suitable cost function is defined and a proper OA is selected from [51]: the number of columns k must be greater or equal to the number of parameters to optimize; if it is greater some columns will be discarded. The number of levels s that a parameter can assume is crucial, because a larger value will lead to better results, but also to a larger number of experiments. Typical values are 3 for simpler problems and 5 for more sophisticated ones [38]. The strength of the OA t indicates the interaction among different parameters, and is usually 2 or 3 [38]. Increasing the strength of the OA increases the number of experiments too.

2.2.2 Mapping

Next, the input parameters need to be selected. Input parameters can assume values within specified optimization ranges $[L_i : U_i]$, with $i = 1, \ldots, k$, possibly different for each of them. OA's levels need hence to be mapped to the corresponding parameters' values. This is done by first assigning the centre of the optimization range $(L_i + U_i)/2$ to the central level (in the case $S = \{0, 1, 2, 3, 4\}$ the central level is 2); other values are computed by adding and subtracting to

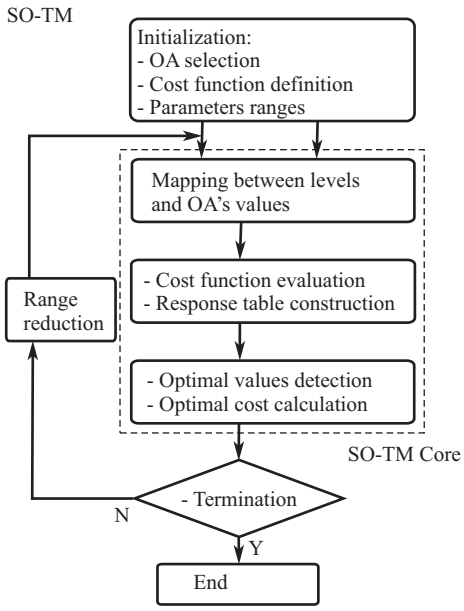


Figure 2.1: Flow chart illustrating single objective Taguchi's method (SO-TM) steps. The entire procedure, enclosed in the dashed box, represents the core of the proposed MO-TM optimization (Fig. 2.2 and Section 2.3).

the central value a quantity Δ_i named level difference, defined as follows:

$$\Delta_i = \frac{U_i - L_i}{s+1} \tag{2.1}$$

In this way a uniform distribution of levels across the optimization range is achieved.

2.2.3 Cost Function Evaluation and Response Table

After determining the input parameters, and after mapping, the cost function is evaluated for every row of the OA, leading to $n = 1, \ldots, N$ cost values c_n. Costs are converted to equivalent signal-to-noise ratios (SNR) via:

$$\mathrm{SNR}_n = -20 \log_{10}(c_n) \tag{2.2}$$

In this way, a small cost value results in a large SNR ratio. A response table is then created by averaging SNRs for each parameter $h = 1, \ldots, k$ and for each level $m = 1, \ldots, s$:

$$\overline{\mathrm{SNR}}(m,h) = \frac{s}{N} \sum_{\substack{\text{over } z \\ \text{such that} \\ \mathrm{OA}(z,h)=m}} \mathrm{SNR}_z. \tag{2.3}$$

Practically, for each parameter, SNRs values deriving from equal values of the same parameter are averaged together, thus leading to a $s \times k$ table.

2.2.4 Optimal Values Detection

Optimal values for the current iteration are extracted from the response table: in the minimization case, for each column (each parameter) the largest SNR is selected. The corresponding level, and the relative value, is the optimal for that parameter. The combination of optimal values for each parameter identifies a point $P^{(j)}$ in the parameter space which is considered the *optimum* of the current j-th iteration.

Since the OA analysis is a fractional factorial search over the parameter space, the optimal combination just found has a high chance of not having been included in the iteration's mapping table. Therefore the cost function needs to be evaluated for the parameters' optimal combination; the corresponding cost value $c^{(j)}$ is the *optimum cost* of the current iteration.

2.2.5 Range Reduction

At the first iteration ($j = 1$), each parameter's central levels are mapped to central values of the optimization range and level difference is given by (2.1). For the $(j + 1)$ iteration, central levels are mapped to previous iteration's optimal values $P^{(j)}$, and level differences are reduced, to refine search and ensure convergence. The level difference reduction takes place as:

$$\Delta_i^{(j+1)} = r\Delta_i^{(j)} = r^j\Delta_i \tag{2.4}$$

where subscripts indicates the parameter and superscripts the iteration, Δ_i being the original level difference computed when mapping was performed the first time, and $r \in (0, 1)$ being a constant. The larger r is, the slower the convergence rate. If $\Delta_i^{(j)}$ is a large value, the corresponding values of extreme levels may reside outside the optimization range: therefore, a boundary check is necessary to guarantee that all level values are located within the optimization range. In that case, boundary values are assigned as extreme levels.

2.2.6 Termination

The iterative procedure repeats points from 2.2.2 to 2.2.5 and stops when one of the following criteria is met:

- *Stop on level difference* - if the level difference $\Delta_i^{(j)}$ becomes too small, levels tend to overlap, hence returning very similar cost values. The termination criterion is:

  $$\frac{\Delta_i^{(j)}}{\Delta_i} \leq \epsilon_\Delta \tag{2.5}$$

  for a predefined small value ϵ_Δ;

- *Stop on stall* - if the difference between successive iterations' cost becomes very small, and this is maintained over a predefined number J_s of itera-

tions, the procedure stops. The termination criterion is:

$$\left|\frac{c^{(j+g)} - c^{(j)}}{c^{(j)}}\right| \leq \epsilon_s \quad \forall g \in \{1, 2, 3, \ldots, J_s\} \tag{2.6}$$

for a predefined small value ϵ_s;

- *Stop on maximum number of iterations* - the algorithm stops if, after a predefined number $J_{\max}$ of iterations, none of the other criteria has been verified.

2.2.7 Random and Semi-Random variants

Taguchi's procedure as described here is fully deterministic. It is well known that deterministic methods are likely to fall in local minima, and that their effectiveness can depend on the search starting point.

Stochastic methods on the other hand have very good global search capabilities, hence a random variant of the original TM has been first proposed in [38]. A semi-random variant too, which is described in the following, is proposed here. Both variants still rely on the previous iteration's optimal point $P^{(j)}$ as the search area central point for the $(j+1)$-th iteration. The random algorithm exploits stochastic functions to map the OA levels, without the use of $\Delta_i^{(j)}$ nor of r. This is done first by assigning the central level to the previous iteration's optimal point $P^{(j)}$, and then by dividing the optimization range in $(s-1)$ sub-intervals. Each level is then mapped with a random point located in one of these sub-intervals, hence leading to s non-overlapping mappings. The semi-random method instead maps levels as in the deterministic TM (using $\Delta_i^{(j)}$ and r), but a random shift, proportional to the magnitude of the level difference, is added to each value. With the introduction of randomness, the algorithm increases his capability of avoiding local minima, the trade-off being the increased number of iterations generally needed for convergence [38].

2.3 Multi-Objective Taguchi's Method

Taguchi's method described in the previous Section and presented in literature is a single objective technique; in many practical engineering design problems such as planar arrays, horn antennas, frequency selective surfaces and many others, more than one objective may be present [46], [52]. In these situations, either scalarization (Section 1.1.1) is used, or a multi-objective optimization algorithm needs to be implemented, as pointed out in Section 1.1. As anticipated, the former technique is usually very critical in the choice of cost combination, while the latter has the advantage of leading to a pool of optimal solutions from which the designer can choose. At each iteration, only non-dominated solutions are selected to form the optimal Pareto set, as described in Section 1.1.2. The non-dominance sorting involves an implicit comparison among solutions at

each iteration. Taguchi method, in all of its variants, produces only one optimal solution at each iteration, so an extension of this algorithm to a multiple optimization problem is not straightforward. On the other hand evolutionary techniques like GA and PSO, operating on populations, provide natively a large set of possible solutions at each iteration. Taking inspiration from this characteristics, a multi-objective Taguchi's method (MO-TM) is here proposed. The core of the optimization procedure is the single-objective procedure (SO-TM) described in the previous Section 2.2, but a strategy to preserve a set of possible solutions at each iteration, hence constructing a Pareto set, has been added. The MO-TM flow chart is presented in Fig. 2.2, while a step by step description follows.

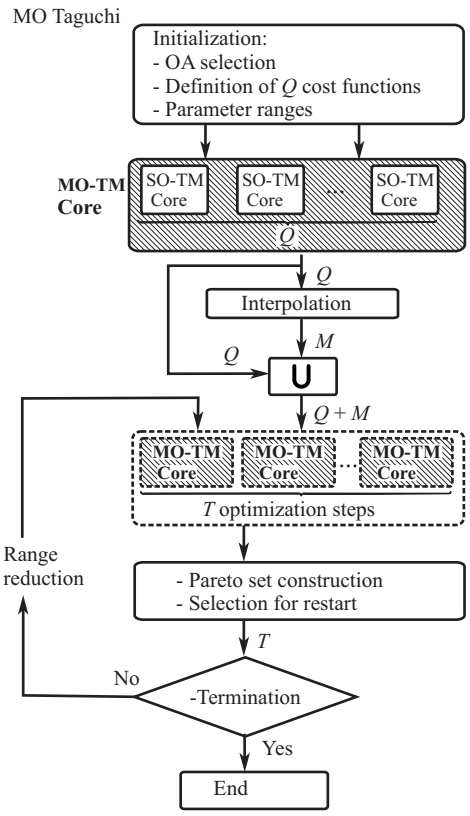


Figure 2.2: Flow chart illustrating multi-objective Taguchi's method (MO-TM) steps. SO-TM cores correspond to the dashed box of Fig. 2.1, and together they constitute MO-TM cores (hatched boxes).

2.3.1 Initialization

The initialization phase involves, like for the single-objective one, the selection of a proper OA, based on the number of parameters to optimize and on the complexity of the problem. As a rule of thumb, for a given parameter space, more complex OAs are generally needed for multi-objective optimization with respect to the single-objective one. Then the definition of $Q \geq 2$ cost functions to be minimized at the same time is performed. Levels are then mapped to their initial corresponding parameters' values for the first iteration, and initial level difference Δ_i is defined.

2.3.2 Separate Optimization

A SO-TM core step is executed for each design objective separately, hence leading to Q distinct solutions, one for each cost. These solutions are shown in Fig. 2.3 - in a simple $Q = 2$ case used for the sake of clarity - as filled squares. The Q SO-TM are conceptually grouped together and will be the MO-TM Core (reported as hatched boxes in Fig. 2.2)

2.3.3 Interpolation

An interpolation is then performed among the Q solutions found by the SO-TM algorithms, to find M new points (crosses in Fig. 2.3), which hopefully will be near the Pareto front. These interpolated points will be considered, together with the original Q ones, as starting points to continue the iteration. The interpolation is executed only once, at the very beginning of the procedure. The value of M doesn't strongly affect the outcome of the algorithm for what concerns bi-objective optimizations. A value of M greater then 8 helps to have a sufficient number of points to start the procedure from, hence $M = 10$ has been chosen for the $Q = 2$ problem.

2.3.4 Main Optimization

Steps from 2.3.1 to 2.3.3 are executed only at startup, producing $Q+M$ starting points for the iterative procedure.

On the first iteration the algorithm executes a step starting from these solutions and performs a MO-TM Core optimization on each of them, producing $(Q + M) \times Q$ solutions. Fig. 2.3 reports these additional first-step solutions as empty boxes; arrows shows, for a limited number of original points, which new ones derive.

2.3.5 Pareto Front Construction

The new, large, set of possible optimal solutions is ranked via a non-dominated sorting. The non-dominated solutions are kept as starting points for next iterations, while the dominated ones are discarded.

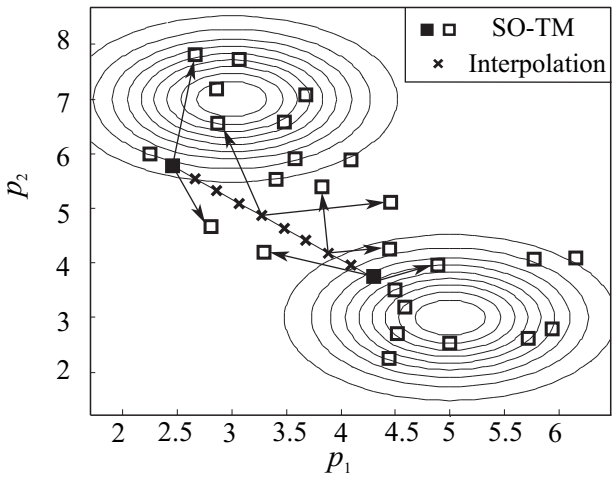


Figure 2.3: Multi-objective Taguchi's method - example of parameter space, obtained with two Gaussian functions as costs, on the first iteration. Filled boxes and crosses correspond to points found from steps 2.3.1 to 2.3.3, while empty boxes correspond to points found in 2.3.4. Contour levels are also shown.

2.3.6 Selection for Restart

If at each iteration the algorithm should run a MO-TM Core for every solution in the Pareto set, the problem complexity would dramatically increase. To avoid that, at every iteration, the algorithm restart its search procedure only on a reduced set of the Pareto set, comprising at most T solutions. Restart solutions are selected as the farthest among each other, implementing the previously cited MaxiMin selection [34]. In this way the search should have a better coverage of the cost space, leading to a more uniform front. The algorithm restarts from these reduced set for the next iterations, while the remaining non dominated solutions are kept in memory for further comparisons, until they will be eventually discarded if dominated. This strategy is necessary because, as a matter of fact, the search procedure is far more time/memory consuming with respect to a simple comparison made during the construction of a Pareto front. In this way the complexity of the problem is kept under control because, at each iteration, at most T MO-TM Cores are executed. Different experiments have been conducted, showing that $T < 10$ leads to poor results. With $T = 10$, results were quite good in terms of convergence, but in some cases they were not good for what concerns spread over the front: for some complicated problems, Pareto sets appeared clustered. By choosing $T \in (10, 20]$ this issue has been resolved and Taguchi algorithm has been able to achieve a correct spreading. Obviously if the number of Pareto set solutions is less than T, all solutions are considered as new starting points.

Chapter 3

Results and Benchmark Comparison

The proposed multi-objective Taguchi's method (MO-TM) algorithm has been applied first to a well-known set of test functions, and then to some practical array synthesis problems ([P2], [P3]). Array synthesis problems will be the synthesis of a concentric circular antenna array for satellite applications and a multi-beam array complying to different radiation pattern masks. To try out the proposed MO-TM capabilities, a comparison is made with a well established MO-GA algorithm, the NSGA-II [33]. In all cases performances of the proposed algorithm over the MO-GA benchmark will be better or equal. Without any loss of generality bi-objective ($Q = 2$) and tri-objective problems ($Q = 3$) will be considered: this will lead to easily comprehensible 2D and 3D Pareto fronts. To ensure a fair comparison, the total number of cost functions evaluations has been kept equal among TM variants and GA. The number of cost function evaluations in MO-TM depends on the OA choice: since the first iterations might occur on less than T points, the three variants of MO-TM are hence run on a predetermined number of iterations and the total number of cost functions evaluations averaged together; this number is finally used, together with the number of iterations, to determine the size of the GA population. An equal number of generations for GA and MO-TM is chosen, and the number of cost function evaluations in MO-TM has been divided by the number of iterations, giving the NSGA-II population size.

NSGA-II parameters are the following. GA's crossover function is *Scattered*: a random binary vector is created and the genes where the vector is a 1 are selected from the first parent, while the genes where the vector is a 0 are selected from the second parent. Crossover fraction is set to 0.8. Mutation is *Adaptive Feasible*, which randomly generates directions to be adaptive with respect to the last generation. The feasible region is bounded by the constraints.

Selection of the most performing algorithm has been made by running MO-TM, exploiting SO-TM in its three variants (deterministic (D), random (R)

and semi-random (S)), and NSGA-II. For each algorithm A, and for each variant V, the output consists of a Pareto set $H_A^{[V]}$ containing the non-dominated solutions, whose size will be indicated as $|H_A^{[V]}|$. Once all algorithms are executed a new set $H = \{H_{GA}, H_{MO-TM}^{[D]}, H_{MO-TM}^{[R]}, H_{MO-TM}^{[S]}\}$ can be constructed. H is not in general a Pareto set. A non-dominated sorting over H generates a new, smaller, set $\bar{H}$ which is a Pareto set. If track is kept of the algorithm which generated the members of $\bar{H}$ it is possible to write $\bar{H} = \{\bar{H}_{GA}, \bar{H}_{MO-TM}^{[D]}, \bar{H}_{MO-TM}^{[R]}, \bar{H}_{MO-TM}^{[S]}\}$. Performances of each algorithm are hence assessed on the basis of different metrics, described in the next Section 3.1. The comparison, as previously mentioned, has taken place over two set of problems:

1. test functions, mainly deriving from well known test toolboxes have been used as MO-TM benchmark. Functions deriving from the Zitler - Deb - Thiele's (ZDT) [35], MOP [53] and Tanaka [54] test suites have been chosen.

2. practical planar antenna array synthesis problems, dependent on an high number of parameters. Problems like this offer the algorithm a chance to prove its capabilities over real design situations.

3.1 Metrics

Some of the metrics used to assess the algorithms' performances are original ($d^{\%}$, $D^{\%}$), while the remaining (GD, SS, SP) can be found in [55]–[57]. The first set of metrics is intended to measure the algorithms' ability to produce solutions close to the global Pareto front.

1. $d^{\%}$, percentage of non-dominated solutions produced by an algorithm with respect to all the solutions produced by the algorithm itself.

$$d^{\%} = |\bar{H}_A^{[V]}| \; / \; |H_A^{[V]}| \; \times 100; \tag{3.1}$$

2. $D^{\%}$, percentage of non-dominated solutions produced by an algorithm with respect to the total number of non-dominated solutions produced by all algorithms.

$$D^{\%} = |\bar{H}_A^{[V]}| \; / \; |\bar{H}| \; \times 100; \tag{3.2}$$

3. Generational Distance (GD) [55]–[57], which measures the distance between the algorithm Pareto set and the true Pareto front G, whose size is $|G|$. This metric has been evaluated only for test functions, which have an analytically known Pareto front.

$$GD = \left(\sum_{i=1}^{|H_A^{[V]}|} {}^{E} d_i^2 \right)^{1/2} \Big/ \; |H_A^{[V]}|, \tag{3.3}$$

where ${}^{E}d_i$ is the Euclidean distance between an algorithm's solution and its nearest member in G:

$$ {}^{E}d_i = \min_{k \in G} \sqrt{\sum_{q=1}^{Q} \left({}^{H}c_q^{(i)} - {}^{G}c_q^{(k)} \right)^2}, \tag{3.4} $$

and ${}^{H}c_q^{(i)}, {}^{G}c_q^{(k)}$ are the q-th cost function value of the i-th and k-th member of $H_A^{[V]}$ and G. A small GD value indicates that the Pareto set is close to the Pareto front.

The second set of metrics is instead intended to measure the spread of the solutions; an uniform spreading over the front is desirable.

4. Schott Spacing (SS) [55], which measures the standard deviation of distances between neighbouring solutions:

$$ SS = \sqrt{\frac{1}{|H_A^{[V]}|} \sum_{i=1}^{|H_A^{[V]}|} \left({}^{s}d_i - \bar{d} \right)^2}, \tag{3.5} $$

where ${}^{s}d_i$ is the city block distance between neighbouring elements and $\bar{d}$ is its mean value

$$ {}^{s}d_i = \min_{k \neq i} \sum_{q=1}^{Q} |{}^{H}c_q^{(i)} - {}^{H}c_q^{(k)}| \ , \ \bar{d} = \sum_{i=1}^{|H_A^{[V]}|} \frac{{}^{s}d_i}{|H_A^{[V]}|}. \tag{3.6} $$

A small SS value indicates that distances between consecutive solutions are similar, hence the distribution on the Pareto set tends to be uniform.

5. Spread D_Δ [55], given by

$$ D_\Delta = \left[\sum_{q=1}^{Q} d_q^e + \sum_{i=1}^{|H_A^{[V]}|} |d_i - \bar{d}| \right] \Big/ \left[\sum_{q=1}^{Q} d_q^e + |H_A^{[V]}| \bar{d} \right], \tag{3.7} $$

where d_i can be ${}^{s}d_i$ or ${}^{E}d_i$, and $\bar{d}$ represents its mean. Instead d_q^e is the distance between the extreme solutions (i.e. the ones corresponding to extreme values of one of the cost) of $H_{A[V]}$ and G. In the case of an analytically unknown Pareto front, $d_q^e = 0$. This metric approaches 0 as the distribution on the front approaches a uniform one.

3.2 Test functions

The first performances assessment is conducted over functions deriving from the ZDT [35] and MOP [53] test toolboxes, plus the Tanaka function [54]. These

functions include unimodal, multimodal and separable functions leading to convex, concave and discontinuous fronts. To clearly explain the analysis, test parameters are expressed introducing the same terminology as in [56]. Given

$$\mathbf{x} = \{x_1, \dots x_n\} = \{x_1, \dots, x_j, x_{j+1,} \dots x_n\},$$

a bi-objective optimization is intended to minimize/maximize

$$\begin{cases} f_1(\mathbf{y}) \\ f_2(\mathbf{y}, \mathbf{z}) = g(\mathbf{z})h(f_1(\mathbf{y}), g(\mathbf{z})) \end{cases}$$

where $\mathbf{y} = \{y_1, \dots, y_j\} = \{x_1, \dots, x_j\}$ are j position parameters and $\mathbf{z} = \{z_1, \dots, z_k\} = \{x_{j+1}, \dots, x_n\}$ are k distance parameters, where $n = j + k$. Position parameters are responsible for the position of solutions over a same front, whereas distance parameters determine solutions belonging to different fronts, hence with different distances from the optimal front. $f_1(\mathbf{y})$ is a distribution function, which tests an algorithm's capability of diversification of elements along the local Pareto front, $g(\mathbf{z})$ is a distance function, intended to determine an algorithm's solution set's distance from the global Pareto front, and $h(f_1, g)$ is a shape function, which determines the shape of the local Pareto front. Test functions analyzed in this text are the following:

1. **ZDT1** [35], characterized by a convex front, f_1 unimodal and separable, f_2 unimodal and separable

$$f_1 = y_1;\ g = 1 + 9\sum_{i}^{k} z_i/k;\ h = 1 - \sqrt{f_1/g} \tag{3.8}$$

   where $n = 30, j = 1, k = 29$ and $x_i \in [0, 1]$.

2. **ZDT2** [35], characterized by a concave front, f_1 unimodal and separable, f_2 unimodal and separable

$$f_1 = y_1;\ g = 1 + 9\sum_{i}^{k} z_i/k;\ h = 1 - (f_1/g)^2 \tag{3.9}$$

   where $n = 30, j = 1, k = 29$ and $x_i \in [0, 1]$;

3. **ZDT3** [35], characterized by a disconnected convex front, f_1 unimodal and separable, f_2 multimodal and separable

$$\begin{aligned} &f_1 = y_1;\ g = 1 + 9\sum_{i}^{k} z_i/k; \\ &h = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1) \end{aligned} \tag{3.10}$$

   where $n = 30, j = 1, k = 29$ and $x_i \in [0, 1]$;

4. **ZDT6** [35], characterized by a concave and non uniform front, f_1 multimodal and separable, f_2 multimodal and separable

$$f_1 = \exp(-4y_1)\sin^6(6\pi y_1);\ g = 1 + 9(\sum_i^k z_i/k)^{0.25}$$
$$h = 1 - (f_1/g)^2 \tag{3.11}$$

where $n = 10, j = 1, k = 9$ and $x_i \in [0, 1]$;

5. **MOP3** [53], characterized by a disconnected front, f_1 multimodal and non-separable, f_2 multimodal and separable.

$$\begin{aligned} f_1 &= -1 - (A_1 - B_1)^2 - (A_2 - B_2)^2 \\ f_2 &= -(y+3)^2 - (z+1)^2, \end{aligned} \tag{3.12}$$

where

$$\begin{aligned} A_1 &= 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2 \\ A_2 &= 1.5\sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2 \\ B_1 &= 0.5\sin y - 2\cos y + \sin z - 1.5\cos z \\ B_2 &= 1.5\sin y - \cos y + 2\sin z - 0.5\cos z \end{aligned}$$

and $n = 2, j = 1, k = 1$ and $x_i \in [-\pi, \pi]$. This function has been maximized.

6. **TNK** [54], characterized by a disconnected front, c_1 unimodal and separable, c_2 unimodal and separable, This function does not come from [35] and hence does not follow the same structure, but rather c_1 and c_2 are defined independently:

$$\begin{aligned} c_1(\mathbf{x}) &= x_1 \\ c_2(\mathbf{x}) &= x_2 \end{aligned} \tag{3.13}$$

with

$$\begin{aligned} x_1^2 + x_2^2 - 1 - \frac{1}{10}\cos\left[16\tan^-1\left(\frac{x_1}{x_2}\right)\right] &\geq 0 \\ (x_1 - 0.5)^2 + (x_2 - 0.5)^2 &\leq 0.5 \end{aligned} \tag{3.14}$$

and $N = 2$ and $x_i \in [0, 1]$.

The test strategy is the following: first a test over 500 iterations is set for both GA and TM, in all of its variants. Variable parameters are $T \in [10, 20]$ and $r \in [0.8, 0.95]$. On the basis of the results of this preliminary analysis, values for T and r are chosen as the ones leading to better results in terms of convergence to the true front and spread over the algorithm's set. With these parameters, a $N_t = 10$ number of independent runs of each algorithm is conducted over $N_i = 200$ iterations (i.e. generations for GA), to ensure consistency; mean and standard deviation of each performance metric over these repeated runs are finally reported. MO-TM algorithms are generally capable of finding

Table 3.1: Optimization parameters for test functions.

	T	r	N_v	OA	N_i	f	GA_p	N_t
ZDT1	20	0.95	30	$(81, 40, 3, 2)$	200	328071	1640	10
ZDT2	15	0.9	30	$(81, 40, 3, 2)$	200	246454	1232	10
ZDT3	20	0.9	30	$(81, 40, 3, 2)$	200	324613	1623	10
ZDT6	15	0.9	10	$(50, 11, 5, 2)$	200	152412	762	10
MOP3	20	0.9	2	$(9, 4, 3, 2)$	200	42958	214	10

a larger number of solutions with respect to MO-GA. This is a plus, because it allows the designer to choose among a larger number of possibilities; because of that, however, a solution to provide a fair comparison for what concerns spreading over the front needs to be provided: TM solutions are decimated (and labeled with a subscript d), choosing the farthest among each other, so that $|H_{GA_d}| = |H^{[D]}_{MO-TM_d}| = |H^{[R]}_{MO-TM_d}| = |H^{[S]}_{MO-TM_d}|$. SS and D_Δ are then calculated only on decimated fronts. Table 3.1 reports, for each test function, the parameters' values used in the optimization process, deriving from the preliminary run. The proper OA is chosen accordingly to the number of parameters to optimize N_v. GA's population size GA_p is determined by dividing the cost function evaluations number f by the number of iteration N_i.

Table 3.2 reports the results of the optimization process. It is clear how Taguchi's algorithms are capable of finding a greater percentage of non-dominated solutions ($d^{\%}$ and $D^{\%}$) with respect to GA in all cases; only in one case (MOP3) the percentage of GA solutions is close to one of Taguchi's values. MO-TM outperforms GA also for what concerns the distance from the Pareto front: Taguchi's GD values are often several orders of magnitude smaller than GA's ones, confirming that TM is capable of providing a better approximation of the true front. To confront spread metrics, as previously stated, MO-TM resulting populations need to be decimated, to ensure a fair comparison with GA. A comparison among GA's SS and SP and TM's SS_d and SP_d confirms that both sets of algorithms reach good spreading values; TM performs better in all cases except MOP3, where performances are similar to those of GA. Results hence indicate a generally better behavior of Taguchi's algorithms with respect to GA. Table 3.2 results are confirmed by Fig. 3.2, where Pareto set and fronts only for ZDT test problems are depicted; MO-TM algorithms are always capable of finding the Pareto front, whereas GA is sometimes unable to reach it. In MOP3 optimization (not reported in Fig. 3.2), GA and TM solutions are overlapping over the front, as confirmed by Table 3.2 results.

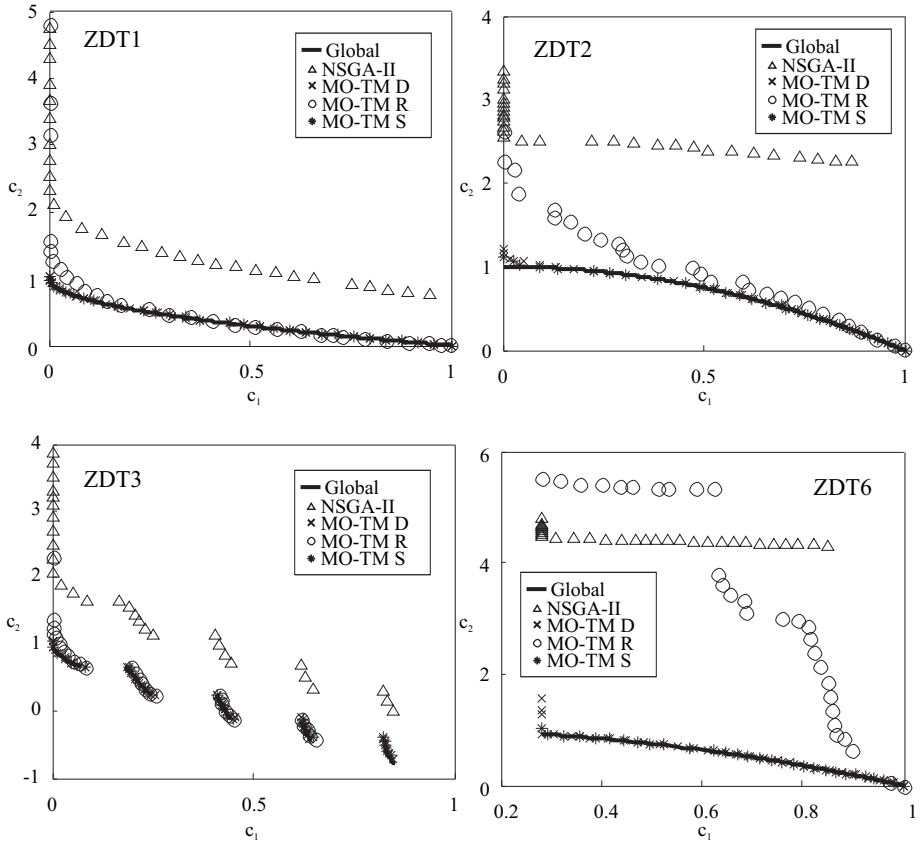


Figure 3.1: Comparison among optimization algorithms over test functions. Figures show true Pareto fronts (continuous lines) and Pareto sets (markers). Solutions have been further decimated to provide a clearer visualization.

Table 3.2: Comparison among optimization algorithms over test functions. Average values and, in parenthesis, standard deviations over $N_t = 10$ runs.

		NSGA-II	MO-TM[D]	MO-TM[R]	MO-TM[S]
ZDT1	$d^{\%}$	0.0% (0.0%)	79.8% (4.9%)	23.4% (7.7%)	77.1% (4.1%)
	$D^{\%}$	0.0% (0.0%)	48.6% (2.9%)	5.3% (2.0%)	46.1% (2.8%)
	GD	$4.8{\cdot}10^{-2}$ ($2.0{\cdot}10^{-2}$)	$7.5{\cdot}10^{-5}$ ($2.8{\cdot}10^{-5}$)	$1.8{\cdot}10^{-2}$ ($3.3{\cdot}10^{-3}$)	$8.4{\cdot}10^{-5}$ ($1.9{\cdot}10^{-5}$)
	SS_d	$1.2{\cdot}10^{-2}$ ($7.3{\cdot}10^{-3}$)	$1.4{\cdot}10^{-3}$ ($1.8{\cdot}10^{-4}$)	$9.5{\cdot}10^{-2}$ ($6.1{\cdot}10^{-2}$)	$1.5{\cdot}10^{-3}$ ($3.1{\cdot}10^{-4}$)
	SP_d	$8.9{\cdot}10^{-1}$ ($4.5{\cdot}10^{-2}$)	$3.3{\cdot}10^{-1}$ ($2.9{\cdot}10^{-1}$)	1.1 ($8.5{\cdot}10^{-2}$)	$3.4{\cdot}10^{-1}$ ($2.7{\cdot}10^{-1}$)
ZDT2	$d^{\%}$	0.0%(0.0%)	78.2%(9.7%)	6.9%(2.5%)	85.2%(5.1%)
	$D^{\%}$	0.0%(0.0%)	46.7%(4.2%)	2.8%(1.1%)	50.5%(4.3%)
	GD	$1.7{\cdot}10^{-1}$ ($7.5{\cdot}10^{-2}$)	$4.7{\cdot}10^{-4}$ ($3.2{\cdot}10^{-4}$)	$1.0{\cdot}10^{-2}$ ($2.3{\cdot}10^{-3}$)	$3.1{\cdot}10^{-4}$ ($2.5{\cdot}10^{-4}$)
	SS_d	$1.4{\cdot}10^{-2}$ ($9.4{\cdot}10^{-3}$)	$5.6{\cdot}10^{-3}$ ($9.5{\cdot}10^{-4}$)	$1.3{\cdot}10^{-2}$ ($7.9{\cdot}10^{-3}$)	$5.6{\cdot}10^{-3}$ ($2.1{\cdot}10^{-3}$)
	SP_d	$9.7{\cdot}10^{-1}$ ($3.6{\cdot}10^{-2}$)	$3.8{\cdot}10^{-1}$ ($2.1{\cdot}10^{-1}$)	$6.8{\cdot}10^{-1}$ ($2.1{\cdot}10^{-1}$)	$4.0{\cdot}10^{-1}$ ($2.4{\cdot}10^{-1}$)
ZDT3	$d^{\%}$	0.0%(0.0%)	57.8%(16.9%)	5.1%(4.8%)	67.5%(14.4%)
	$D^{\%}$	0.0%(0.0%)	44.2%(12.6%)	2.2%(2.4%)	53.6%(12.5%)
	GD	$5.9{\cdot}10^{-2}$ ($1.2{\cdot}10^{-2}$)	$2.4{\cdot}10^{-4}$ ($2.3{\cdot}10^{-5}$)	$6.2{\cdot}10^{-3}$ ($2.7{\cdot}10^{-3}$)	$2.9{\cdot}10^{-4}$ ($1.8{\cdot}10^{-4}$)
	SS_d	$1.2{\cdot}10^{-2}$ ($1.9{\cdot}10^{-3}$)	$3.8{\cdot}10^{-3}$ ($8.7{\cdot}10^{-4}$)	$1.1{\cdot}10^{-1}$ ($6.6{\cdot}10^{-2}$)	$2.7{\cdot}10^{-3}$ ($7.9{\cdot}10^{-4}$)
	SP_d	$9.0{\cdot}10^{-1}$ ($1.6{\cdot}10^{-2}$)	$3.8{\cdot}10^{-1}$ ($5.4{\cdot}10^{-2}$)	1.2 ($1.6{\cdot}10^{-1}$)	$4.5{\cdot}10^{-1}$ ($6.9{\cdot}10^{-2}$)
ZDT6	$d^{\%}$	0.2%(0.6%)	99.3%(0.7%)	15.5%(13.8%)	98.6%(1.3%)
	$D^{\%}$	0.0%(0.0%)	48.6%(1.6%)	2.2%(2.6%)	49.2%(1.6%)
	GD	$5.2{\cdot}10^{-1}$ ($1.3{\cdot}10^{-1}$)	$6.2{\cdot}10^{-4}$ ($4.0{\cdot}10^{-4}$)	$1.3{\cdot}10^{-1}$ ($3.4{\cdot}10^{-2}$)	$5.6{\cdot}10^{-4}$ ($2.7{\cdot}10^{-4}$)
	SS_d	$3.1{\cdot}10^{-2}$ ($3.2{\cdot}10^{-2}$)	$1.7{\cdot}10^{-2}$ ($1.1{\cdot}10^{-2}$)	$5.0{\cdot}10^{-2}$ ($2.6{\cdot}10^{-2}$)	$1.2{\cdot}10^{-2}$ ($8.9{\cdot}10^{-3}$)
	SP_d	$9.8{\cdot}10^{-1}$ ($9.8{\cdot}10^{-3}$)	$3.7{\cdot}10^{-1}$ ($9.4{\cdot}10^{-2}$)	$8.9{\cdot}10^{-1}$ ($6.0{\cdot}10^{-2}$)	$3.2{\cdot}10^{-1}$ ($7.4{\cdot}10^{-2}$)
MOP3	$d^{\%}$	42.1%(3.0%)	39.2%(6.0%)	55.9%(13.7%)	43.9%(9.5%)
	$D^{\%}$	7.7%(1.2%)	31.6%(5.8%)	29.5%(7.4%)	31.2%(6.7%)
	GD	$2.1{\cdot}10^{-5}$ ($2.1{\cdot}10^{-6}$)	$3.9{\cdot}10^{-4}$ ($4.0{\cdot}10^{-5}$)	$1.9{\cdot}10^{-4}$ ($2.1{\cdot}10^{-4}$)	$3.7{\cdot}10^{-4}$ ($4.9{\cdot}10^{-5}$)
	SS_d	$6.4{\cdot}10^{-4}$ ($5.9{\cdot}10^{-5}$)	$9.9{\cdot}10^{-4}$ ($6.2{\cdot}10^{-5}$)	$9.0{\cdot}10^{-4}$ ($1.6{\cdot}10^{-4}$)	$1.0{\cdot}10^{-3}$ ($8.5{\cdot}10^{-5}$)
	SP_d	$6.5{\cdot}10^{-1}$ ($5.4{\cdot}10^{-2}$)	$7.7{\cdot}10^{-1}$ ($5.8{\cdot}10^{-2}$)	$7.3{\cdot}10^{-1}$ ($1.1{\cdot}10^{-1}$)	$8.4{\cdot}10^{-1}$ ($6.6{\cdot}10^{-2}$)

3.3 Scalarization Method

To provide a set of easily comparable results with the MO techniques described above, tests have been performed on the previously introduced test functions ZDT3 and the TNK. Both are particularly difficult having disconnected Pareto Fronts, the former having only convex segments in the front, the latter having also a non-convex front.

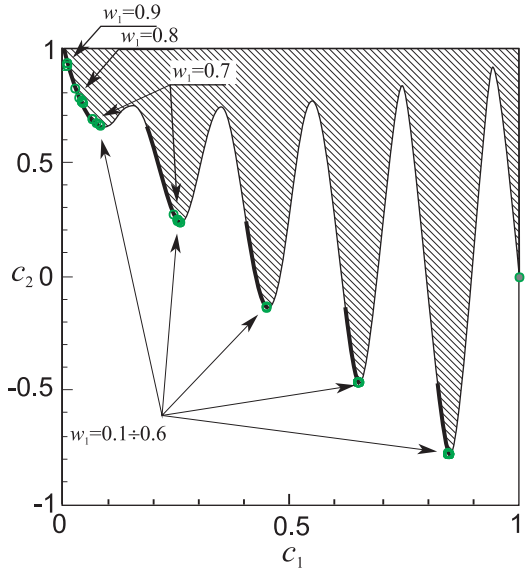
Benchmarks used here in a scalarized SO optimization produce unsatisfactory results, as foretold in earlier, but are reported so as to get a better insight on the importance of MO techniques: algorithms are tuned to perform nearly the same number of cost function evaluations. This will ensure a fair comparison among the different techniques, regardless of different implementations or different machine architectures. For the sake of simplicity, the SO GA implementation in Matlab Optimization toolbox is here exploited. The variable space dimension is 2 in both cases, and the domain is $x_i \in [0, 1]$ for $i = 1, 2$; no constraints are present, besides those intrinsically present in the functions' definitions. Population size is set to 20, maximum generations number to 1000, but the algorithms never performed so many runs, stopping on the stall criteria much before, usually at about generation 50. All other parameters are set to default.

In the first case, ZDT3, the front is not connected but is convex. A series of standard SO GA runs as described earlier are performed, varying weight values of the single-objective function $c_s(\mathbf{x})$ defined by:

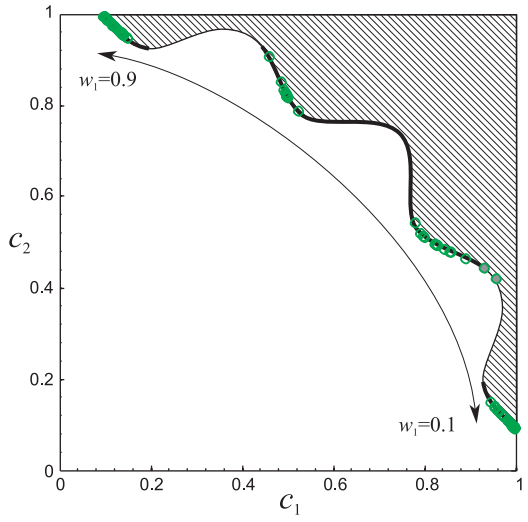
$$c_s(\mathbf{x}) = w_1 c_1(\mathbf{x}) + w_2 c_2(\mathbf{x}) \tag{3.15}$$

with $w_1 \in (0, 1)$ and $w_2 = 1 - w_1$. Ten runs for each weight value in the finite set
$w_1 \in [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$ are performed, for a total of 90 independent runs. Due to the stochastic nature of GA, runs with the same weight values do not fall exactly on the same point, as a deterministic SO optimization would, but lead to a cluster of optimal solutions (Fig. 3.2a). These clusters are clearly separated for high w_1 values, and partially superimposed otherwise, so that it would be hard to identify them on the figure; for $w_1 < 0.6$ in this test case the optimal point is located at a random position at the bottom of one of the three *valleys*. As a general rule results with small values of w_1 are clustered in the bottom right corner of the figure, while those with an high value of w_1 are clustered on the top left corner. A solution which is not on the Pareto Front (gray bullet) is also found. Please note how these 90 independent runs lead to very clustered solutions, and a very poor approximation of the Pareto Front. True MO techniques, as those described above, explore the whole front with greater efficiency in a single run.

In the second case, TNK, the front is not connected and non-convex. A series of standard single-objective GA runs with the same parameters as before are performed, varying the weight values as in the previous example and with ten independent runs for each set of weights. Again, solutions are clustered according to the chosen w_1 value and, furthermore, as expected (Section 1.1.1),



(a) ZDT3



(b) TNK

Figure 3.2: Scalarized MO problem: hatched area, cost function co-domain; thick lines on the contour, Pareto Front; empty circles, solution found by the SO technique on or close to the Pareto Front; grey filled bullets, solutions which are not Pareto-optimal.

no solutions are found in the concave part of the Pareto Front (Fig. 3.2b). Moreover, two solutions are found (gray bullets in Fig. 3.2b) which are not on, nor close to, the Pareto Front.

3.4 Array synthesis problem

A real problem is now analyzed: the design of a concentric circular antenna array (CCAA) [58], [59]. This problem has been addressed in the past using GA [60], PSO [61] and other techniques [62]. The array synthesis problem here tested is the design of a sector beam pattern. The problem's specifications are those for a geostationary satellite antenna in the Ka-band ($19.7 - 20.2$ GHz) designed to provide a multibeam coverage of Europe [63], [64]. The broadside beam should be $0.56°$ wide, with a maximum ripple equal to 3dB, and side lobe level level equal to -30dB, to be checked only in $[-10°, -0.9°] \cup [0.9°, 10°]$. The single elements composing the CCAA are chosen to have characteristics comparable to horns in [64]; in particular circular horns with a 4λ diameter are exploited. For what concerns the element field pattern, a $[\cos(\theta)]^{35}$ is used, which is a good approximation for angles closer than $\pi/5$ to broadside [64]. The optimization will focus on a sparsified array of equi-amplitude elements with the dual objective of satisfying the masks and minimizing the number of elements.

The CCAA comprises M concentric rings plus a central element [65] (Fig. 3.3). For ring m, N_m and r_m are the number of elements and the radius, respectively; element $n = 0, \ldots, N_m - 1$ is at angular position $\phi_{mn} = 2\pi n/N_m + \phi_m^{(0)}$, being $\phi_m^{(0)}$ the angular offset of the first element. Consequently the spacing between neighbouring elements is uniform and equal to $d_m = 2\pi r_m/N_m$ [59], [62].

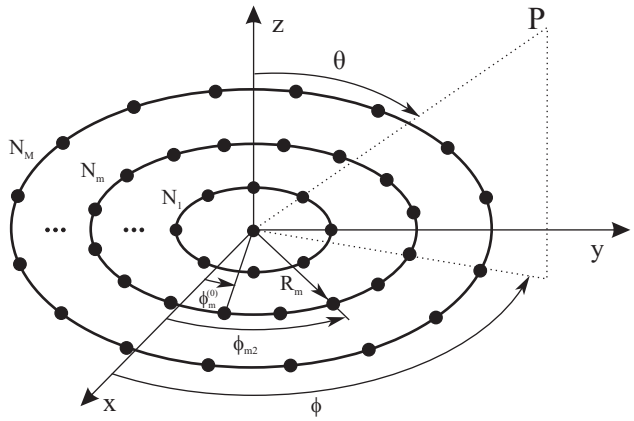


Figure 3.3: Geometry of a concentric circular antenna array (CCAA) with a central element.

The array factor for a CCAA is:

$$AF(\theta, \phi) = 1 + \sum_{m=1}^{M} \sum_{n=0}^{N_m - 1} e^{j2\pi r_m \sin\theta \cos(\phi - \phi_{mn})}, \tag{3.16}$$

Masks fitting is the first objective of this test. This is done by first defining an upper and a lower masks (Fig. 3.6); differences between the array's total

pattern and masks for every θ are calculated, and then the out-of-masks ones are minimized, leading to a value of c_{mask}; namely an array with a 0 value for this cost is perfectly fitted between masks. A CCAA has a rather symmetrical pattern; however the cost evaluation optimization is conducted for $\phi = 0°$ and $\phi = 90°$, and $c_{mask}^{(0)} + c_{mask}^{(90)}$ is the first objective of the optimization.

The second objective of the optimization is the minimization of the total number of elements the CCAA is composed of, $N_{tot} = \sum_{m=1}^{M} N_m$. Arrays with a smaller number of elements are easier to realize, more lightweight and cheaper, especially if they have uniform feeding. Being the second objective a discrete parameter, the Pareto set is going to be discontinuous.

The optimization is conducted over a CCAA composed of $M = 9$ rings; arrays with a smaller number of rings are unable to achieve similar performances as the structures developed in [63], [64]. Input parameters are ideally, $(R_m, N_m, \phi_m^{(0)})$, but since constraints on spacing to guarantee that horns 4λ in diameter are physically placeable on the designed array, a constraint to provide a correct spacing of at least 4λ between consecutive elements on the same ring and between consecutive rings is enforced. This is easier if $(\Delta R_m, N_m, \phi_m^{(0)})$ is chosen as parameter set, with $R_m = R_{m-1} + \Delta R_m$ and $R_0 = 0$.
MO-TM parameters are $T = 20$ and $r = 0.95$ (Table 3.3), and $N_t = 10$ independent runs of each algorithm are conducted over $N_i = 200$ iterations (i.e. generations for GA), to generate statistics. GA's population size GA_p is determined by dividing the cost function evaluations number f by the number of iteration N_i. Mean and standard deviation of each performance metric over these repeated runs are finally reported in Table 3.4, and Pareto sets are depicted in Fig. 3.4; MO-TM solutions have been decimated in this case too to match GA's population size. Obviously metrics like GD cannot be computed, being the Pareto front unknown. The best results are obtained with MO-

Table 3.3: Optimization parameters for the CCAA synthesis problem.

	T	r	N_v	OA	N_i	f	GA_p	N_t
CCAA	20	0.95	20	$(81, 40, 3, 2)$	200	320463	1602	10

TM[R], as confirmed by values of $d^{\%}$ and $D^{\%}$ in Table 3.4: indeed in Fig. 3.4 MO-TM[R] solutions (circles) dominate the others, especially GA's (triangles). SS and SP values indicate a similar behavior among different algorithms for what concerns spreading over the Pareto set. What really matters is that only MO-TM algorithms (Random and Deterministic in this case) have been capable of finding some sets of input parameters leading to radiation patterns comprised between masks, as pointed out in Table 3.5, whereas MO-GA designs are incapable of fulfilling this goal. Parameters composing the best synthesized CCAA are in Table 3.6: MO-TM[R] has led to a 196 elements design with a 48.2λ radius (Fig. 3.5), which is perfectly fitted between masks. The antenna pattern deriving from it is reported in Fig. 3.6. By increasing the tolerance to 5%, MO-

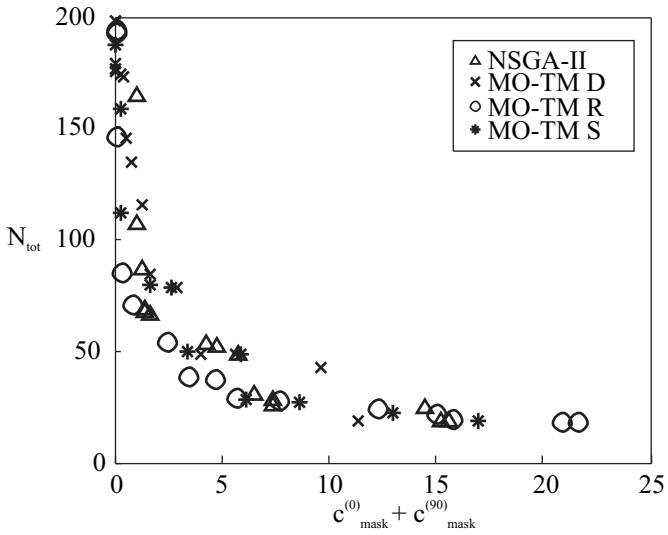


Figure 3.4: Comparison among optimization algorithms over the CCAA synthesis problems. Different markers correspond to different Pareto sets.

TM[R] is capable of significantly further decreasing the number of elements, reaching a 148 elements design, with a 47.4λ radius, as shown in Table 3.6. That allows the designer to choose between a larger array, but perfectly fitted between masks, and a more compact one, characterized by only a small fitting error. A comparison with the sunflower array in [63] is straightforward: in

Table 3.4: Comparison among optimization algorithms over the CCAA synthesis problem. Average values and, in parenthesis, standard deviations over $N_t = 10$ runs.

	NSGA-II	MO-TM[D]
$d^{\%}$	20.2% (12.2%)	11.8% (8.4%)
$D^{\%}$	20.3% (10.6%)	12.0% (8.0%)
SS_d	$6 \cdot 10^{-2}$ $(2 \cdot 10^{-2})$	$7.8 \cdot 10^{-2}$ $(2.5 \cdot 10^{-2})$
SP_d	$5.5 \cdot 10^{-1}$ $(1.3 \cdot 10^{-1})$	$6 \cdot 10^{-1}$ $(1.7 \cdot 10^{-1})$
	MO-TM[R]	MO-TM[S]
$d^{\%}$	57.2% (19.0%)	28.6% (17.5%)
$D^{\%}$	44.7% (15.3%)	23.1% (12.5%)
SS_d	$9.2 \cdot 10^{-2}$ $(3.8 \cdot 10^{-2})$	$9.3 \cdot 10^{-2}$ $(3.1 \cdot 10^{-2})$
SP_d	$6 \cdot 10^{-1}$ $(2 \cdot 10^{-1})$	$7.4 \cdot 10^{-1}$ $(1.8 \cdot 10^{-1})$

that work masks are satisfied with a $R_N = 56\lambda$ radius array composed of 250

Table 3.5: Optimizations results for a $M = 9$ CCAA.

	NSGA-II	MO-TM[D]	MO-TM[R]	MO-TM[S]
	N_{tot} $[R_N]$	N_{tot} $[R_N]$	N_{tot} $[R_N]$	N_{tot} $[R_N]$
100% fitting	–	199 $[49.9\lambda]$	196 $[48.2\lambda]$	–
95% fitting	–	180 $[47\lambda]$	148 $[47.4\lambda]$	–

Table 3.6: Design parameters deriving from the optimization for a $M = 9$ CCAA.

	0	1	2	3	4	5	6	7	8	9
MO-TM[R] 196 elements, $R_N = 48.2\lambda$, 0% masks fitting error										
N_m	1	8	14	18	23	26	30	20	28	28
R_m/λ	–	6.5	10.8	15.0	19.3	23.4	28.2	33.5	38.3	48.2
$\phi_m^{(0)}$ (deg)	–	283.2	178.6	304.7	164.0	269.0	117.5	131.6	79.9	2.6
MO-TM[R] 148 elements, $R_N = 47.4\lambda$, 5% masks fitting error										
N_m	1	6	8	17	14	22	24	24	12	20
R_m/λ	–	6.4	10.7	14.8	20.5	24.8	30.2	37.1	42.2	47.4
$\phi_m^{(0)}$ (deg)	–	192.5	345.8	260.0	263.8	103.3	257.2	272.7	338.0	95.4

elements. Hence the proposed CCAA optimizations has led to a set of structures capable of achieving the same performances as in [63] with a lower number of elements and a smaller diameter, confirming that MO-TM algorithms are a viable and convenient solution also in practical design problems.

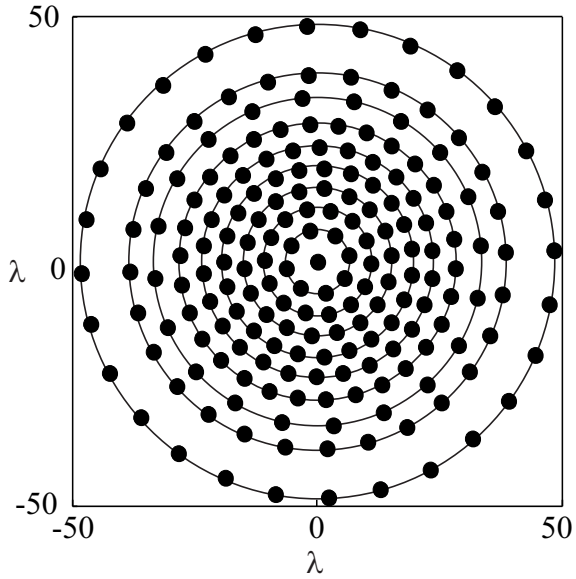


Figure 3.5: The 196 elements, $R_N = 48.2\lambda$ CCAA obtained with MO-TM[R], composed of $M = 9$ rings.

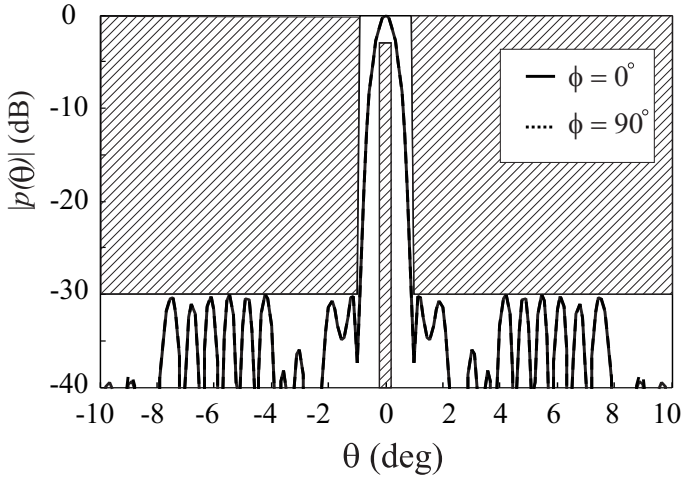


Figure 3.6: Pattern deriving from the 196 elements CCAA obtained with MO-TM[R]. The array is perfectly fitted between masks for $\phi = 0°$ (solid line) and for $\phi = 90°$ (dotted line). The pattern is practically symmetrical. Masks to be satisfied are reported as non-hatched regions.

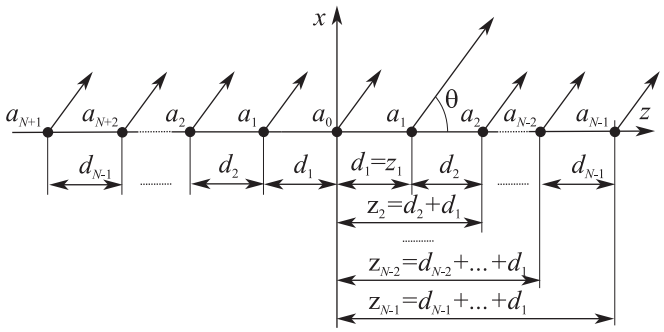


Figure 3.7: Symmetric array made of $2N-1$ elements, where the central element is the combination of two identical elements. The optimization process involves N amplitudes, N phases and $N-1$ spacings.

3.5 Array beam shaping

A seond real world ptoblem is that of multiple beam shaping. Beam shaping for single- and multi- beam antenna arrays is an interesting possibility for telecommunication devices, yet a full control over amplitudes and phases of every single element in the array leads to costly and complex feeding networks; some constraints, especially on amplitudes, would lead to much more convenient designs. In particular for multi-beam arrays, a phase-only control, where different beams are generated varying only the phase of the single array elements, would be a much cheaper solution with respect to those where also the element amplitude are changed when switching from beam to beam [66]–[68]. Being the problem non linear, a direct synthesis is impossible and optimization techniques are often exploited, yet, when several beams are to be synthesized, a single cost function encompassing all of them can be inadequate, especially if, besides beams, there are other constrains to be satisfied, for example on the maximum difference between element amplitudes. Indeed, if an array has very different excitation amplitudes among its elements, the proper design of an efficient feeding network can be tricky, while if element amplitudes are more uniformly distributed, the feeding network is easier to realize [69].

The array factor (AF) of linear symmetric array with $2N-1$ elements aligned along the z-axis (Fig. 3.7)is [68]

$$AF = 2\sum_{n=0}^{N/2} a_n \cos(2\pi \frac{z_n}{\lambda}\cos\theta + \phi_n) \tag{3.17}$$

where a_n are the excitation amplitudes, ϕ_n are the excitation phases and z_n are the element positions.

Arrays will be optimized with MO-TM to comply to different radiation pattern masks, while at the same time keeping low the difference in element amplitudes. Different patterns will be synthesized on phase-only variations, while

Table 3.7: Masks parameters for the $Q = 2$ and $Q = 3$ problems

Centre(s)	Width (Lower)	Width (Upper)	Ripple	SLL
75° and 120°	6°	20°	3dB	−23dB

maintaining the same element amplitudes. In the proposed optimization scheme, excitation amplitudes will be allowed to vary within $[0, 1]$, excitation phases within $[-\pi, \pi]$ range and element spacings $d_n = z_n - z_{n-1}$ within $[0.3\lambda, 0.9\lambda]$.

Two different multi-objective array design problems are investigated: first, a bi-objective ($Q = 2$), and then a tri-objective linear array synthesis problem ($Q = 3$). MO-TM results will be compared to those generated by the NSGA-II [33]. Comparison will be based on the metrics in Section 3.1. To ensure a fair comparison, the total number of cost functions evaluations shall be kept equal among MO-TM and MO-GA. Also here, an equal number of generations for GA and MO-TM has been chosen, and the number of cost function evaluations in MO-TM has been divided by the number of iterations, giving the corresponding NSGA-II population size.

3.5.1 Dual-beam shaping ($Q = 2$)

In this first problem two identical masks centered on different angular directions will be considered: they are reported as dashed lines in Fig. 3.8a and 3.8b and have their characteristics reported in Table 3.7. The cost function value is obtained by calculating differences between the AF and masks on a discrete set of angles (181 θ values $\in [-90, 90]$), and then summing up the absolute values of all differences. First and second masks fitting are the two objectives of this optimization.

Results are relative to a symmetric array made of 13 elements, hence the optimization algorithms work on 27 parameters (7 amplitudes, 7+7 phases, 6 spacings). The selected OA has been an $\mathrm{OA}(81, 40, 3, 2)$, and it has been reduced to an $\mathrm{OA}(81, 27, 3, 2)$. MO-TM number of iterations and GA number of generations and population size are in Table 3.8. The Pareto set is reported in Fig. 3.9a, while numerical comparison is in Table 3.9. In this case Random Taguchi's method has been the most performing algorithm, finding the largest part of the non-dominated solutions. Its set is in fact the best among other algorithms' ones in Fig. 3.9a. Its spreading is by far the smallest, hence the best. Obtained optimized amplitudes, spacings and phases are reported in Table 3.10 for one of these synthesized arrays. The excitation dynamic is 3.4. Fig. 3.8a shows the array factors obtained with each phases' set.

3.5.2 Dual-beam shaping and amplitudes spread ($Q = 3$)

In addition to first and second masks fitting, the minimization of the excitation amplitudes spread as in (3.18) has been added. The optimization has been

carried out on the same structure as in the $Q = 2$ case, hence a symmetric array made of 13 elements. MO-TM and GA parameters are the same than in the previous case (Table 3.8).

$$\frac{\max(a_n) - \min(a_n)}{\min(a_n)} \tag{3.18}$$

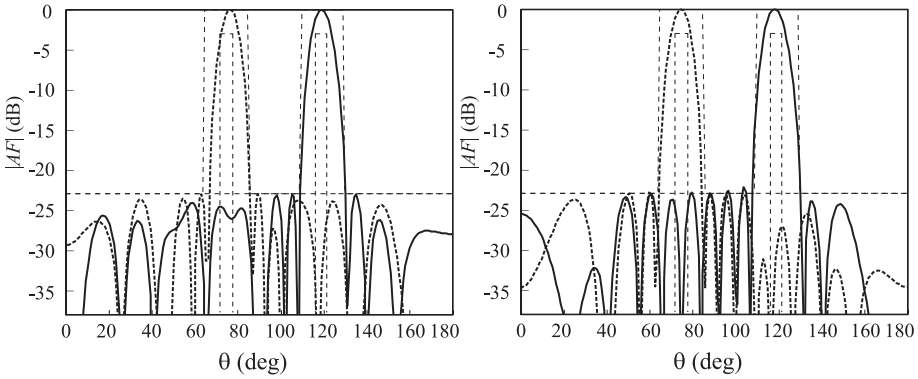
Results show that both GA and TM perform well, but only Random Taguchi's method has been capable of finding amplitude and phases capable of fitting the AF under the selected masks (Table 3.10), while maintaining a low excitation dynamic (2.4 in this case). Different views of the 3-D Pareto front are reported in Fig. 3.9b, while a numerical comparison is in Table 3.9. Random TM spreading is the smallest in this case too. In Fig. 3.8b array factors obtained with each phases' set are shown to be comprised between their relative masks.

Table 3.8: Optimization settings

	TM iter./GA gen.	Fcount	r	T	GA popsize
$Q = 2$	200	167000	0.95	10	835
$Q = 3$	200	235151	0.95	15	1175

Table 3.9: Comparison among optimization algorithms

	Algorithm	Non-dominated solutions		Spreading
		$d^{\%}$	$D^{\%}$	SS
$Q = 2$	NSGA-II	0%	0%	$15.1 \cdot 10^{-3}$
	MO-TM deterministic	18.88%	23.95%	$3.7 \cdot 10^{-3}$
	MO-TM random	100%	76.04%	$2.7 \cdot 10^{-3}$
	MO-TM semi-random	0%	0%	$3.2 \cdot 10^{-3}$
$Q = 3$	NSGA-II	93.55%	30.90%	$12 \cdot 10^{-3}$
	MO-TM deterministic	47.06%	24.60%	$10.5 \cdot 10^{-3}$
	MO-TM random	76.60%	33.40%	$8.6 \cdot 10^{-3}$
	MO-TM semi-random	35.60%	11.10%	$6.4 \cdot 10^{-3}$



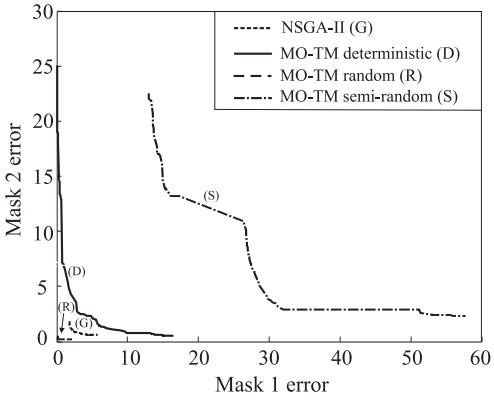
(a) Dual beam shaping ($Q = 2$)

(b) Dual beam shaping + ampl. spread ($Q = 3$)

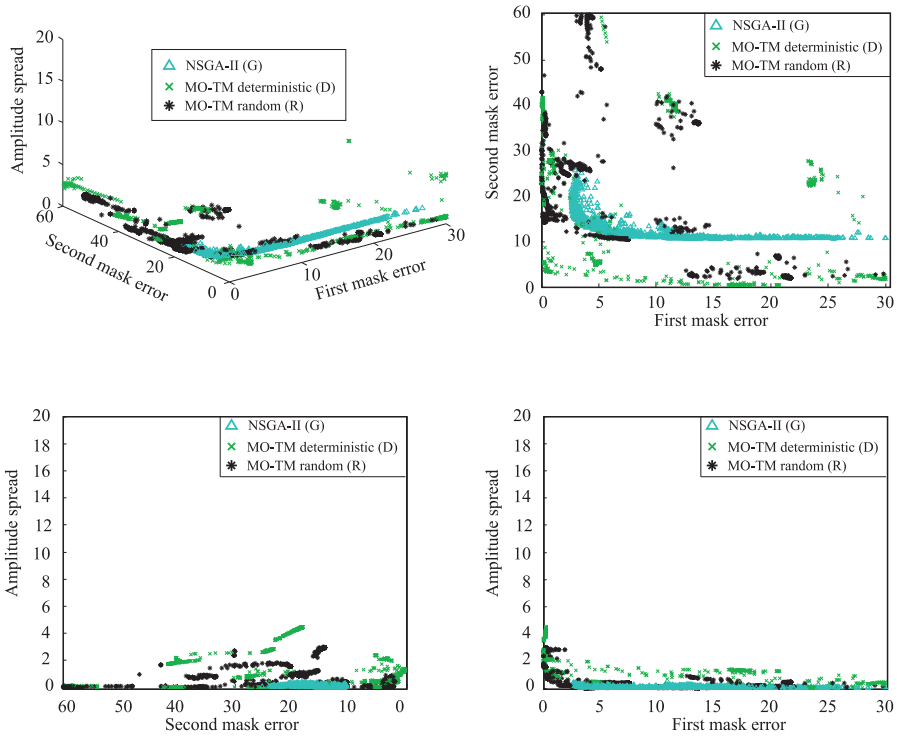
Figure 3.8: Array factor satisfying masks. Masks are reported as dashed lines.

Table 3.10: Optimized array characteristics; phases are in radiants, distances in free-space wavelengths

		elements						
		0	1	2	3	4	5	6
$Q = 2$	a_n	0.888	0.806	0.775	0.592	0.565	0.382	0.200
	ϕ_n^1	-0.371	-0.996	-1.740	-2.667	2.584	1.980	0.650
	ϕ_n^2	0.915	1.723	-2.494	-0.897	1.088	2.835	-1.693
	d_n		0.605	0.596	0.610	0.585	0.616	0.574
$Q = 3$	a_n	0.784	0.740	0.662	0.529	0.433	0.321	0.231
	ϕ_n^1	1.020	-1.014	-1.836	-2.848	2.555	1.642	1.018
	ϕ_n^2	0.985	1.750	-2.821	-1.134	0.446	2.279	3.141
	d_n		0.573	0.592	0.558	0.543	0.517	0.480



(a) Dual beam shaping ($Q = 2$)



(b) Dual beam shaping ($Q = 3$)

Figure 3.9: Cost spaces for $Q = 2$ and $Q = 3$ problems. Costs are first and second masks fitting for both problems, plus amplitudes spread minimization for the tri-objective case. Pareto sets are reported for each algorithm.

Bibliography for Multi Objective Optimization

[1] G.B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, Princeton (NJ), 1963.

[2] J. K. Lenstra, A. H. G. Rinnooy Kan, and A. Schrijver, eds. *History of Mathematical Programming: A Collection of Personal Reminiscences*. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1991.

[3] A. Cauchy. "Méthode Générale pour la Résolution des Systèms dEquations Simultanées". In: *Comptes Rendus de l Académie des Sciences* 25 (1847), pp. 46–89.

[4] H.B. Curry. "The Method of Steepest Descent for Nonlinear Minimization Problems". In: *Quarterly of Applied Mathematics* 2 (1944), pp. 258–261.

[5] M.R. Hestenes and E. Stiefel. "Methods of Conjugate Gradients for Solving Linear Systems". In: *Journal of Research of the National Bureau of Standards* 49 (1952), pp. 409–436.

[6] R. Horst and P.M. Pardalos, eds. *Handbook of Global Optimization*. Kluwer, Dordrecht (NL), 1995.

[7] R. B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer, Dordrecht (NL), 1996.

[8] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd. Springer, Berlin (D), 1996.

[9] C. Grosan and A. Abraham. "Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews". In: *Studies in Computational Intelligence* 75 (2007), pp. 1–17.

[10] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading (MA), 1989.

[11] J. Kennedy and R. Eberhart. "Particle Swarm Optimization". In: *Proc. Conf. IEEE Int Neural Networks*. Vol. 4. 1995, pp. 1942–1948. DOI: 10.1109/ICNN.1995.488968.

[12] Y. Shi and R.C. Eberhart. "A Modified Particle Swarm Optimizer". In: *Proc. IEEE International Conference on Evolutionary Computation*. 1998, pp. 69–73.

[13] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. "Optimization by Simulated Annealing". In: *Science* 220 (1983), pp. 671–680.

[14] V. Cerny. "Thermodynamical Approach to the Traveling Salesman Problem: an Efficient Simulation Algorithm". In: *Journal of Optimization Theory and Applications* 45 (1985), pp. 41–51.

[15] F. Glover and C. McMillan. "The General Employee Scheduling Problem: an Integration of MS and AI". In: *Computers and Operations Research* 13 (1986), pp. 563–573.

[16] F. Glover. "Tabu Search - Part 1". In: *ORSA Journal on Computing* 1 (1989), pp. 190–206.

[17] F. Glover. "Tabu Search - Part 2". In: *ORSA Journal on Computing* 2 (1990), pp. 4–32.

[18] L.A. Rastrigin. "The Convergence of the Random Search Method in the Extremal Control of a Many Parameter System". In: *Automation and Remote Control* 24 (1963), pp. 1337–1342.

[19] D.C. Karnopp. "Random Search Techniques for Optimization Problems". In: *Automatica* 1 (1963), pp. 111–121.

[20] A. Colorni, M. Dorigo, and V. Maniezzo. "Distributed Optimization by Ant Colonies". In: *Actes de la Première Conférence Européenne sur la Vie Artificielle*. 1991, pp. 134–142.

[21] A.R. Mehrabian and C. Lucas. "A Novel Numerical Optimization Algorithm inspired from Weed Colonization". In: *Ecological Informatics* 1 (2006), 355366.

[22] S. Karimkashi and A.A. Kishk. "Invasive Weed Optimization and its Features in Electromagnetics". In: *Antennas and Propagation, IEEE Transactions on* 58 (2010), pp. 1269–1278.

[23] S.D. Muller, J. Marchetto, S. Airaghi, and P. Kournoutsakos. "Optimization based on Bacterial Chemotaxis". In: *Energy Conversion, IEEE Transactions on* 6 (2002), pp. 16–29.

[24] C.K. Goh and K.C. Tan. *Evolutionary Multi-objective Optimization in Uncertain Environments: Issues and Algorithms*. Springer, New York (NY), 2009.

[25] C. A. Coello Coello, G. B. Lamont, and D.A. Van Veldhuisen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York (NY), 2007.

[26] J. Branke, K. Deb, K. Miettinen, and R. Slowinski. *Multiobjective Optimization: Interactive and Evolutionary Approaches*. Springer, New York (NY), 2008.

[27] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston (MA), 1999.

[28] F.Y. Edgeworth. *Mathematical Psychics*. P. Keagan, London (UK), 1881.

[29] V. Pareto. *Manuale di Economia Politica*. Societa Editrice Libraria, Milan (I), 1906 [Eng. ed by A.S. Schwier *Manual of Political Economy*, Macmillan, New York (NY), 1971].

[30] C.M. Fonseca and P.J. Fleming. "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms Part I and II". In: *Systems, Man, and Cybernetics, IEEE Transactions on* 28 (1997), 26–37 and 38–47.

[31] I.Y. Kim and O.L. de Weck. "Adaptive Weighted-sumMethod for Bi-objective Optimization: Pareto Front Generation". In: *Structural and Multidisciplinary Optimization* 29 (2005), pp. 149–158.

[32] O.L. de Weck. "Multiobjective Optimization: History and Promise". In: *Proc. of The Third China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems*. 2004.

[33] K. Deb, A. Pratap, S. Agarwal, and T. Meyariva. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II". In: *Evolutionary Computation, IEEE Transactions on* 6 (2002), 182197.

[34] S.I. Valdez Peña, S. Botello Rionda, and A. Hernandez Aguirre. "Improved MaxiMin Selection for well spread Pareto Fronts". In: *Comunicaciones del CIMAT* I-07-02 (2007).

[35] E. Zitzler, K. Deb, and L. Thiele. "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results". In: *Evolutionary Computation* 8 (2000), pp. 173–195.

[36] G. Taguchi, S. Chowdhury, and Y. Wu. *Taguchi's Quality Engineering Handbook*. John Wiley & Sons Inc., Hoboken (NJ), 2005.

[37] W. C. Weng, F. Yang, and A. Z. Elsherbeni. "Linear Antenna Array Synthesis Using Taguchi's Method: A Novel Optimization Technique in Electromagnetics". In: *Antennas and Propagation, IEEE Transactions on* 55.3 (2007), pp. 723–730. DOI: 10.1109/TAP.2007.891548.

[38] W. C. Weng, F. Yang, and A. Z. Elsherbeni. *Electromagnetics and Antenna Optimization using Taguchi's Method*. Morgan & Claypool, CA, 2007.

[39] W. C. Weng, F. Yang, V. Demir, and A. Z. Elsherbeni. "Optimization using Taguchi Method for Electromagnetic Applications". In: *Proc. First European Conf. Antennas Propag. EuCAP 2006*. 2006, pp. 1–6. DOI: 10.1109/EUCAP.2006.4584773.

[40] W. C. Weng, F. Yang, V. Demir, and A. Z. Elsherbeni. "Electromagnetic Optimization using Taguchi Method: a case study of Linear Antenna Array Design". In: *Proc. IEEE Antennas Propag. Soc. Int. Symp. 2006*. 2006, pp. 2063–2066. DOI: 10.1109/APS.2006.1710987.

[41] D. W. Boeringer and D. H. Werner. "Particle Swarm Optimization versus Genetic Algorithms for Phased Array Synthesis". In: *Antennas and Propagation, IEEE Transactions on* 52.3 (2004), pp. 771–779. DOI: 825102.

[42] C. H. Liu, Y. L. Chen, and J. Y. Chen. "Ameliorated Particle Swarm Optimization by Integrating Taguchi Methods". In: *Proc. Int Mach. Learn. and Cybern. (ICMLC) Conf.* Vol. 4. 2010, pp. 1823–1828. DOI: 10.1109/ICMLC.2010.5580960.

[43] J. T. Tsai, T. K. Liu, and J. H. Chou. "Hybrid Taguchi-Genetic Algorithm for Global Numerical Optimization". In: *Evolutionary Computation, IEEE Transactions on* 8.4 (2004), pp. 365–377. DOI: 826895.

[44] L. Lucci, R. Nesti, G. Pelosi, and S. Selleri. "Phase Centre Optimization in Profiled Corrugated Circular Horns with Parallel Genetic Algorithms". In: *PIER* 46 (2004), pp. 127–142.

[45] E. Agastra, G. Bellaveglia, L. Lucci, R. Nesti, G. Pelosi, G. Ruggerini, and S. Selleri. "Genetic Algorithm Optimization of High-Efficiency Wide-Band Multimodal Square Horns for Discrete Lenses". In: *PIER* 83 (2008), pp. 335–352.

[46] E. Agastra, L. Lucci, R. Nesti, G. Pelosi, and S. Selleri. "Modified NSGA-II Algorithm for Multiobjective Optimization of Compact High-Efficiency Square Horns". In: *International Journal of RF and Microwave Computer-Aided Engineering* 21.2 (2011), pp. 174–181.

[47] S. Selleri, M. Mussetta, P. Pirinoli, R. E. Zich, and L. Matekovits. "Some Insight Over New Variations of the Particle Swarm Optimization Method". In: *Antennas and Wireless Propagation, IEEE Letters* 5.1 (2006), pp. 235–238. DOI: 10.1109/LAWP.2006.874071.

[48] S. Selleri, M. Mussetta, P. Pirinoli, R. E. Zich, and L. Matekovits. "Differentiated Meta-PSO Methods for Array Optimization". In: *Antennas and Propagation, IEEE Transactions on* 56.1 (2008), pp. 67–75. DOI: 10.1109/TAP.2007.912942.

[49] M. Mussetta, P. Pirinoli, S. Selleri, and R.E. Zich. "Multi-Objective Swarm Intelligent Systems". In: *Multi-Objective Swarm Intelligent Systems, N. Nedjah, L.d.S. Coelho, L.d.M. Mourelle, [Eds.],* Springer-Verlag, Berlin (D), 2010. Chap. Meta-PSO for Multi-Objective EM Problems, pp. 125–150.

[50] A. S. Hedayat, N. J. A. Sloane, and J. Stufken. *Orthogonal Arrays: theory and Applications*. Springer-Verlag : New York (NY), 1999.

[51] N. J. A. Sloane. *A Library of Orthogonal Array*. URL: http://www.research.att.com/~njas/oadir/.

[52] A. Konak, D. W. Coit, and A. E. Smith. "Multi-Objective Optimization using Genetic Algorithms: a Tutorial". In: *Reliability Engineering & System Safety* 91 (2006), pp. 992–1007. DOI: 10.1016/j.ress.2005.11.018.

[53] D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. Tech. rep. Evolutionary Computation, 1999.

[54] M. Tanaka, H. Watanabe, Y. Furukawa, and T. Tanino. "GA-based Decision Support System for Multicriteria Optimization". In: *Proc. IEEE Int Syst., Man and Cybern. Intell. Syst. for the 21st Century. Conf.* Vol. 2. 1995, pp. 1556–1561. DOI: 10.1109/ICSMC.1995.537993.

[55] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & sons, Hoboken (NJ), 2001.

[56] K. Deb. "Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems". In: *Evolutionary Computation* 7 (1999), pp. 205–230.

[57] S. Huband, P. Hingston, L. Barone, and L. While. "A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit". In: *Evolutionary Computation, IEEE Transactions on* 10.5 (2006), pp. 477–506. DOI: 10.1109/TEVC.2005.861417.

[58] P. Ioannides and C. A. Balanis. "Uniform Circular Arrays for Smart Antennas". In: *Antennas and Propagation Magazine, IEEE* 47.4 (2005), pp. 192–206. DOI: 10.1109/MAP.2005.1589932.

[59] S. Pal, A. Basak, S. Das, A. Abraham, and I. Zelinka. "Concentric Circular Antenna Array Synthesis using a Differential Invasive Weed Optimization algorithm". In: *Proc. Int. Soft. Comput. and Pattern Recog. (SoCPaR)*. 2010, pp. 395–400. DOI: 10.1109/SOCPAR.2010.5686527.

[60] R. L. Haupt. "Optimized Element Spacing for Low Sidelobe Concentric Ring Arrays". In: *Antennas and Propagation, IEEE Transactions on* 56.1 (2008), pp. 266–268. DOI: 10.1109/TAP.2007.913176.

[61] N. N. Pathak, G. K. Mahanti, S. K. Singh, J. K. Mishra, and A. Chakraborty. "Sythesis of Thinned Planar Circular Array Antenna using Modified Particle Swarm Optimization". In: *PIER Letters* 12 (2009), pp. 87–97.

[62] G. G. Roy, S. Das, P. Chakraborty, and P. N. Suganthan. "Design of Non-Uniform Circular Antenna Arrays Using a Modified Invasive Weed Optimization Algorithm". In: *Antennas and Propagation, IEEE Transactions on* 59.1 (2011), pp. 110–118. DOI: 10.1109/TAP.2010.2090477.

[63] M. C. Viganó, G. Toso, G. Caille, C. Mangenot, and I. E. Lager. "Sunflower Array Antenna with Adjustable Density Taper". In: *International Journal of Antennas and Propagation* 2009 (2009).

[64] M. C. Viganó, G. Toso, S. Selleri, C. Mangenot, P. Angeletti, and G. Pelosi. "GA Optimized Thinned Hexagonal Arrays for Satellite Applications". In: *Proc. IEEE Antennas Propag. Soc. Int. Symp.* 2007, pp. 3165–3168. DOI: 10.1109/APS.2007.4396208.

[65] V. Zuniga, N. Haridas, A. T. Erdogan, and T. Arslan. "Effect of a Central Antenna Element on the Directivity, Half-Power Beamwidth and Side-Lobe Level of Circular Antenna Arrays". In: *Proc. NASA/ESA Conf. Adap. Hardw. and Syst. AHS 2009*. 2009, pp. 252–256. DOI: 10.1109/AHS.2009.63.

[66] R.L. Haupt and J.M. Johnson. "Phase-only Array Beam Control using a Genetic Algorithm". In: *Proc. First NASA/DoD Workshop on Evolvable Hardware*. 1999, pp. 217–224. DOI: 10.1109/EH.1999.785456.

[67] C. Luison, A. Landini, P. Angeletti, G. Toso, P. Valle, P. Capece, S. Selleri, and G. Pelosi. "Aperiodic Arrays for Spaceborne SAR Applications". In: *Antennas and Propagation, IEEE Transactions on* 60.5 (2012), pp. 2285–2294. DOI: 10.1109/TAP.2012.2189714.

[68] C. A. Balanis. *Antenna Theory: Analysis and Design, 2nd Edition*. Wiley, 1996.

[69] M. Mussetta, P. Pirinoli, S. Selleri, and R.E. Zich. "Differentiated Meta-PSO Techniques for Antenna Optimization". In: *Proc. of ICEAA*. 2007, pp. 2674–2679.

Part II

Method of Moments acceleration via ASM-MBF for Antenna Arrays

Chapter 4

Method of Moments

The Method of Moments (MoM) is probably the most widely used numerical technique in RF and antenna CEM, and has a long history in the field [1], [2] as the most popular Integral Equation method. Central to the MoM, as to other full-wave techniques, is the idea of discretizing some unknown electromagnetic property, typically surface currents or fields. This process of discretization is also known as *meshing*. It entails subdividing the geometry into a (large) number of small elements, that may be one-dimensional segments, two-dimensional surface patches (often triangles), three-dimensional tetrahedral elements, or a regular three-dimensional grid, depending on the problem at hand and the method used. Within each element, a simple functional dependence is assumed for the spatial variation of the unknown, but the amplitude (and possibly phase) of the unknown is determined by application of the method to the elements' mesh which approximates the original geometry. This functional dependence is also known as a basis (or expansion) function. Generally, the accuracy of the methods is related to the discretization (i.e. to the mesh size): the finer the mesh, the better is the accuracy of the methods. The largest mesh size (alternatively, the finest geometrical resolution) is limited by the available computational resources; for electromagnetic applications, the mesh fineness is usually determined by the requirement to sample the phase adequately.

Real numerical modelling techniques may be broadly classified into integral methods [3], [4], differential methods [5], and variational methods [6]. Variational methods are really based on the differential or integral form of the equation to be solved. Integral schemes for materially homogeneous problems have long been known and used: simple concepts, derived from an understanding of the physics of the fields, fall under the more general boundary elements schemes that are derived mathematically from the governing equations [7]. Integral equation methods have now been around for several decades, and their introduction to electromagnetics has been due to the seminal works of Richmond and Harrington in the 1960s, after that phase, integral methods assumed a sophisticated form under the boundary integral method [4], [8]. There was a surge in the interest in this topic in the 1980s, mainly due to the work of Wilton,

due to the increased power of computers. The interest in this area decreased when it was demonstrated that differential equation methods, with their sparse matrices, could solve many problems more efficiently than integral equations. However, in recent decades, due to the advent of fast algorithms, there was a revival in integral equation methods in electromagnetics. Traditional integral equation solvers were inefficient, but with the advent of fast solvers and smart acceleration techniques, they are more efficient than before, and have become even more efficient than differential equation solvers in many applications.

In the method of moments, the radiating/scattering structure is replaced by surface or volumetric equivalent currents [4], [9]–[11]. These surface currents are discretized into wire segments and/or surface patches. A matrix equation is then derived, representing the effect of every mesh element on every other one. This interaction is computed using the Green function for the problem. Most MoM codes use the free-space Green function; the relevant boundary condition is then applied to all the interactions, yielding a set of linear equations. The solution of this linear system yields the (approximate) current on each element. The resulting matrix which must be factored (or used in an iterative solution scheme) is dense, with complex valued entries. The merits of the method are few equations per solution with respect to a volumetric approach, since a reduction in dimensionality from a volume to a surface is performed, and ease in treating open boundary problems such as are posed by a radiating antenna propagating in space. Traditionally, the MoM has been applied in the frequency domain (FDIE), i.e. single frequency, or monochromatic, sinusoidal excitation, with an $e^{j\omega t}$ convention assumed. The working variables (unknowns) are thus complex valued, with a magnitude and phase, as for any phasor analysis. Time domain integral equation (TDIE) formulations have been used on occasions, but stability and other issues have proven difficult, and TDIE codes are rare. The use of the MoM for antenna analysis was given a major boost by the US government's de facto decision during the late 1980s to release the Numerical Electromagnetic Code Method of Moments (widely known as NEC-2) into the public domain. NEC-2 was a powerful, general-purpose antenna modelling program, but with no graphical abilities whatsoever and very limited meshing abilities. At present, there are some excellent commercial codes which offer all the functionality of NEC-2, but with proper graphical user tools and frequently greatly enhanced abilities, like FEKO [12].

The strong points of the MoM are the following [13]:

- Efficient treatment of perfectly or highly conducting surfaces. Only the surface is meshed; no *air box* around the antenna needs to be meshed.
- The MoM automatically incorporates the *radiation condition*, i.e. the correct behaviour of the field far from the source (proportional to $1/r$ in free space).
- The working variables are current densities, from which many important antenna parameters (impedance, gain, radiation patterns etc.) may be derived, some directly and some via straightforward numerical integration.

The weak points of the MoM may be summarized as follows [13]:

- The MoM does not handle electromagnetically penetrable materials as well as differential equation formulations. If the materials are homogeneous a fictitious, equivalent surface current formulation may be used, but inhomogeneous materials require fictitious equivalent volumetric currents, and become very expensive computationally.

- The MoM does not scale gracefully with frequency - for typical applications requiring a surface mesh, the scaling is $O((kd)^6)$ where kd is the electromagnetic size of the structure. Note that this implies an $O(f^6)$ scaling - doubling the frequency can result in a run-time 64 times as long. This is a major problem with all the computational methods, although the details do vary slightly from method to method. For a MoM volumetric mesh, required by an inhomogeneous structure, the scaling is $O((kd)^9)$; this is so large that such methods are usually very limited in application.

In conclusion, the MoM is the preferred method for frequency domain radiation and scattering problems involving perfectly or highly conducting surfaces, and is well suited for problems with homogeneous dielectric material. If the problem involves inhomogeneous dielectric materials, it is unlikely to be the best formulation.

In this context, a general purpose Method of Moments (MoM) has been developed, focused on the efficient treatment of finite arrays, composed of perfectly conducting materials or homogeneous dielectrics. The main purpose of the proposed code is to provide a fast and reliable tool to analyze large, yet finite arrays. Conventional MoM codes [10], [14] usually require a fine mesh to obtain a good accuracy: this can result in a large dense MoM impedance matrix, especially for finite array problems, where the mesh of a single element is replicated. Memory requirements and solution time for such a problem are $[O(N^2)]$ and $[O(N^3)]$ respectively for a direct solution. To address this issue, an approach based on the exploitation of Characteristics/Macro Basis Functions (CBF/MBF) [15]–[19] deriving from the Array Scanning Method (ASM) [20] is going to be exploited to shrink the size of the impedance matrix and accelerate the solution by several orders of magnitude. In this way, even large system of equations, usually solved by iterative techniques, could be reduced and efficiently solved by direct methods. Section 4.1.1 will present some useful method of moments definitions, including Rao Wilton Glisson basis functions,a fundamental building block exploited in the code's implementation for their useful properties; Section 4.2 will present the MoM problem derivation for perfectly conducting structures, while Section 4.3 will introduce the derivation for composite structures. In Section 4.4 Green Function calculation for periodic structures and their acceleration techniques will be presented.

4.1 MoM Definitions

In the following, the surface of a Perfect Electric Conductor (PEC) object $\sigma = \infty$ will be called a metallic surface, while the interface between two homogeneous penetrable domains with $\sigma < \infty$ will be called a dielectric interface. Note that the interfaces of homogeneous non-metallic domains are called dielectric surfaces although the domains may be magnetic, i.e. $\mu_r > 1$. Metallic surfaces are further classified as closed and open surfaces, where a surface is said to be closed if it's compact and without boundary, and open if it's not closed. Boundary conditions of the electromagnetic fields directly determine how the surface currents behave on the metallic and dielectric surfaces. Boundary conditions are:

1. $\hat{\mathbf{n}} \times \mathbf{E}$ and $\hat{\mathbf{n}} \cdot \mathbf{H}$ vanish on metallic surfaces,

2. $\hat{\mathbf{n}} \times \mathbf{E}$ and $\hat{\mathbf{n}} \times \mathbf{H}$ are continuous across dielectric surfaces.

The first boundary condition immediately yields the important result: the magnetic current $\mathbf{M} = \mathbf{E} \times \hat{\mathbf{n}}$ vanishes on metallic surfaces.

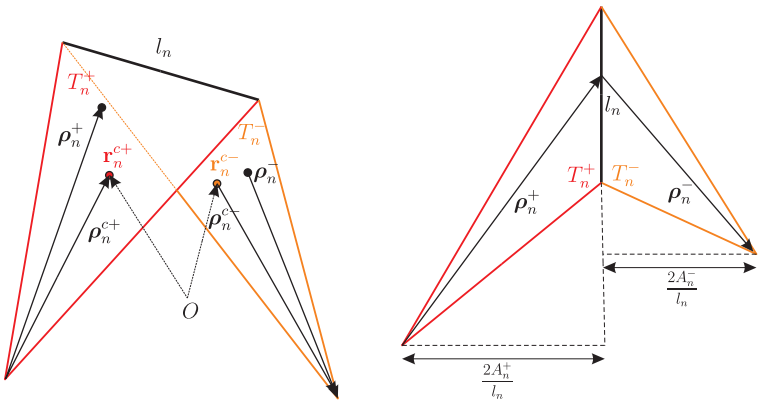
4.1.1 Rao Wilton Glisson basis functions

Rao Wilton Glisson (RWG) basis functions [21] are a set of basis functions suitable for use with EFIE/MFIE and triangular patch modeling. This formulation assumes that a suitable triangulation, defined in terms of an appropriate set of faces, edges and vertices has been found to approximate the surface S under investigation. Each basis function is associated with an interior (i.e non-boundary edge) edge of the patch model and is to vanish everywhere on S except in the two triangles attached to that edge. Fig. 4.1 shows two such triangles, T_n^+ and T_n^-, corresponding to the n-th edge of a triangulated surface model. Points in T_n^+ and T_n^- may be designated either by the position vector $\mathbf{r}$ defined with respect to the origin O, or by the position vectors $\boldsymbol{\rho}_n^+$ and $\boldsymbol{\rho}_n^-$, defined with respect to the free vertex of T_n^+ and T_n^- respectively. $\boldsymbol{\rho}_n^-$ is directed toward the free vertex of T_n^-. The plus or minus designation of the triangles is determined by the choice of a positive current reference direction for the n-th edge, the reference for which is assumed to be from T_n^+ to T_n^-. The RWG vector basis function associated with the n-th edge is:

$$\mathbf{f}_n(\mathbf{r}) = \begin{cases} \dfrac{l_n}{2A_n^+}\boldsymbol{\rho}_n^+ \text{ if } \mathbf{r} \text{ in } T_n^+ \\ \dfrac{l_n}{2A_n^-}\boldsymbol{\rho}_n^- \text{ if } \mathbf{r} \text{ in } T_n^+ \\ 0 \text{ otherwise} \end{cases} \tag{4.1}$$

where l_n is the length of the edge and $A_n^\pm$ is the area of triangle $T_n^\pm$.

Fundamental properties of RWGs are:



(a) Triangle pair and geometrical parameters

(b) Geometry for construction of component normal to edge.

Figure 4.1: Rao Wilton Glisson geometry and definition

1. The current has no component normal to the boundary (which excludes the common edge) of the surface formed by the triangle pair T_n^+ and T_n^-, and hence no line charges exist along this boundary.

2. The component of current normal to the n-th edge is constant and continuous across the edge. The normal component of $\boldsymbol{\rho}_n^\pm$ along edge n is just the height of the triangle $T_n^\pm$ with edge n as the base and the height expressed as $(2A_n^\pm)/l_n$. This latter factor normalizes $\mathbf{f}_n$ such that its flux density normal to edge n is 1, ensuring continuity of current normal to the edge. This result, together with point 1., implies that all edges of T_n^+ and T_n^- are free of line charges.

3. The surface divergence of $\mathbf{f}_n$, which is proportional to the surface charge density associated with the basis element, is

$$\boldsymbol{\nabla}_s \cdot \mathbf{f}_n(\mathbf{r}) = \begin{cases} \dfrac{l_n}{A_n^+} \text{ if } \mathbf{r} \text{ in } T_n^+ \\ \dfrac{l_n}{A_n^-} \text{ if } \mathbf{r} \text{ in } T_n^+ \\ 0 \text{ otherwise} \end{cases} \tag{4.2}$$

   The charge density is thus constant in each triangle, and the total charge associated with the triangle pair T_n^+ and T_n^- is zero.

4. The moment of $\mathbf{f}_n$ is given by $(A_n^+ + A_n^-)\mathbf{f}_n^{avg}$, where

$$\mathbf{f}_n^{avg} = \frac{1}{(A_n^+ + A_n^-)} \iint_{T_n^+ + T_n^-} \mathbf{f}_n \, \mathrm{dS} = \frac{l_n}{2}(\boldsymbol{\rho}_n^{c+} + \boldsymbol{\rho}_n^{c-}) = l_n(\mathbf{r}_n^{c+} - \mathbf{r}_n^{c-}) \tag{4.3}$$

and $\boldsymbol{\rho}_n^{c\pm}$ is the vector between the free vertex and the centroid of $T_n^{\pm}$ with $\boldsymbol{\rho}_n^{c-}$ directed toward and $\boldsymbol{\rho}_n^{c+}$ directed away from the vertex, as shown in Fig. 4.1b, and $\mathbf{r}_n^{c\pm}$ is the vector from O to the centroid of $T_n^{\pm}$.

4.1.2 Excitation

In the following, two excitation models are presented; the first is the plane wave excitation, that will be exploited in *scattering* problems; the second is the delta-gap excitation, that will be instead exploited for *transmission* problems.

Plane Wave

A plane wave is a constant-frequency wave whose wavefronts are infinite parallel planes of constant amplitudes, orthogonal to the direction of propagation of the wave (hence to the Poynting vector). The plane wave is a useful representation for many types of waves in the far field region. A plane wave in complex exponential form is

$$\mathbf{E}^i = E_0 e^{j\mathbf{k}\cdot\mathbf{r}} \tag{4.4}$$

where E_0 is the complex amplitude, $\mathbf{k}$ is the wave vector and $\mathbf{r}$ is the position vector. Consequently,

$$\mathbf{H}^i = \frac{1}{\zeta}\mathbf{k} \times \mathbf{E}^i \tag{4.5}$$

where ζ is the complex impedance of the medium.

Delta Gap Voltage

The Delta-gap generator model is a model that exploits a gap of negligible thickness h to provide excitation to an antenna. By the application of a voltage V (from positive to negative terminal) to the gap, the electric field $\mathbf{E}$ is

$$\mathbf{E} = -\nabla\phi = \frac{V}{h}\hat{\mathbf{n}} \tag{4.6}$$

where ϕ is the electric potential and $\hat{\mathbf{n}}$ is the normal to the edge. When the gap thickness h tends to zero, the electric field has an infinite values, hence

$$\mathbf{E} = V\delta(h)\hat{\mathbf{n}} \tag{4.7}$$

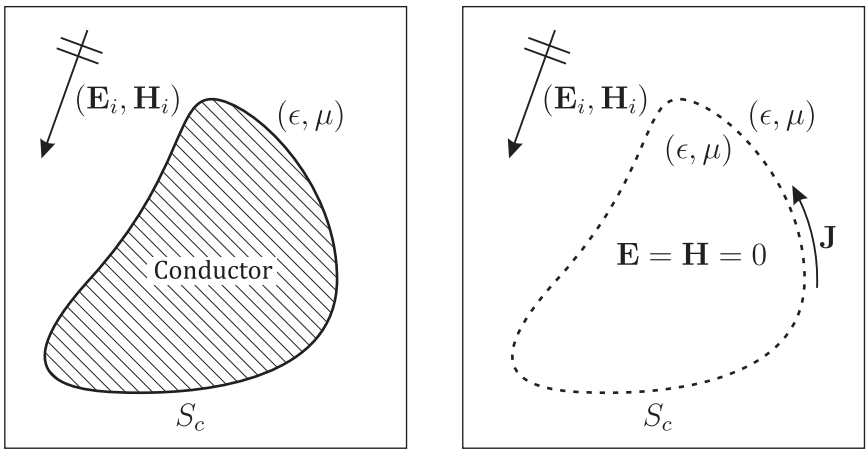
If the gap is associated with an internal edge of the structure, hence the driving electric field will be non-zero only on that RWG element. So, integrating on $T_m^+ + T_m^-$ leads to

$$V_{m=n} = \iint \mathbf{E}\cdot\mathbf{f}_n \ \mathrm{dS} = \iint \delta(h)\hat{\mathbf{n}}\cdot\mathbf{f}_n \ \mathrm{dS} = l_n V \tag{4.8}$$

for edge element $m = n$, and 0 otherwise. This exploiting the fact that an RWG has a component normal to the edge equal to 1, as pointed out above. The feeding voltage V will be defined by the application: for isolated elements it will have a conventional value of 1, while for arrays its values will depend on the desired amplitude taper.

4.2 MoM problem derivation for Conducting structures

Consider a perfectly conducting body, surrounded by an homogeneous media with electrical permittivity ϵ and permeability μ, as in Fig. 4.2a. $\mathbf{E}_i, \mathbf{H}_i$ are the incident electric and magnetic fields. Let S_c denote the surface enclosing the conducting body, with unit normal $\hat{\mathbf{n}}$. Fields outside the body can be



(a) Conductor (b) Equivalent surface

Figure 4.2: Equivalence principle for Conducting structures

represented with the help of equivalent sources [22], distributed over the surface S_c (Fig. 4.2b). The field generated by this equivalent (electric-only, as the surface is metallic) current $\mathbf{J} = \hat{\mathbf{n}} \times \mathbf{H}$ is such that the superposition with the incident field produces the original field outside, and zero field in the region inside S_c. Hence, the presence of any arbitrary material can be considered inside the surface, because of the absence of the field: the region inside S_c can be replaced by a region whose characteristics are the same as the medium surrounding the body. The Electric Field Integral Equation (EFIE) can be derived by enforcing the boundary condition on S_c [21]:

$$\hat{\mathbf{n}} \times [\mathbf{E}_i + \mathbf{E}_s(\mathbf{J})] = 0 \quad \text{on } S_c \tag{4.9}$$

The scattered electric field $\mathbf{E}_s$ can be computed from the surface current by

$$\mathbf{E}_s(\mathbf{J}) = -jw\mathbf{A} - \boldsymbol{\nabla}\phi_e \tag{4.10}$$

where $\mathbf{A}$, ϕ are respectively the electric vector and electric scalar potential:

$$\mathbf{A}(\mathbf{r}) = \frac{\mu}{4\pi} \iint_{S_c} \mathbf{J}(\mathbf{r}') \frac{e^{-jkR}}{R}\, dS \tag{4.11}$$

$$\phi_e(\mathbf{r}) = \frac{1}{4\pi\epsilon} \iint_{S_c} \rho_e(\mathbf{r}') \frac{e^{-jkR}}{R}\, dS \tag{4.12}$$

and the continuity equation is:

$$\nabla_{\mathbf{s}} \cdot \mathbf{J} = -jw\rho_e, \tag{4.13}$$

where ρ_e is the electric charge density. Finally, $R = |\mathbf{r} - \mathbf{r}'|$ is the distance between an arbitrarily located observation point $\mathbf{r}$ and a source $\mathbf{r}'$ point on S_c.

Inserting (4.10) into (4.9) leads to

$$-\mathbf{E}_{i,tan} = (-jw\mathbf{A} - \boldsymbol{\nabla}\phi_e)_{tan} \quad \text{on } S_c \tag{4.14}$$

The current on S_c can be expanded in terms of RWG basis function $\mathbf{f_n}$ as

$$\mathbf{J}(\mathbf{r}') = \sum_n I_n \mathbf{f}_n(\mathbf{r}') \tag{4.15}$$

Boundary conditions are imposed in weak form, i.e., the integral equation is weighted with a set of testing functions. The same expansion functions $\mathbf{f_m}$ are chosen as test basis function, hence exploiting a Galerkin testing procedure. It is interesting to notice that, besides leading to generally better posed systems of equations, this methodology also exactly ensures reciprocity in antenna problems. The testing procedure continues with the definition of a symmetric product as

$$< \mathbf{f}, \mathbf{g} >= \iint_S \mathbf{f} \cdot \mathbf{g}\ \mathrm{dS} \tag{4.16}$$

Testing (4.14) with $\mathbf{f_m}$ yields

$$< \mathbf{E}_i, \mathbf{f_m} >= jw < \mathbf{A}, \mathbf{f_m} > + < \nabla\phi_e, \mathbf{f_m} > \tag{4.17}$$

Subscripts m, n indicates source and test edges, l_m and l_n are the m-th and n-th element edge lengths, while $\boldsymbol{\rho}_m^{c\pm}$ are the RWG position vectors.

$$\boldsymbol{\rho}_m^{c+} = \mathbf{r}_m^{c+} - \mathbf{v}_m^{+}\ ,\ \boldsymbol{\rho}_m^{c-} = -\mathbf{r}_m^{c-} - \mathbf{v}_m^{-}. \tag{4.18}$$

After some manipulation [21], exploiting RWG properties described in Section 4.1.1 and the surface vector calculus identity $\boldsymbol{\nabla} \cdot \psi\mathbf{A} = \mathbf{A} \cdot \boldsymbol{\nabla}\psi + \psi\ \boldsymbol{\nabla} \cdot \mathbf{A}$ [22], terms in (4.17) can be expressed as

$$< \mathbf{E_i}, \mathbf{f_m} > \simeq\ \frac{l_m}{2}\left(\mathbf{E}_i(\mathbf{r}_m^{c+}) \cdot \boldsymbol{\rho}_m^{c+} + \mathbf{E}_i(\mathbf{r}_m^{c-}) \cdot \boldsymbol{\rho}_m^{c-}\right) \tag{4.19}$$

$$< \mathbf{A}, \mathbf{f_m} > \simeq\ \frac{l_m}{2}\left(\mathbf{A}(\mathbf{r}_m^{c+}) \cdot \boldsymbol{\rho}_m^{c+} + \mathbf{A}(\mathbf{r}_m^{c-}) \cdot \boldsymbol{\rho}_m^{c-}\right) \tag{4.20}$$

$$< \nabla\phi_e, \mathbf{f_m} > \simeq\ l_m\big(\phi_e(\mathbf{r}_m^{c+}) - \phi_e(\mathbf{r}_m^{c-})\big) \tag{4.21}$$

The inclusion of (4.19), (4.20) and (4.21) in (4.17) leads to the complete system of equations (4.22)

$$\begin{bmatrix} \quad Z_{mn} \quad \end{bmatrix} \begin{bmatrix} \ I_n \ \end{bmatrix} = \begin{bmatrix} \ V_m \ \end{bmatrix} \tag{4.22}$$

where the impedance term is

$$Z_{mn} = l_m \left[\frac{j\omega}{2} \left(\mathbf{A}^+_{mn} \cdot \boldsymbol{\rho}^{c+}_m + \mathbf{A}^-_{mn} \cdot \boldsymbol{\rho}^{c-}_m \right) + \phi^+_{e,mn} - \phi^-_{e,mn} \right], \tag{4.23}$$

and $\mathbf{A}_{mn}$, $\phi_{e,mn}$ are respectively the discretized electric vector and electric scalar potential:

$$\mathbf{A}^{\pm}_{mn} = \frac{\mu}{4\pi} \left[\frac{l_n}{2A^+_n} \int_{T^+_n} \boldsymbol{\rho}^{c+}_n g^{\pm}_m dS + \frac{l_n}{2A^-_n} \int_{T^-_n} \boldsymbol{\rho}^{c-}_n g^{\pm}_m dS \right], \tag{4.24}$$

$$\phi^{\pm}_{e,mn} = \frac{1}{j4\pi\omega\epsilon} \left[\frac{l_n}{A^+_n} \int_{T^+_n} g^{\pm}_m dS - \frac{l_n}{A^-_n} \int_{T^-_n} g^{\pm}_m dS \right], \tag{4.25}$$

considering $A^{\pm}_n$ as the area of the corresponding positive or negative RWG sub-triangle. I_n is the current solution on the n-th edge.

The free space Green's function is

$$g^{\pm}_m(\mathbf{r}^{c\pm}_m, \mathbf{r}') = \frac{e^{-jk|\mathbf{r}^{c\pm}_m - \mathbf{r}'|}}{|\mathbf{r}^{c\pm}_m - \mathbf{r}'|}, \tag{4.26}$$

where $\mathbf{r}^{c\pm}_m$ is the centroid of the m-th triangle. The $\pm$ sign in $\mathbf{A}^{\pm}_{mn}$ and $\phi^{\pm}_{e,mn}$ depends on the sign of the Green's function $g^{\pm}_m$.

Finally, each term of the RHS V_m is

$$\begin{aligned} V_m &= \frac{l_m}{2} \left(\mathbf{E}_i(\mathbf{r}^{c+}_m) \cdot \boldsymbol{\rho}^{c+}_m + \mathbf{E}_i(\mathbf{r}^{c-}_m) \cdot \boldsymbol{\rho}^{c-}_m \right) \text{ for scattering} \\ V_m &= \begin{cases} l_n V & \text{if feeding edge} \\ 0 & \text{otherwise} \end{cases} \text{ for edges feeding (4.8)} \end{aligned} \tag{4.27}$$

where V is the excitation assigned to edge m.

4.3 MoM problem derivation for Composite structures

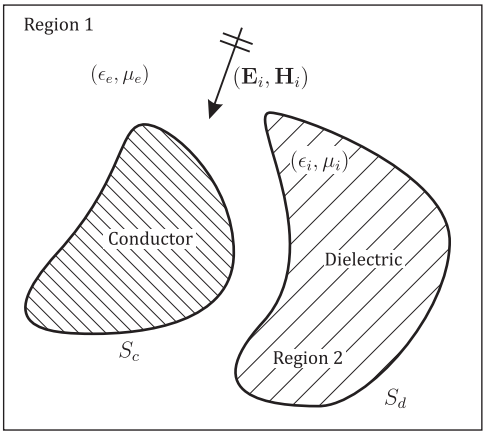
The surface integral equation (SIE) approach ([23], [24] and [25]–[28]) is very well suited to the analysis of homogeneous dielectric objects or to objects made up of homogeneous layers. A possible approach is to exploit the Green's function of the layered media; however, this solution assumes that the layered media is infinite in both directions. The procedure exploited here is to set up coupled integral equations in terms of equivalent electric and magnetic currents on the surfaces of the homogeneous regions. For an object made up of a large number of layers, fields induced in any region are expressed in terms of the equivalent currents on the adjacent interfaces. The following formulations of the MoM have been found to be generally suited for certain scattering problems based upon their geometry and material characteristics:

1. Conducting Scatterers (homogeneous, isotropic)
   (a) Electric field integral equation formulation (EFIE) for closed and open bodies
   (b) Magnetic field integral equation formulation (MFIE) for closed bodies
2. Dielectric Scatterers (homogeneous, isotropic)
   Combined field integral equation formulation (CFIE)
3. Anisotropic Scatterers (homogeneous)
   (a) Combined field integral equation formulation (CFIE) modified for material characteristics.

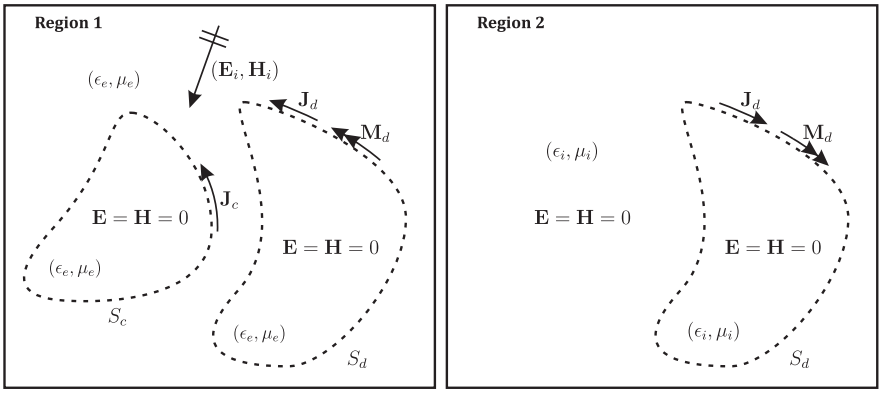
It is well known that the usage of the EFIE at internal resonant frequencies of the PEC can result in spurious solutions. This problem is not considered here.

With the help of the equivalence principle [22], starting from Fig. 4.3a, the following two problems are formulated, each valid for regions external and internal to the dielectric body, in terms of equivalent electric conductor current $\mathbf{J}_c$, and equivalent electric and magnetic currents $\mathbf{J}_d$ and $\mathbf{M}_d$, respectively. Equivalent currents are $\mathbf{J} = \hat{\mathbf{n}} \times \mathbf{H}$ and $\mathbf{M} = \mathbf{E} \times \hat{\mathbf{n}}$.

For the problem valid external to the dielectric region, Region 1, shown in Fig. 4.3b, the conductor and dielectric bodies are replaced by fictitious mathematical surfaces and the entire region is filled with the homogeneous material (ϵ_e, μ_e) of Region 1. The current $\mathbf{J}_c$ is allowed to flow on the mathematical surface S_c. In addition, two equivalent currents, $\mathbf{J}_d$ and $\mathbf{M}_d$, are introduced on the mathematical surface S_d. Finally, fields inside the surfaces S_c and S_d are set to zero. By enforcing the continuity of the tangential fields at S_c and S_d,the following equations corresponding to the Electric Field Integral Equation (EFIE) and the Magnetic Field Integral Equation (MFIE) are derived:



(a) Conductor+Dielectric



(b) Equivalent external problem (c) Equivalent internal problem

Figure 4.3: Equivalence principle for Composite structures

$$\begin{cases} \hat{\mathbf{n}} \times [\mathbf{E}_i + \mathbf{E}_s(\mathbf{J_c}) + \mathbf{E}_s(\mathbf{J_d}) + \mathbf{E}_s(\mathbf{M_d})] = 0 \quad \text{on } S_c \\ \hat{\mathbf{n}} \times [\mathbf{E}_i + \mathbf{E}_s(\mathbf{J_c}) + \mathbf{E}_s(\mathbf{J_d}) + \mathbf{E}_s(\mathbf{M_d})] = 0 \quad \text{on } S_d \end{cases} \qquad \text{EFIE} \qquad (4.28)$$

$$\begin{cases} \hat{\mathbf{n}} \times [\mathbf{H}_i + \mathbf{H}_s(\mathbf{J_c}) + \mathbf{H}_s(\mathbf{J_d}) + \mathbf{H}_s(\mathbf{M_d})] = 0 \quad \text{on } S_c \\ \hat{\mathbf{n}} \times [\mathbf{H}_i + \mathbf{H}_s(\mathbf{J_c}) + \mathbf{H}_s(\mathbf{J_d}) + \mathbf{H}_s(\mathbf{M_d})] = 0 \quad \text{on } S_d \end{cases} \qquad \text{MFIE} \qquad (4.29)$$

For the problem valid for the interior region to the dielectric body, as shown Fig. 4.3c, the entire space is filled with the material of the dielectric medium (ϵ_i, μ_i). On the mathematical surface S_d, the equivalent currents $\mathbf{J}_d$ and $\mathbf{M}_d$ are introduced. Fields are zero outside S_d. By enforcing the continuity of the tangential fields on S_d, the following integral equations are derived:

$$\hat{\mathbf{n}} \times [\mathbf{E}_s(-\mathbf{J_d}) + \mathbf{E}_s(-\mathbf{M_d})] = 0 \quad \text{on } S_d \quad \text{EFIE} \tag{4.30}$$
$$\hat{\mathbf{n}} \times [\mathbf{H}_s(-\mathbf{J_d}) + \mathbf{H}_s(-\mathbf{M_d})] = 0 \quad \text{on } S_d \quad \text{MFIE} \tag{4.31}$$

The scattered electric and magnetic field $\mathbf{E}_s$ and $\mathbf{H}_s$ due to the electric currents $\mathbf{J}$ and the magnetic currents $\mathbf{M}$ are given by

$$\mathbf{E}_s(\mathbf{J},\mathbf{M}) = -jw\mathbf{A}(\mathbf{J}) - \boldsymbol{\nabla}\phi_e(\mathbf{J}) - \frac{1}{\epsilon}\boldsymbol{\nabla} \times \mathbf{F}(\mathbf{M}) \tag{4.32}$$

$$\mathbf{H}_s(\mathbf{J},\mathbf{M}) = -jw\mathbf{F}(\mathbf{M}) - \boldsymbol{\nabla}\phi_m(\mathbf{M}) + \frac{1}{\mu}\boldsymbol{\nabla} \times \mathbf{A}(\mathbf{J}) \tag{4.33}$$

Testing is conducted via the symmetric product defined in (4.16); testing of the EFIEs (4.28) leads to

$$< \mathbf{E}_i, \mathbf{f_m} >= jw < \mathbf{A}, \mathbf{f_m} > + < \nabla\phi_e, \mathbf{f_m} > + \frac{1}{\epsilon} < \nabla \times \mathbf{F}, \mathbf{f_m} > \tag{4.34}$$

and for the MFIEs (4.29) to

$$< \mathbf{H}_i, \mathbf{f_m} >= jw < \mathbf{F}, \mathbf{f_m} > + < \nabla\phi_m, \mathbf{f_m} > - \frac{1}{\mu} < \nabla \times \mathbf{A}, \mathbf{f_m} > \tag{4.35}$$

while EFIE (4.30) is

$$0 = jw < \mathbf{A}, \mathbf{f_m} > + < \nabla\phi_e, \mathbf{f_m} > + \frac{1}{\epsilon} < \nabla \times \mathbf{F}, \mathbf{f_m} > \tag{4.36}$$

and MFIEs (4.31)

$$0 = jw < \mathbf{F}, \mathbf{f_m} > + < \nabla\phi_m, \mathbf{f_m} > - \frac{1}{\mu} < \nabla \times \mathbf{A}, \mathbf{f_m} > \tag{4.37}$$

where $\mathbf{A}$, $\mathbf{F}$, ϕ_e, ϕ_m are respectively the electric vector, magnetic vector, electric scalar and magnetic scalar potential:

$$\mathbf{A}(\mathbf{r}) = \frac{\mu}{4\pi} \iint_S \mathbf{J}(\mathbf{r}') \frac{e^{-jkR}}{R}\, dS \tag{4.38}$$

$$\mathbf{F}(\mathbf{r}) = \frac{\epsilon}{4\pi} \iint_S \mathbf{M}(\mathbf{r}') \frac{e^{-jkR}}{R}\, dS \tag{4.39}$$

$$\phi_e(\mathbf{r}) = \frac{1}{4\pi\epsilon} \iint_S \rho_e(\mathbf{r}') \frac{e^{-jkR}}{R}\, dS \tag{4.40}$$

$$\phi_m(\mathbf{r}) = \frac{1}{4\pi\mu} \iint_S \rho_m(\mathbf{r}') \frac{e^{-jkR}}{R}\, dS \tag{4.41}$$

and the continuity equations hold:

$$\nabla_{\mathbf{s}} \cdot \mathbf{J} = -jw\rho_e, \tag{4.42}$$

$$\nabla_{\mathbf{s}} \cdot \mathbf{M} = -jw\rho_m, \tag{4.43}$$

where ρ_e and ρ_m are the electric and magnetic charge density.

Electric and magnetic currents on S_c and S_d can be expanded in terms of basis function $\mathbf{f_n}$ as

$$\mathbf{J}(\mathbf{r}') = \sum_{n=1}^{N_c+N_d} I_n \mathbf{f}_n(\mathbf{r}') \tag{4.44}$$

$$\mathbf{M}(\mathbf{r}') = \sum_{n=1}^{N_d} M_n \mathbf{f}_n(\mathbf{r}') \tag{4.45}$$

Exploiting RWG as source and test basis functions, quantities in (4.34), (4.35), (4.36) and (4.37) can be expressed in a convenient form: $< \mathbf{E_i}, \mathbf{f_m} >$, $< \mathbf{A}, \mathbf{f_m} >$ and $< \phi_e, \mathbf{f_m} >$ take advantage of (4.19), (4.20) and (4.21). More,

$$< \mathbf{H_i}, \mathbf{f_m} > \simeq \frac{l_m}{2}\left(\mathbf{H}_i(\mathbf{r}_m^{c+}) \cdot \boldsymbol{\rho}_m^{c+} + \mathbf{H}_i(\mathbf{r}_m^{c-}) \cdot \boldsymbol{\rho}_m^{c-}\right) \tag{4.46}$$

$$< \mathbf{F}, \mathbf{f_m} > \simeq \frac{l_m}{2}\left(\mathbf{F}(\mathbf{r}_m^{c+}) \cdot \boldsymbol{\rho}_m^{c+} + \mathbf{F}(\mathbf{r}_m^{c-}) \cdot \boldsymbol{\rho}_m^{c-}\right) \tag{4.47}$$

$$< \nabla\phi_m, \mathbf{f_m} > \simeq l_m\big(\phi_m(\mathbf{r}_m^{c+}) - \phi_m(\mathbf{r}_m^{c-})\big) \tag{4.48}$$

and

$$< \nabla \times \mathbf{F}, \mathbf{f_m} >= \frac{l_m}{2}\Big((\boldsymbol{\nabla} \times \mathbf{F}(\mathbf{r}_m^{c+}) \cdot \boldsymbol{\rho}_m^{c+} + (\boldsymbol{\nabla} \times \mathbf{F}(\mathbf{r}_m^{c-}) \cdot \boldsymbol{\rho}_m^{c-}\Big) \tag{4.49}$$

$$< \nabla \times \mathbf{A}, \mathbf{f_m} >= \frac{l_m}{2}\Big((\boldsymbol{\nabla} \times \mathbf{A}(\mathbf{r}_m^{c+}) \cdot \boldsymbol{\rho}_m^{c+} + (\boldsymbol{\nabla} \times \mathbf{A}(\mathbf{r}_m^{c-}) \cdot \boldsymbol{\rho}_m^{c-}\Big) \tag{4.50}$$

The inclusion of (4.19), (4.20), (4.21), (4.46), (4.47), (4.48), (4.49), (4.50) in (4.34), (4.35), (4.36) and (4.37) leads to the complete system of equations :

$$\left[\begin{array}{c|c} Z_{mn} & C_{mn} \\ \hline D_{mn} & Y_{mn} \end{array}\right] \left[\begin{array}{c} I_{mn} \\ \hline M_{mn} \end{array}\right] = \left[\begin{array}{c} V_{mn} \\ \hline H_{mn} \end{array}\right] \tag{4.51}$$

where the impedance block, analogous to that of (4.23), is

$$Z_{mn} = l_m \left[\frac{j\omega}{2}\left(\mathbf{A}_{mn}^{+} \cdot \boldsymbol{\rho}_m^{c+} + \mathbf{A}_{mn}^{-} \cdot \boldsymbol{\rho}_m^{c-}\right) + \phi_{e,mn}^{+} - \phi_{e,mn}^{-}\right], \tag{4.52}$$

while the first off-diagonal block is

$$C_{mn} = l_m \left[\frac{1}{2\epsilon} \Big((\boldsymbol{\nabla} \times \mathbf{F}^+_{mn}) \cdot \boldsymbol{\rho}^{c+}_m + (\boldsymbol{\nabla} \times \mathbf{F}^-_{mn}) \cdot \boldsymbol{\rho}^{c-}_m \Big) \right], \tag{4.53}$$

the admittance block is

$$Y_{mn} = l_m \left[\frac{j\omega}{2} \left(\mathbf{F}^+_{mn} \cdot \boldsymbol{\rho}^{c+}_m + \mathbf{F}^-_{mn} \cdot \boldsymbol{\rho}^{c-}_m \right) + \phi^+_{m,mn} - \phi^-_{m,mn} \right] \tag{4.54}$$

and the last off-diagonal block is

$$D_{mn} = l_m \left[-\frac{1}{2\mu} \Big((\boldsymbol{\nabla} \times \mathbf{A}^+_{mn}) \cdot \boldsymbol{\rho}^{c+}_m + (\boldsymbol{\nabla} \times \mathbf{A}^-_{mn}) \cdot \boldsymbol{\rho}^{c-}_m \Big) \right]; \tag{4.55}$$

$\mathbf{A}_{mn}$, $\mathbf{F}_{mn}$, $\phi_{e,mn}$, $\phi_{m,mn}$ are respectively the discretized electric vector, magnetic vector, electric scalar and magnetic scalar potential:

$$\mathbf{A}^{\pm}_{mn} = \frac{\mu}{4\pi} \left[\frac{l_n}{2A^+_n} \int_{T^+_n} \boldsymbol{\rho}^{c+}_n g^{\pm}_m dS + \frac{l_n}{2A^-_n} \int_{T^-_n} \boldsymbol{\rho}^{c-}_n g^{\pm}_m dS \right], \tag{4.56}$$

$$\mathbf{F}^{\pm}_{mn} = \frac{\epsilon}{4\pi} \left[\frac{l_n}{2A^+_n} \int_{T^+_n} \boldsymbol{\rho}^{c+}_n g^{\pm}_m dS + \frac{l_n}{2A^-_n} \int_{T^-_n} \boldsymbol{\rho}^{c-}_n g^{\pm}_m dS \right], \tag{4.57}$$

$$\phi^{\pm}_{e,mn} = \frac{1}{j4\pi\omega\epsilon} \left[\frac{l_n}{A^+_n} \int_{T^+_n} g^{\pm}_m dS - \frac{l_n}{A^-_n} \int_{T^-_n} g^{\pm}_m dS \right], \tag{4.58}$$

$$\phi^{\pm}_{m,mn} = \frac{1}{j4\pi\omega\mu} \left[\frac{l_n}{A^+_n} \int_{T^+_n} g^{\pm}_m dS - \frac{l_n}{A^-_n} \int_{T^-_n} g^{\pm}_m dS \right], \tag{4.59}$$

and with some manipulation and the exploitation of the vector calculus identity $\boldsymbol{\nabla} \times \psi\mathbf{A} = \psi \ \boldsymbol{\nabla} \times \mathbf{A} + \boldsymbol{\nabla}\psi \times \mathbf{A}$ [22]:

$$\nabla \times \mathbf{F}^{\pm}_{mn} = \frac{\epsilon}{4\pi} \left[\frac{l_n}{2A^+_n} \int_{T^+_n} \boldsymbol{\rho}^{c+}_n \times \boldsymbol{\nabla}' g^{\pm}_m dS + \frac{l_n}{2A^-_n} \int_{T^-_n} \boldsymbol{\rho}^{c-}_n \times \boldsymbol{\nabla}' g^{\pm}_m dS \right], \tag{4.60}$$

$$\nabla \times \mathbf{A}^{\pm}_{mn} = \frac{\mu}{4\pi} \left[\frac{l_n}{2A^+_n} \int_{T^+_n} \boldsymbol{\rho}^{c+}_n \times \boldsymbol{\nabla}' g^{\pm}_m dS + \frac{l_n}{2A^-_n} \int_{T^-_n} \boldsymbol{\rho}^{c-}_n \times \boldsymbol{\nabla}' g^{\pm}_m dS \right] \tag{4.61}$$

Also in this case, the free space Green's function is

$$g^{\pm}_m(\mathbf{r}^{c\pm}_m, \mathbf{r}') = \frac{e^{-jk|\mathbf{r}^{c\pm}_m - \mathbf{r}'|}}{|\mathbf{r}^{c\pm}_m - \mathbf{r}'|}, \tag{4.62}$$

and the free space Green's function gradient in prime coordinates $\boldsymbol{\nabla}' g = -\boldsymbol{\nabla} g$ is

$$\boldsymbol{\nabla}' g^{\pm}_m(\mathbf{r}^{c\pm}_m, \mathbf{r}') = (\mathbf{r}^{c\pm}_m - \mathbf{r}')(1 + jk|\mathbf{r}^{c\pm}_m - \mathbf{r}'|) \frac{e^{-jk|\mathbf{r}^{c\pm}_m - \mathbf{r}'|}}{|\mathbf{r}^{c\pm}_m - \mathbf{r}'|^3}, \tag{4.63}$$

where $\mathbf{r}_m^{c\pm}$ is the centroid of the m-th triangle.

Each term of the RHS is

$$V_m = \begin{cases} \dfrac{l_m}{2}\left(\mathbf{E}_i(\mathbf{r}_m^{c+})\cdot\boldsymbol{\rho}_m^{c+} + \mathbf{E}_i(\mathbf{r}_m^{c-})\cdot\boldsymbol{\rho}_m^{c-}\right) & \text{Region 1} \\ 0 & \text{Region 2} \end{cases} \quad \text{for scattering}$$

$$V_m = \begin{cases} l_n V & \text{if feeding edge} \\ 0 & \text{otherwise} \end{cases} \quad \text{for edges feeding (4.8)} \tag{4.64}$$

and

$$H_m = \begin{cases} \dfrac{l_m}{2}\left(\mathbf{H}_i(\mathbf{r}_m^{c+})\cdot\boldsymbol{\rho}_m^{c+} + \mathbf{H}_i(\mathbf{r}_m^{c-})\cdot\boldsymbol{\rho}_m^{c-}\right) & \text{Region 1} \\ 0 & \text{Region 2} \end{cases} \quad \text{for scattering} \tag{4.65}$$

where V is the excitation assigned to edge m.

4.3.1 Dielectric and Metallic Junctions

The MoM derivation explained in the previous Section is relative to a problem where the conductor and dielectric are separate one from each other. The treatment of junctions among different dielectrics, metallic and mixed bodies is of paramount importance to solve real-world design problems, where such situations are common [11], [29]–[31].

Let e_n be an edge on the triangular mesh. It is called a junction, if more than two domains or surfaces meet at e_n. Otherwise, e_n is a single edge. Edges and junctions e_n are classified as:

1. Dielectric edges or junctions, if e_n lies at an intersection of two or more dielectric surfaces but does not meet any metallic surfaces.

2. Metallic edges or junctions, if e_n lies on the intersection of open or closed metallic surfaces but does not meet any dielectric surfaces.

3. Composite metal-dielectric junctions, if e_n lies at an intersection of at least one open or closed metallic surface and at least one dielectric surface.

Dielectric Edges or Junctions

Considering the surface currents and their basis functions expansions on a dielectric interface $e_n = S_{1,2}$ between two dielectric domains D_1 and D_2. Surface currents are related to the fields by $\mathbf{J} = \hat{\mathbf{n}} \times \mathbf{H}$ and $\mathbf{M} = \mathbf{E} \times \hat{\mathbf{n}}$; boundary conditions imply that the tangential components of the fields are continuous across a dielectric interface, and since the normal vectors $\hat{\mathbf{n}}$ point into opposite directions

$$\mathbf{J}_1 = -\mathbf{J}_2 \text{ and } \mathbf{M}_1 = -\mathbf{M}_2. \tag{4.66}$$

This implies that the components of the surface currents $\mathbf{J}_i$ and $\mathbf{M}_j$,normal to the edge,are continuous across the edge at the interface. Because the two basis functions assigned to an edge on $S_{1,2}$ and on the opposite sides of it flow into opposite directions, in the surface current representations they must have equal coefficients. This is illustrated in Fig. 4.4. Hence, the unknown coefficients of the oriented basis functions assigned to the same dielectric edge or junction must have the same value and must be combined into a single unknown. Thus, at an edge or a junction of a number of dielectric surfaces, there are only two independent unknowns, one for $\mathbf{J}$ and another for $\mathbf{M}$ [29].

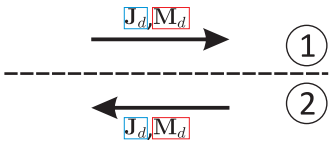


Figure 4.4: Unknowns associated to the electric and magnetic basis functions assigned at an edge on the interface of two dielectric domains. Dashed lines denote dielectric surfaces, numbers indicate domains. Colored boxes indicate basis functions with the same unknown coefficient.

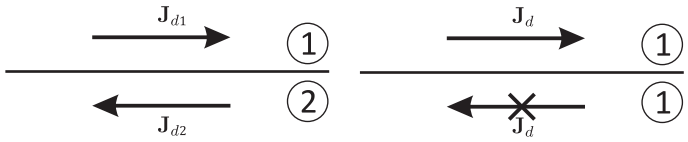
Metallic Edges or Junctions

Here, surface currents and their expansions at a junction of a number of metallic surfaces are considered. The surfaces may be open or closed. Since boundary conditions, as pointed out in Section 4.1, imply that $\mathbf{M}$ vanishes on metallic surfaces, the magnetic basis functions associated to e_n will be removed. In addition,since the magnetic field is not necessarily continuous across metallic surfaces, $\mathbf{J}$ has independent values on the opposite sides of the metallic surfaces, hence the unknown coefficients associated to the electric basis functions assigned on a metallic junction or edge can not be combined. This is illustrated in Fig. 4.5a. Naturally the electric basis functions inside closed metallic objects are removed and such basis functions should not ever be created.

However, if all metallic surfaces associated to e_n are open and e_n is completely in the interior of one homogeneous dielectric domain, the following applies: as already is mentioned, $\mathbf{J}$ must have independent values on the opposite sides of S, denoted by $\mathbf{J}_1$ and $\mathbf{J}_2$, and let $\mathbf{f}_1$ and $\mathbf{f}_2$ be the basis functions assigned to e_n with unknown coefficients α_1 and α_2. These two basis functions produce the same fields with opposite signs due to the orientation of the basis functions. Therefore,in order to avoid linear dependence of the columns of the system matrix, the fields of the basis functions must be presented by only one of them, and $\alpha_1 - \alpha_2$ can be considered as a new single unknown. This is illustrated in Fig. 4.5b.

Hence, at a metallic edge or junction meeting no dielectric surfaces,all the magnetic basis functions in the expansion of M, with their unknowns, must be removed. In addition,if all surfaces assigned to en are open metallic surfaces

and en lies completely in the interior of a homogeneous dielectric domain,one of the electric basis functions in the expansion of J must be removed [29].



(a) Metal interface between different domains

(b) Metal interface in a homogeneous domain

Figure 4.5: Unknowns associated to the electric basis functions assigned at metallic junctions. Solid lines denote open metallic surfaces, numbers indicate domains. The unknowns associated to the electric basis functions denoted with crosses are removed.

Metal-Dielectric Junction

Finally, let e_n a general metal-dielectric junction. Since $\mathbf{M}$ vanishes on metallic surfaces, then the component of $\mathbf{M}$, normal to the edge, also vanishes,and so the magnetic basis functions assigned at e_n and the unknowns associated with them. Again, due to the fact that the magnetic field is not continuous across open metallic surfaces, $\mathbf{J}$ must have two independent values on the opposite sides of an open metallic surface. This implies that in the expansion of $\mathbf{J}$ the unknown coefficients of the basis functions assigned to e_n on the opposite sides of metallic surfaces should be considered as independent unknowns, too. This is illustrated in Fig. 4.6.

Hence, the unknown coefficients pertaining to the expansions of $\mathbf{J}$ assigned to a general metal-dielectric junction between two metallic surfaces must have the same value and the corresponding unknowns must be combined to a single unknown. In the expansions of $\mathbf{M}$ the magnetic basis functions, with their unknowns, assigned to a general metal-dielectric junction must be removed [29].

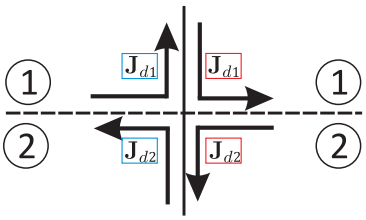


Figure 4.6: Unknowns associated to the electric basis functions assigned at an edge at a composite metal-dielectric interface. Solid lines denote open metallic surfaces, dashed lines denote dielectric surfaces, numbers indicate domains. Colored boxes indicate basis functions with the same unknown coefficient.

PMCHWT at Dielectric Edges or Junctions

Poggio-Miller-Chang-Harrington-Wu-Tsai (PMCHWT) formulation [29], [30], [32] is applied on dielectric interfaces For a single edge on an interface of two dielectric domains the PMCHWT formulation is a summation of the EFIEs and MFIEs, respectively, defined on the opposite sides of the interface. If e_n is a dielectric edge or a junction, the adjacent EFIEs and MFIEs are summed as

$$\sum_{m=1}^{D} EFIE_n^m \quad \sum_{m=1}^{D} MFIE_n^m \tag{4.67}$$

where D is the number of domains meeting at e_n, and subscript m indicates the discretized EFIEs and MFIEs in those domains tested with the oriented RWG functions assigned to e_n.

4.4 Green Function Calculation for periodic structures

Infinite array simulations are used here as a starting point to improve the solution of finite array simulations; hence they must be conducted in a fast and reliable way. Infinite array analysis are reduced to single cell analysis, via the enforcement of periodic Floquet boundary conditions: in this way the geometrical periodicity is embedded in the Green's Function (now GF). This provides a great numerical advantage as compared to direct simulations of very large (and possibly truncated) structures. MoM techniques described in the previous Sections hence can be applied, provided that a periodic GF is exploited in place of the free-space one. Assuming that the periodic structure is made of scatterers that are not electrically connected with those in contiguous cells, for perfectly conducting scatterers, the use of the periodic GF is sufficient to obtain correct results. When the scattering body is composite, then the periodic GF must be considered only for the exterior problem, while the interior problem one remains the homogeneous free-space one.

Considering a periodic array along skew axes $y = 0$ (index n) and $y = x \tan\phi$ (index m), with periods d_x and d_y and phase shifts $k_{x,0}$ and $k_{y,0}$ in the x and y direction, the following parameters are defined, with reference to Fig. 4.7, to accurately analyze the problem [33]:

- $A = d_x d_y \cos\phi$ is the area of the unit cell;
- $\boldsymbol{\beta}_0 = k_{x,0}\hat{\mathbf{x}} + k_{y,0}\hat{\mathbf{y}}$ is the phase shifts vector, containing phase shift between consecutive elements;
- $\boldsymbol{\rho}_{nm} = (nd_x + md_y/\tan\phi)\hat{\mathbf{x}} + md_y\hat{\mathbf{y}}$ is the array lattice vector, describing array element positions;
- $R_{nm} = |\mathbf{r} - \boldsymbol{\rho}_{nm}|$ is the array position distance.

Propagation vectors are:

- $\mathbf{k}_{t,nm} = \left(k_{x,0} + \frac{2\pi n}{d_x}\right)\hat{\mathbf{x}} + \left(k_{y,0} + \frac{2\pi m}{d_y} - \frac{2\pi n}{d_x \tan\phi}\right)\hat{\mathbf{y}}$ transverse;
- $k_{z,nm} = \sqrt{k^2 - \mathbf{k}_{t,nm} \cdot \mathbf{k}_{t,nm}}$ longitudinal.

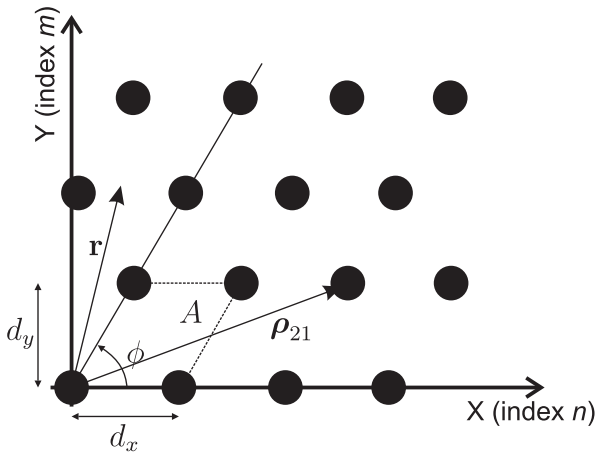


Figure 4.7: Array parameters for a skewed grid array geometry. The array lattice vector $\boldsymbol{\rho}_{21}$ is depicted.

4.4.1 Free space periodic GF

The spatial representation of the periodic GF is

$$g_\infty(\mathbf{r}) = \frac{1}{4\pi} \sum_{n=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} \frac{e^{-jkR_{nm}}}{R_{nm}} e^{-j\boldsymbol{\beta}_0 \cdot \boldsymbol{\rho}_{nm}} \tag{4.68}$$

This expression has a poor $[O(1/n)]$ convergence, usually requiring thousands of terms to be summed up to attain an acceptable accuracy. Moreover, the spatial series (4.68) does not converge for a complex wavenumbers vector $\boldsymbol{\beta}_0$. Hence some acceleration techniques must be exploited to ensure a proper convergence [34]–[37]. Two different periodic GF accelerations have been analyzed.

4.4.2 Acceleration via Poisson formula

A first attempt to improve convergence of the periodic GF is to transform it into the spectral domain utilizing an infinite discrete spectrum of plane waves, via Fourier transformation [34], [38]; the result is the *Poisson Summation Formula*

$$g_{\infty,P}(\mathbf{r}) = \frac{1}{2jA} \sum_{n=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} \frac{e^{-jk_{z,nm}|z|}}{k_{z,nm}} e^{-j\mathbf{k}_{t,nm} \cdot \mathbf{r}} \tag{4.69}$$

The branch of $k_{z,nm}$ is chosen so that $Im\{k_{z,nm}\} < 0$. The drawback of series (4.69) is that it converges slowly for observation points close to the array plane, i.e. $|z| \to 0$. In this way a better convergence with respect to (4.68) is achieved, but a significant number of terms must be calculated anyway.

4.4.3 Acceleration via Ewald formula

A more relevant acceleration in convergence can be achieved via the exploitation of *Ewald's Method* [34], [39], [40]. This technique has been recognized as one of the most efficient techniques for the computation of periodic GFs. This technique transforms the periodic GF into a sum of a spatial and a spectral series:

$$g_{\infty,E}(\mathbf{r}) = G_{spectral}(\mathbf{r}) + G_{spatial}(\mathbf{r}) \tag{4.70}$$

where

$$\begin{aligned} G_{spectral}(\mathbf{r}) = \sum_{n=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} \frac{1}{j4A} \frac{e^{-j\mathbf{k}_{t,nm}\cdot\mathbf{r}}}{k_{z,nm}} \cdot \\ \left(e^{-j|z|k_{z,nm}} \mathrm{erfc}\left(\frac{jk_{z,mn}}{2E_{opt}} - |z|E_{opt} \right) + \right. \\ \left. e^{j|z|k_{z,nm}} \mathrm{erfc}\left(\frac{jk_{z,mn}}{2E_{opt}} + |z|E_{opt} \right) \right) \end{aligned} \tag{4.71}$$

and

$$\begin{aligned} G_{spatial}(\mathbf{r}) = \sum_{n=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} \frac{1}{8\pi} \frac{e^{-j\boldsymbol{\beta}_0\cdot\boldsymbol{\rho}_{nm}}}{R_{nm}} \cdot \\ \left(e^{-jkR_{nm}} \mathrm{erfc}\left(R_{nm}E_{opt} - \frac{jk}{2E_{opt}} \right) + \right. \\ \left. e^{jkR_{nm}} \mathrm{erfc}\left(R_{nm}E_{opt} + \frac{jk}{2E_{opt}} \right) \right) \end{aligned} \tag{4.72}$$

erfc is the complementary error function, and $E_{opt} = \sqrt{\pi/A}$ is the optimal split parameter. The use of this optimal splitting parameter ensures that the same number of terms in the spatial and spectral series is used to achieve a certain rate of convergence [41]. With the exploitation of this acceleration technique a dramatical improvement in convergence of the periodic GF is achieved: the two series exhibit Gaussian convergence, and only 3 terms ($m, n = -1, 0, 1$) must be calculated to achieve an accuracy equal to that of (4.68) calculated on thousands of terms. It is important to point out that the Ewald representation can be used for complex wavenumbers (complex $\boldsymbol{\beta}_0$). Ewald's method has then been chosen as the most suitable technique for an efficient periodic GF evaluation. In the implementation, the Faddeeva function is exploited instead of the error function erfc in (4.71) and (4.72); this has proven to lead to an high speed-up in the calculation of the GF [42].

Chapter 5

Method of Moments code

This Section presents the proposed Method of Moments implementation. The code structure has been described in [P4], and is depicted in Fig. 5.1. All the code is written in a standard C++ implementation [43], exploiting open-source libraries. This framework is capable of analyzing isolated structures, infinite arrays via periodic boundary conditions and finite arrays, on rectangular and skewed grids (Fig. 5.2).

Section 5.1 will present the details of the code's implementation from a numerical point of view, describing libraries exploited and providing a brief background on dense linear system solving. Sections 5.2, 5.3, 5.4, 5.5 and 5.6 will describe the code main capabilities;.

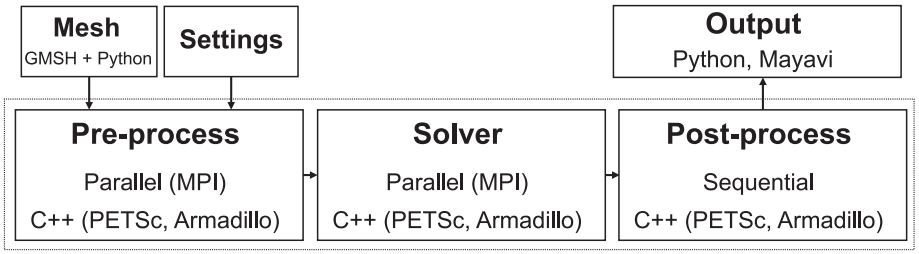


Figure 5.1: MoM code flow chart.

5.1 Numerical Treatment

5.1.1 Open Source Libraries

The code is entirely implemented using some open-source libraries:

- PETSc 3.3.6 (C++) [44], [45]
  Portable Extensible Toolkit for Scientific Computation, a suite of data

structures and routines for the parallel solution of scientific applications modeled by integral / differential equations;

- Armadillo 4.450.0 (C++) [46]
  Provides MATLAB-style syntax to C++, used to manipulate vectors and matrices;
- Lapack 3.5.0 (Fortran) [47], [48]
  Linear Algebra Package, based on BLAS [49];
- OpenBLAS 0.2.11 (Fortran) [50]
  High Speed Replacement for reference BLAS;
- MPIch2 1.4.1 (C++) [51]
  Message Passing Interface, for parallel execution;
- Matplotlib 1.4.2 (Python) [52] 2D Data visualizer;
- Mayavi2 1.5 (Python) [53]
  3D Data visualizer, based on VTK.

Libraries are installed and linked under Ubuntu 12.10. Code documentation is generated via Doxygen 1.8.8

MoM matrix, Voltage and Current vectors are implemented as PETSc containers; in this way parallel calculations can be performed exploiting MPI without any increase in code complexity. Auxiliary structures are instead implemented via Armadillo containers, to improve code readability.

5.1.2 Dense Linear System Solving

The problem in solving linear systems is: given a matrix A and a vector b, find a vector x such that $Ax = b$. The simplest method of solving a matrix system is to use Gaussian elimination or LUD (lower-upper triangular decomposition) [54]. These solvers are also known as direct inversion solvers. One reason for their convenience is that there are many known routines for such methods. However, their complexity is bad: if the matrix system has N unknowns, the CPU time grows as N^3. One advantage of such a method is that when the LU factorization is performed, the CPU cost of solving for a new right-hand side is proportional to N^2. However, the matrix system has to be available, and hence, all the matrix elements have to be stored in the computer memory, requiring storage proportional to N^2.

Another way of solving a matrix system is to use an iterative method [55]. This method allows one to solve a matrix system by performing a small number of matrix-vector multiplies at each iteration. As one can produce the matrix-vector product without generating or storing the matrix, the method can be made matrix free, greatly reducing the storage requirements. In this case, only the unknowns need to be stored, and the memory requirement is proportional to N instead of N^2. For sparse matrices, and fast algorithms for dense matrices,

such matrix-vector product can be effected in $O(N)$ or $O(NlogN)$ operations, instead of $O(N^2)$ operations. So if the number of iterations can be kept small in these methods, one can greatly improve the solution time of the equation when N is large.

Distributed Dense System Solving

On a single processor, the algorithm for dense linear system solving is fairly obvious, although a good deal of optimization is needed for high performance In a distributed context, achieving high performance -especially performance that scales up with increasing processor numbers - requires radical rethinking about the basic data structures. ScaLAPACK and PLAPACK , are the two most popular packages for solving a linear system with a *distributed* dense coefficient matrix.

ScaLAPACK [56]
ScaLAPACK is a parallel version of LAPACK, both in function and in software design. Like the earlier package, ScaLAPACK targets linear system solution and eigenvalue calculation for dense and banded matrices. Note that, while sparse matrices are often of banded form, use of the band storage is usually not an efficient way of dealing with dense systems. In a way, ScaLAPACK is the culmination of a line of linear algebra packages that started with LINPACK and EISPACK. The coding of those packages was fairly straightforward, using at most Basic Linear Algebra Subprograms (BLAS) Level-1 operations as an abstraction level. LAPACK attains high efficiency on a single processor (or a small number of shared-memory processors) through the introduction of blocked algorithms and the concomitant use of BLAS Level-3 operations. ScaLAPACK uses these blocked algorithms in a parallel context to attain scalably high performance on parallel computers. In ScaLAPACK the relevant parts of the code are confined to two subroutine libraries: the BLAS for the computational kernels and the BLACS (Basic Linear Algebra Communication Subprograms) [57] for communication kernels, which offers an abstraction layer over MPI. ScaLAPACK are exploited in FEKO.

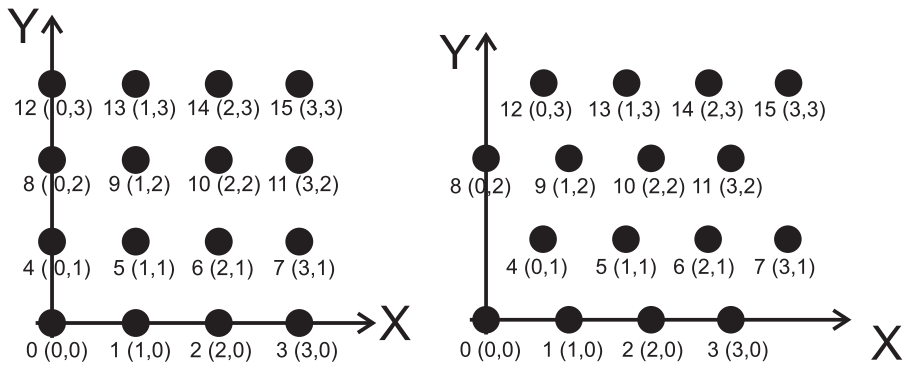
PLAPACK [58]
PLAPACK is a package with functionality similar to that of ScaLAPACK but with a different calling style. It also relies on optimized BLAS routines and is therefore able to achieve a high performance. Whereas ScaLAPACK uses a calling style that is similar to Fortran, to stay close to its LAPACK roots, PLAPACK uses a more object-oriented style. PLAPACK are exploited in the PETSc package.

5.2 MoM Setup and Mesh

The implemented procedure starts by selecting the geometry to be analyzed, via the CAD interface provided by GMSH [59]. With that tool, a custom ge-

ometry can be drawn, or a pre-existent model can be imported; for each object, discretization accuracy can be specified, so that the subsequent mesh operation can be customized.

The code is tailored to treat three different arrangement situations: single element structures, where the geometry is analyzed "as is", infinite arrays, where the geometry is considered as a unit cell and replicated via periodic boundary conditions exploiting Floquet theorem, and finite arrays, which are formed by brute-force mesh replication of the analyzed geometry. Acceleration techniques will decrease the computational complexity of the finite array problem, by exploiting results obtained from infinite array solutions [18].



(a) Rectangular Grid (b) Skewed Grid

Figure 5.2: Array arrangements

Then, options are read from an options file, and mesh accuracy is specified in a GMSH .dat file. Feeding edges, if present, are labeled differently, so that they can be recognized by the program in the feed assignment phase. The input of the procedure is a .geo file, deriving from a GMSH scipted drawing, or an import of a step file. According to discretization accuracy, the structure is then meshed automatically, calling the GMSH mesher via a system call. An output .msh file, containing the mesh, is produced. The exported mesh .msh file is parsed and transformed in a tuple of 3 files:

- a Points file, containing a list of all mesh points' coordinates as triplets (X, Y, Z)
- a Triangles file, containing a list of all mesh triangles. Each triangle is identified by its 3 vertices $(P1, P2, P3)$, each vertex corresponding to a point in the Points file, hence forming a connectivity list
- a Feeding edges file, containing, if the problem is not a scattering one, a list of all edges labeled as feeding ones

Table 5.1: MoM code parameters

Parameter	Array mode		
	Single el.	Infinite	Finite
Elements feeding /Plane wave mode	√	√	√
Direct/Iterative solution	√	√	√
Preconditioner/Iterative method	√	√	√
Far field 3D & 2D cuts	√	√	√
Feeding edges loading	√	√	√
No. of array elements, magnitudes			√
Elements spacing, skew and phase shift		√	√
Periodic Green function method		√	

In the case of finite array, the mesh is replicated accoding to spacings and skew in the options file. Points and feeding edges are replicated, and the connectivity list is extended to include the new triangles. In this way, only the single element is actively meshed, while the other elements' mesh is just replicated. Then, the index of the array element to which the entity belongs is added as a fourth term in all three files.

Once the geometry realization step is completed, the focus shifts to the definition of the analysis parameters: Table 5.1 presents a list of some customizable settings. Once all variables are defined, the surface meshing phase can begin, exploiting the built-in GMSH mesh generator, whose accuracy has been previously specified. A Python routine reads the parameters' configuration file and calls the *Delaunay 2D* GMSH mesher. The output mesh file is then parsed and mesh points are extracted, together with mesh triangles (each identified by its three vertices), and feeding edges.

5.3 MoM Pre-Process

The Pre-Process phase's purpose is that of building the impedance matrix and the RHS of the MoM linear system of equations, using the techniques described in Section 4.2 and 4.3. This routine reads points, triangles and feeding edges previously extracted from the mesh file, then identifies every inner edge of the structure and assigns a RWG basis function to each of them. Then, each mesh triangle is divided in 9 sub-triangles via barycentric subdivision [60], to ease numerical evaluation of integral quantities; RWG position vectors for triangles and sub-triangles are created.

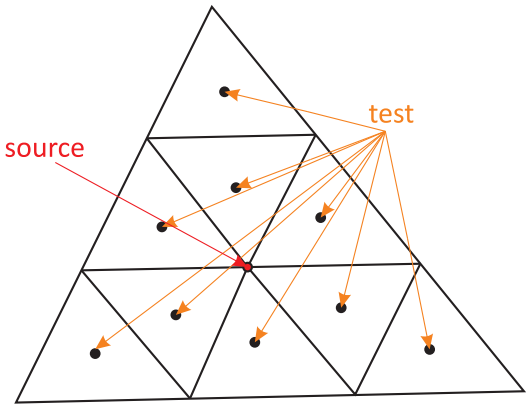


Figure 5.3: Integration strategy

5.3.1 Integration Strategy

A 9-point quadrature is used for all integrals in Section 5.3: since the integrand can be assumed constant within each sub-triangle, the integral over triangle T_m, whose area is A_m, can be calculated as a sum over 9 integration points [14]:

$$\frac{1}{A_m}\int_{T_m} g(r)dS = \frac{1}{9}\sum_{k=1}^{9} g(r_k^c). \tag{5.1}$$

In this way the quadrature points will not coincide with the triangle's midpoint, hence no singularity will occur, so no singularity extraction should be performed.

5.4 MoM Solver

In this phase the impedance matrix and RHS of the MoM linear system of equations created in the Pre-Process Section are read, and the current solution vector is allocated. Finally the MoM system of equations is solved

$$[Z_{mn}]\,[I_n] = [V_m] \tag{5.2}$$

Since all data structures are allocated as PETSc containers, a vast range of preconditioners and solvers is available. Moreover, an arbitrary number of concurrent processes can contribute to the calculation of the solution. If the Z matrix's size is smaller than the available memory, direct solution can be exploited via LU factorization; this is generally the faster and more accurate method. If, on the other hand, the matrix size is larger, several iterative Krylov methods [61] can be exploited. A good combination for dense matrices arising from MoM problems is GMRES, using Jacobi factorization as a preconditioner.

5.5 MoM Post Process

In this phase the Voltage and Current vectors created in the previous Sections are read. Then, the current density on each triangle is determined as a sum of all RWG currents: the surface current on the n-th triangle due to the m-th edge is

$$\mathbf{I}_n^m = \sum_m l_m I_m \left(\boldsymbol{\rho}_{nm}^{c+}/2A_n^+ + \boldsymbol{\rho}_{nm}^{c-}/2A_n^-\right) \tag{5.3}$$

where l_m and I_m are the m-th edge's length and current. Moreover, for *transmission* problem the feeding currents, voltages, impedances and power are calculated.

Finally, 3D and 2D field calculations are performed, exploiting a dipole model: given a couple of triangles corresponding to a RWG, the corresponding dipole is built, to evaluate the field generated: considering the centre of the dipole as

$$\mathbf{c} = \frac{1}{2}(\mathbf{r}_m^{c+} + \mathbf{r}_m^{c-}) \tag{5.4}$$

and its moment as

$$\mathbf{m} = l_m I_m (\mathbf{r}_m^{c-} - \mathbf{r}_m^{c+}) \tag{5.5}$$

3D and 2D field calculations are performed by superposition of fields from each dipole, via the exact formulation provided in [14]. valid in near and far field:

$$\mathbf{H}_m = \frac{jk}{4\pi}\mathbf{m} \times \mathbf{r}\ Ce^{-jkr} \tag{5.6}$$

and

$$\mathbf{E}_m = \frac{\eta}{4\pi}\left((\mathbf{M} - \mathbf{m})\left[\frac{jk}{r} + C\right] + 2\mathbf{M}C\right)e^{-jkr} \tag{5.7}$$

where $C = \frac{1}{r^2}\left[1 + \frac{1}{jkr}\right]$ and $\mathbf{M} = \frac{(\mathbf{r}\cdot\mathbf{m})\mathbf{r}}{r^2}$.

Moreover, far field quantities, like Poynting vector, radiation density, radiation intensity and radiated power are calculated:
Poynting vector $[W/m^2]$

$$\mathbf{S} = \frac{1}{2}\ Re\{\mathbf{E} \times \mathbf{H}^*\} \tag{5.8}$$

Radiation density $[W/m^2]$

$$W_m = ||\mathbf{S}|| \tag{5.9}$$

Radiation intensity $[W/\text{unit solid angle}]$

$$U_m = r^2 W_m \tag{5.10}$$

Radiated power $[W]$

$$P_{rad} = \sum_m W_m \tag{5.11}$$

Gain (dB)

$$G = 10\log_{10}(4\pi U_m/P_{rad}) \tag{5.12}$$

For the *transmission* problem, the radiation resistance [Ω] is

$$R_R = 2P_{rad}/I_{feed}^2 \tag{5.13}$$

5.6 Output

Output quantities like mesh, surface currents and gain are calculated from data exported from the entire MoM procedure, exploiting Matplotlib [52] and Mayavi [53] Python routines. These quantities will be presented during results' comparison in Chapter 7.

5.6.1 Mesh view

This routine reads the Points file and the Triangles file, and visualizes the meshed structure in a Mayavi frame.

5.6.2 Surface current plot

This routine reads the Surface Current exported file, the Points file and the Triangles file, and visualizes the surface current on each mesh triangle in a Mayavi frame.

5.6.3 Gain plot

This routine reads the Radiated Power exported file, and the meshed far-field sphere; then, the far field sphere is deformed: for each triangle, the spherical coordinates of its vertices are calculated, and their radius replaced by the value of the calculated Gain at that point. The gain on each triangle of the deformed sphere is finally visualized in a Mayavi frame, producing a conventional 3D-gain plot. 2D cuts are calculated on the following cuts:

$$\text{XY} \begin{cases} \theta = 90° \\ \phi \in [0°, 360°) \end{cases} \quad \text{XZ} \begin{cases} \theta \in [-180°, 180°) \\ \phi = 0° \end{cases} \quad \text{YZ} \begin{cases} \theta \in [-180°, 180°) \\ \phi = 90° \end{cases} \tag{5.14}$$

The calculated far-field plot 2D cuts are then visualized in polar and cartesian plots in classical Matplotlib frames.

Chapter 6

Array Scanning Method-Macro Basis Function

A class of methods consisting in aggregating basis functions of finite arrays into relatively small sets, each defined over every unit cell of the array, is becoming popular. The aggregated basis functions are sometimes named Macro Basis Functions (MBFs) or Characteristic Basis Functions (CBFs).

A possible implementation of these techniques is focused on the exploitation of the Array Scanning Method. The Array Scanning Method (ASM) [20] allows the determination of the behavior of an active antenna (or any single-source excitation) in an infinite passively terminated array from results obtained for fully periodic array excitations, while accounting for all the effects of mutual coupling [17]. The main idea is to be able to recuperate most of the side products of infinite-array analysis to efficiently analyze finite arrays [18]. Indeed, what is looked for is the current distribution over the array for excitation at any element of the array. When only one element is excited, the main difference between infinite and finite arrays is that, in the latter, currents may be reflected or scattered from the array edges. It is expected that, when the current wave propagates back toward the inside of the array (and may bounce on other edges of the array), the new current distributions created in the array are similar to those obtained in the forward wave case. The latter correspond to the distributions provided by the ASM.

For arrays of complex elements, it is difficult to track the multiple reflections on the edges of the array and to find the corresponding reflection coefficients. However, the above reasoning allows to think that the current distributions obtained on successive elements with the ASM will form *a good basis* for current distributions in finite arrays. In practice, the ASM integral will be discretized, which leads to an aliasing phenomenon (implicit repetition of the source; see dashed lines in Fig. 6.5). Nevertheless, the ASM discretized with very few samples forms a very good basis for currents in infinite arrays. Those current distributions, obtained from infinite and finite arrays, will form a set of *Macro*

Basis Functions. This method is known under the name ASM-MBF technique [62]. Validations carried out with the ASM-MBF for arrays of tapered-slot antennas showed that very high accuracy can be achieved with the application of the ASM with a very low order, i.e., with the ASM integral computed from typically 2 to 4 points in both directions. Hence, the ASM-MBF will allow for the reduction of the effective number of unknowns per antenna by typically 1-2 orders of magnitude when complex elements are considered. This means that relatively large arrays can now be treated in direct form, i.e., by direct solution of a reduced system of equations, whose matrix of coefficients can be inverted (or LU-decomposed) once and for all, such that solutions for all possible excitations are readily obtained. An open-source code for the ASM-MBF applied to the simple case of linear arrays of dipoles has been described in [17]. A previous attempt focused on the application of the ASM-MBF method to 2D planar arrays, sharing some common aspects with the present work, is in [63].

Sections 6.1 will sketch the basics differences between finite and infinite array simulations; Sections 6.2 will introduce the Array Scanning Method, while the exploitation of ASM solutions as Macro Basis Functions will be described in Sections 6.3.

6.1 Finite and Infinite Array Solutions

The starting point for the analysis of both finite and infinite arrays will be the current distributions on an infinite array when only one element is excited. When a single element is excited, different current distributions are found on successive elements, with magnitudes which, most of the time, decay away from the excited element. This is schematically represented in Fig. 6.1. By simple superposition, currents in infinite arrays with constant magnitudes and interelement phase shift ψ are obtained (Fig. 6.2). In a finite array, fields have a distribution similar to that of the infinite array, except for edge elements, due to the contributions reflected by the array ends (Fig. 6.3). These considerations will be important in the analysis of the ASM.

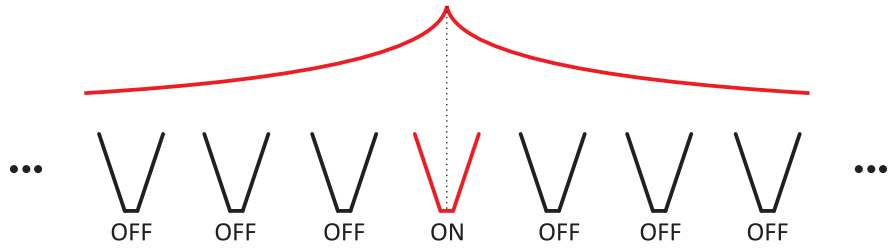


Figure 6.1: Infinite array: one element excited.

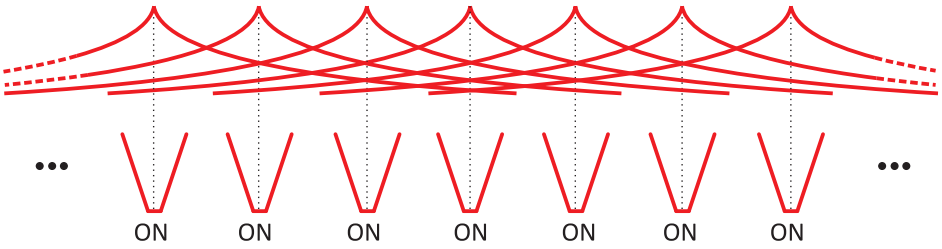


Figure 6.2: Infinite array: all elements excited, with constant amplitude and linear phase progression.

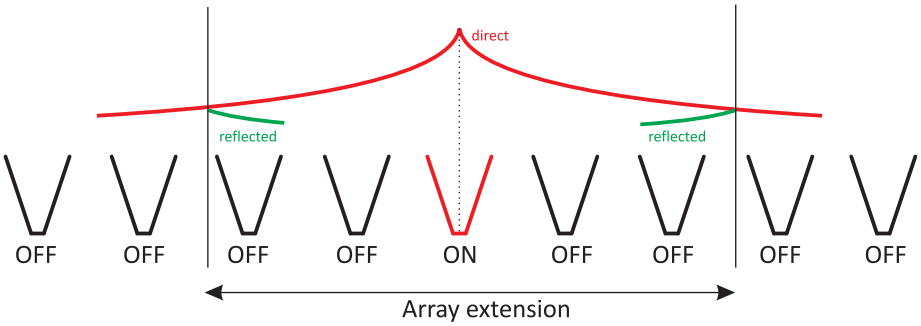


Figure 6.3: Finite array: one element excited (red curve). Fields have a distribution similar to that of the infinite array, except for edge elements, due to the contribution reflected by the array ends (green curve).

6.1.1 Array Nomenclature

The following is a list of conventions used throughout the description of the techniques exploited for the analysis of infinite and finite arrays:

- N_a = number of basis functions for each single array element;
- N_x, N_y = number of array elements in X and Y direction;
- $N_{el} = N_x \times N_y$ = total number of array elements;
- $N = N_a \times N_{el}$ = total number of basis functions for the entire finite array;
- N_ψ = number of points in discretized ASM;
- N_{mbf} = number of Macro Basis Functions for each single array element, after SVD;
- $P = N_{mbf} \times N_{el}$ = total number of Macro Basis Functions for the entire finite array.

6.2 Array Scanning Method

Consider an array with periodicity d_x and d_y along the x and y direction with periodic excitation at the feed-points level; the inter-element phase shift are given by ψ_x and ψ_y. Denoting a given field in the reference unit cell by $I^{\infty}(\psi_x, \psi_y)$, then the field obtained in cell (i_x, i_y) when only the source in the reference unit cell is excited is given by:

$$I(i_x, i_y) = \frac{1}{(2\pi)^2} \int_0^{2\pi} \int_0^{2\pi} I^{\infty}(\psi_x, \psi_y)\ e^{-j(i_x\psi_x + i_y\psi_y)} \mathrm{d}\psi_x \mathrm{d}\psi_y, \tag{6.1}$$

In this way, current on element i_x, i_y is calculated via superposition of infinite array simulations, in which only the unit reference cell is excited. A discretized version of this integral, calculated over N_ψ for each direction, hence a total N_ψ^2 points for each array element, is:

$$I(i_x, i_y) = \frac{1}{(N_\psi)^2} \sum_0^{N_\psi} \sum_0^{N_\psi} I^{\infty}(\psi_x, \psi_y)\ e^{-j(i_x\psi_x + i_y\psi_y)}, \tag{6.2}$$

where

$$\psi_x = 2\pi p/N_\psi\ ,\ \psi_y = 2\pi q/N_\psi\ ,\ p, q = 1, \ldots N_\psi. \tag{6.3}$$

The summation (6.2) has the same form as the FFT: the result of the discretization of integral (6.1) is that the obtained fields actually correspond to those excited by sources located every N_{psi} cells along the array. In other words, $I(i_x, i_y)$ becomes periodic along both directions with period N_{psi}, as illustrated in Fig. 6.5. To avoid this aliasing problem, a possible solution may consist of artificially increasing the number of sampling points by increasing N_{psi}. It may be expected that the aliasing that characterizes the discretized ASM may severely degrade the ability to represent current distributions in the finite array excited by one element only. However, it is important to recall that, in a first instance, what is looked for is just a representative basis for current distributions. In this respect, the overlapping between aliased solutions is not a major difficulty, since the different superimposed solutions are all physical. Of course, for larger values of N_{psi} in the ASM, more current distributions are produced and a higher accuracy can be achieved. Finally, since the ASM solutions are themselves superpositions of infinite-array solutions with constant-amplitude excitation with linear phase progression, the latter may equally well serve as MBFs. Hence, the ASM does not need to be explicitly calculated to form the MBFs, but for each array element the term

$$I_{p,q}(i_x, i_y) = \frac{1}{N_\psi^2} I_{p,q}^{\infty}\ e^{-j(i_x\psi_x + i_y\psi_y)}, \tag{6.4}$$

shall be computed, where $i_x = 1, \ldots N_x$, $i_y = 1, \ldots N_y$.

A key point is that, upon forming MBFs, it is good to have the phase shifts ψ_x and ψ_y of the infinite-array solution uniformly distributed in $[0, 2\pi)$ or

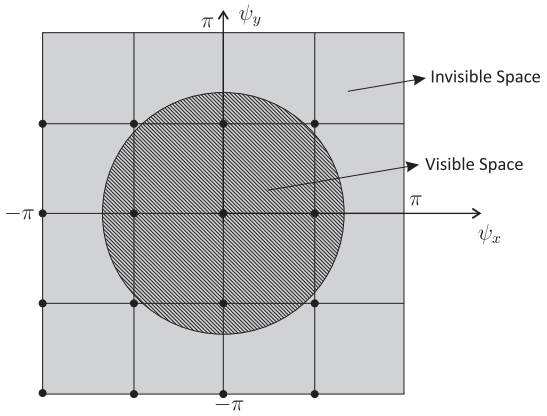


Figure 6.4: Uniform sampling in ψ space for phase shifts in infinite array solutions

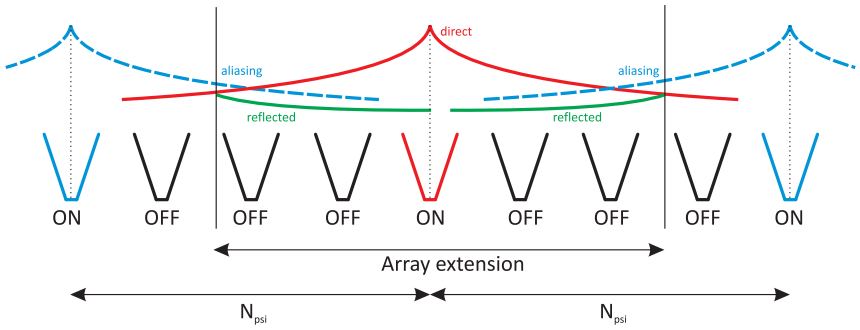


Figure 6.5: Finite array, Array Scanning Method: one element excited (red curve). Fields is reflected by the array ends (green curve). Discretized ASM leads to periodic source aliasing (blue curve).

$[-\pi, \pi)$ (Fig. 6.4). This means that, as soon as the element spacing is smaller than half a wavelength, solutions from outside the visible space (i.e., with active impedances that are purely reactive) will be necessary to form a good set of MBFs. Finally, for each array element, a threshold Singular Value Decomposition (SVD) decomposition is performed, to remove linearly dependent values and improve condition number.

Considering $A_{m\times n}$ as the ASM matrix, SVD decomposition consists of

$$\underset{m\times n}{\left[\begin{array}{c} A \end{array}\right]} = \underset{m\times m}{\left[\begin{array}{c} U \end{array}\right]} \underset{m\times n}{\left[\begin{array}{c} S \end{array}\right]} \underset{n\times n}{\left[\begin{array}{c} V^H \end{array}\right]} \tag{6.5}$$

where $U_{m\times m}$, $V_{n\times n}$ are unitary matrices, and $S_{m\times n}$ is a diagonal matrix containing the singular values of A. By choosing a convenient threshold, only the

first r singular values are retained, hence transforming the matrix to

$$\underset{m \times n}{\begin{bmatrix} A_r \end{bmatrix}} = \underset{m \times r}{\begin{bmatrix} U_r \end{bmatrix}} \underset{r \times r}{\begin{bmatrix} S_r \end{bmatrix}} \underset{r \times n}{\begin{bmatrix} V_r^H \end{bmatrix}} \tag{6.6}$$

The r columns of U_r or the r rows of V_r^H are an orthonormal basis. By the application of this procedure to the ASM matrix, the number of MBF is reduced from N_ψ^2 to N_{mbf}.

6.3 ASM as Macro Basis Functions

Once ASM solutions have been calculated, they can be concatenated in matrix form, in a three-dimensional $\mathbf{Q}$ matrix. Each slice of $\mathbf{Q}$ corresponds to the list of coefficients of a given MBF in terms of elementary basis functions. On each block i, j, the solution $\mathbf{x}_{i,j}$ is approximated by:

$$\mathbf{x}_{i,j} \simeq \mathbf{Q}_{i,j}\mathbf{y}_{i,j} \tag{6.7}$$

where each slice of $\mathbf{Q}$ corresponds to a MBF, i.e. to a solution, in terms of elementary basis functions, of one of the smaller problems referred to above. The main expectation is that the number of slices of $\mathbf{Q}$, i.e., the number of MBFs, be much smaller than the number of unknowns on the unit cell, which is often the case by 1-2 orders of magnitude. In the same way as macro basis functions are defined, macro testing functions are determined, hence exploiting a Galerkin approach. Reductions takes place block by block: each block of the finite array impedance matrix is extracted and reduced via $Q_i^H * Z_{ij} * Q_j$ (6.8), where the H suffix denotes transposed conjugate. Block size is reduced from $N_a \times N_a$ to $N_{mbf} \times N_{mbf}$.

$$\underset{N_{mbf} \times N_{mbf}}{\begin{bmatrix} Z_{ij}^{red} \end{bmatrix}} = \underset{N_{mbf} \times N_a}{\begin{bmatrix} Q_i^H \end{bmatrix}} \underset{N_a \times N_a}{\begin{bmatrix} Z_{ij} \end{bmatrix}} \underset{N_a \times N_{mbf}}{\begin{bmatrix} Q_j \end{bmatrix}} \tag{6.8}$$

Each block of the finite array voltage vector is extracted and reduced via $Q_i^H * V_i$ (6.9). Block size is reduced from N_1 to N_{mbf} unknowns

$$\underset{N_{mbf}}{\begin{bmatrix} V_i^{red} \end{bmatrix}} = \underset{N_{mbf} \times N_a}{\begin{bmatrix} Q_i^H \end{bmatrix}} \underset{N_a}{\begin{bmatrix} V_i \end{bmatrix}} \tag{6.9}$$

In this way, the original large $N \times N$ system of equations (6.10)

$$\underset{N \times N}{\begin{bmatrix} & Z & \\ & & \end{bmatrix}} \underset{N}{\begin{bmatrix} I \\ \end{bmatrix}} = \underset{N}{\begin{bmatrix} V \\ \end{bmatrix}} \tag{6.10}$$

is reduced to a $P \times P$ system of equations (6.11).

$$\underset{P \times P}{\begin{bmatrix} Z^{red} \\ \end{bmatrix}} \underset{P}{\begin{bmatrix} I^{red} \\ \end{bmatrix}} = \underset{P}{\begin{bmatrix} V^{red} \\ \end{bmatrix}} \tag{6.11}$$

and V is reduced from N to P. The reduced system of equation is then solved, a reduced finite P array current vector being the solution. Each block of the finite array current vector is finally remapped to the initial number of variables: $Q_i * I_i$ (6.12): block size is remapped from N_{mbf} to N_a unknowns.

$$\underset{N_a}{\begin{bmatrix} I_i \\ \end{bmatrix}} = \underset{N_a \times N_{mbf}}{\begin{bmatrix} Q_i \\ \end{bmatrix}} \underset{N_{mbf}}{\begin{bmatrix} I_i^{red} \\ \end{bmatrix}} \tag{6.12}$$

In this way, I is remapped from P to the original N unknowns.

Chapter 7

Results and Benchmark Comparison

A comparison between the proposed code and the commercial code FEKO [12], used as reference, is presented in this Chapter. For a fair comparison, mesh is the same for both GMSH and for FEKO. This has been accomplished by specifying mesh accuracy via the GMSH Delaunay mesher, and then exporting that mesh in ASCII format. The mesh is then imported in FEKO, without the need for model re-creation. Excitations are then defined in FEKO as Mesh Ports. In Section 7.1 simulations pertaining to bowtie antennas will be presented, while Section 7.2 is focused on Tapered Slot Antennas. For each problem under investigation, the meshed geometry will be presented. Results comparison will be conducted over surface currents (5.3), 3D patterns and 2D pattern cuts (5.12), illustrated via the routines described in Section 5.6.
For arrays, *brute-force* solutions generated via mesh replication will be compared with the solution obtained with the help of the ASM-MBF method. All tests have been conducted at 10 GHz. Testing architecture is a Quadcore Xeon 5472 @ 3GHz, 32 GB RAM. For what concerns linear algebra packages, the presented MoM Code uses LAPACK for single process and PLAPACK for parallel excutions, while FEKO uses LAPACK for single process and SCALAPACK for parallel execution.

7.1 Bowtie simulations

7.1.1 PEC Bowtie, Coarse Mesh

A resonant $\lambda_0/2 \times \lambda_0/2$ bowtie antenna, with a $\lambda_0/200$ feed gap, is shown in Fig. 7.1. Mesh is finer in regions where currents are supposed to be stronger, hence where better accuracy is needed.

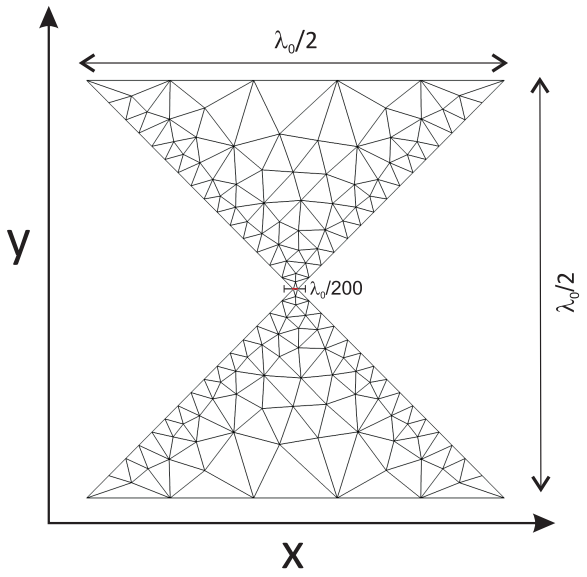
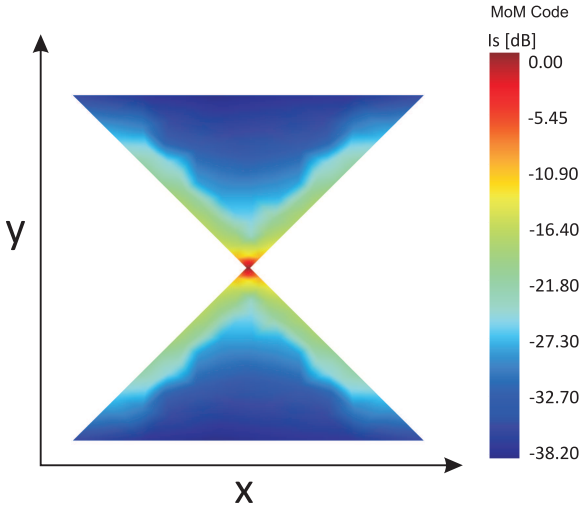
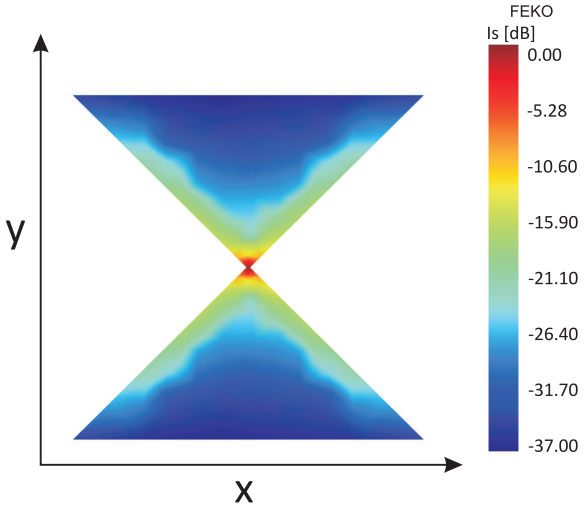


Figure 7.1: PEC Bowtie, Coarse Mesh.

Surface currents calculated after the solution of the MoM system of equations are in Fig. 7.2. Fig. 7.2a is relative to the MoM code, while Fig. 7.2b to FEKO: surface currents plots indicates a very good matching between the two solutions. 3D gain and pattern for the proposed MoM code is illustrated in Fig. 7.3. 2D cuts comparison in Fig. 7.4 shows an excellent agreement between the proposed solution and the commercial one. Finally, results in Table 7.1 show that computational burden and execution time are very similar between the two solutions. For this coarsely meshed geometry, MPI solution with 4 process is slower than the single process one: this is due to the slight computational overhead introduced by a parallel solution, which in this case is comparable to the execution time itself.



(a) MoM code



(b) FEKO

Figure 7.2: PEC Bowtie, Coarse Mesh: Surface Currents.

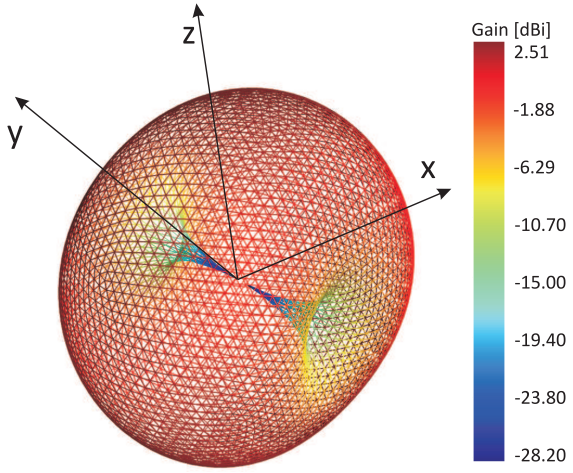
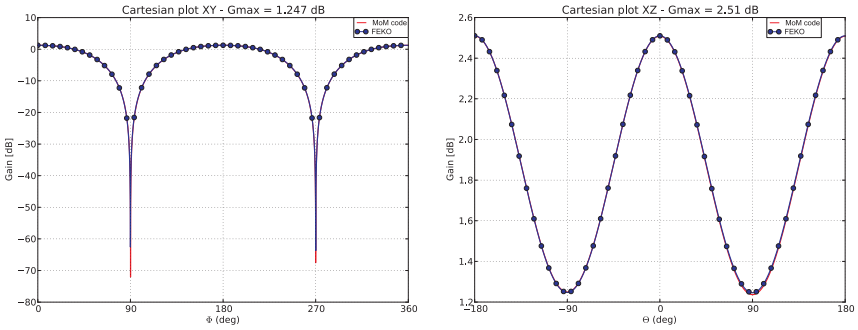
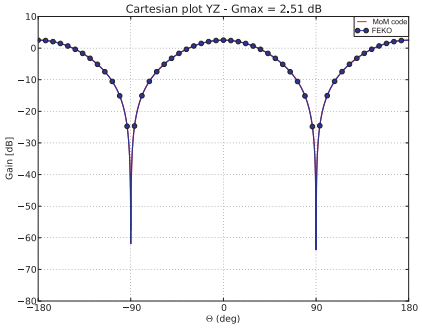


Figure 7.3: PEC Bowtie, Coarse Mesh: 3D pattern and gain.



(a) XY plane cut

(b) XZ plane cut



(c) YZ plane cut

Figure 7.4: PEC Bowtie, Coarse Mesh: 2D pattern cuts.

Table 7.1: PEC Bowtie, Coarse Mesh: execution time comparison

		MoM Code	FEKO
Triangles		256	
Edges		349	
Matrix Size		349 × 349	
Solution time	1 process	0.023 s	0.022 s
	4 process	0.085 s	0.078 s

7.1.2 PEC Bowtie, Fine Mesh

The same resonant bowtie antenna of Fig. 7.1 is in Fig. 7.5: a finer mesh is exploited in this case, to achieve better accuracy.

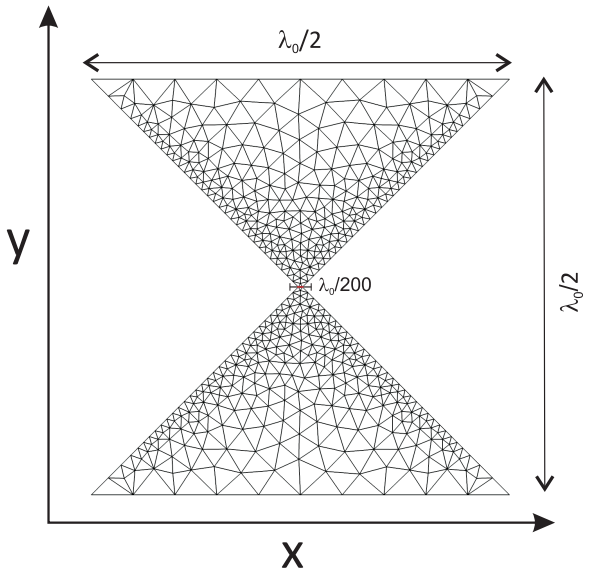
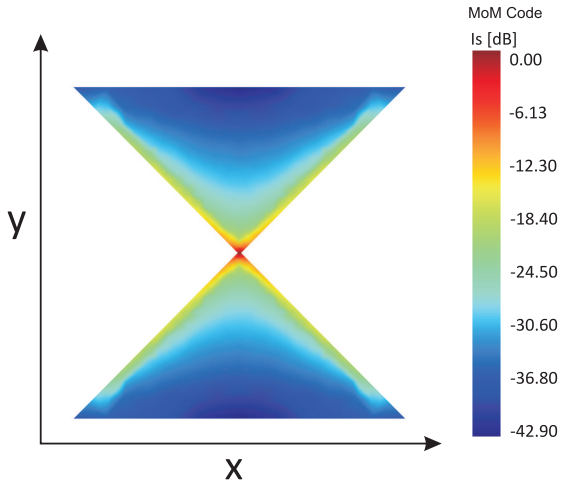
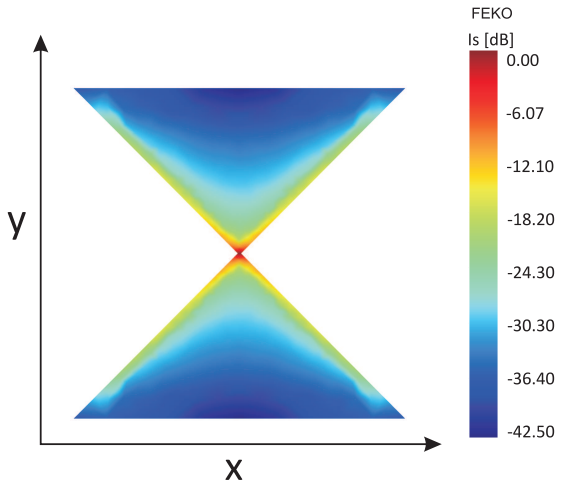


Figure 7.5: PEC Bowtie, Fine Mesh.

Surface currents calculated after the solution of the MoM system of equations are in Fig. 7.6. Fig. 7.6a is relative to the MoM code, while Fig. 7.6b to FEKO: surface currents plots indicates a very good matching between the two solutions also in this case. 3D gain and pattern for the proposed MoM code is illustrated in



(a) MoM code



(b) FEKO

Figure 7.6: PEC Bowtie, Fine Mesh: Surface Currents.

Fig. 7.7. 2D cuts comparison in Fig. 7.8 shows an excellent agreement between the proposed solution and the commercial one. Finally, results in Table 7.2 show that for single process, MoM code's execution time is faster than FEKO's. For parallel execution, FEKO's computation is sped up, while for the MoM code computational overhead is still significant, hence speed-up is modest, and execution time is comparable to FEKO's.

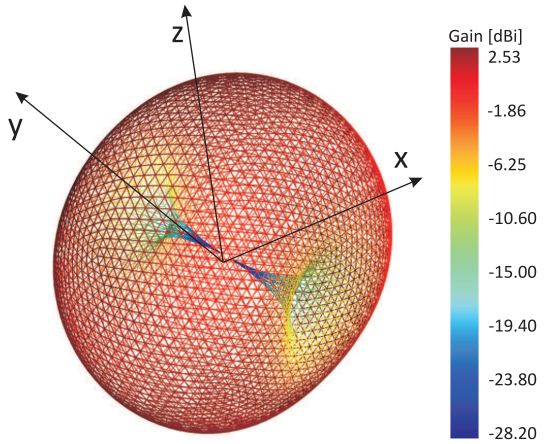
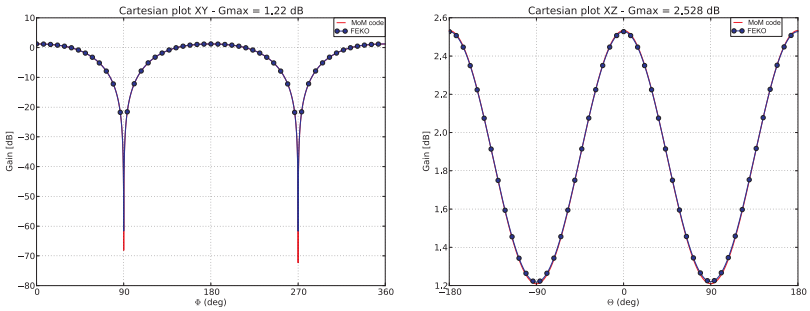
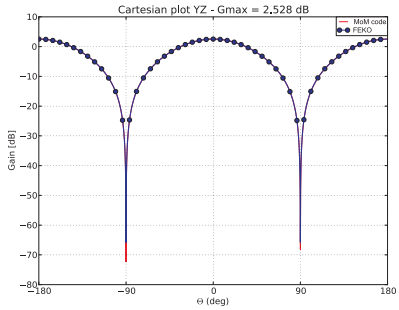


Figure 7.7: PEC Bowtie, Fine Mesh: 3D pattern and gain.



(a) XY plane cut

(b) XZ plane cut



(c) YZ plane cut

Figure 7.8: PEC Bowtie, Fine Mesh: 2D pattern cuts.

Table 7.2: PEC Bowtie, Fine Mesh: execution time comparison

		MoM Code	FEKO
Triangles		1030	
Edges		1471	
Matrix Size		1471×1471	
Solution time	1 process	0.58 s	0.98 s
	4 process	0.54 s	0.53 s

7.1.3 PEC Bowtie, Finite Array

The resonant bowtie antenna meshed as in Fig. 7.1 is replicated in an $N_x = 3$, $N_y = 4$ elements array, placed on a 60° skewed grid. Spacings are $d_x = 0.7\lambda_0$ and $d_y = 0.7\lambda_0$ (Fig. 7.9), and elements are fed with constant amplitude and phase.

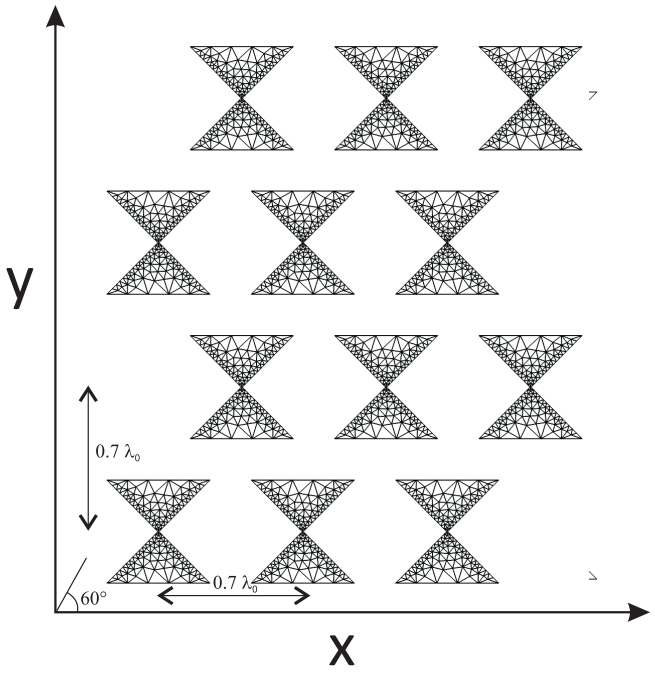
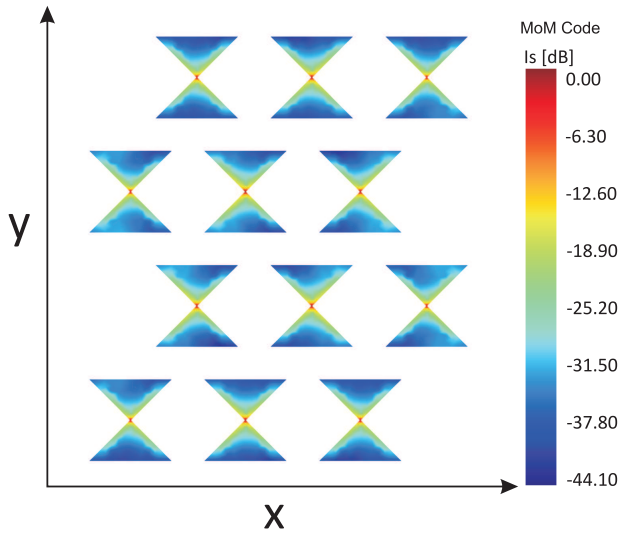


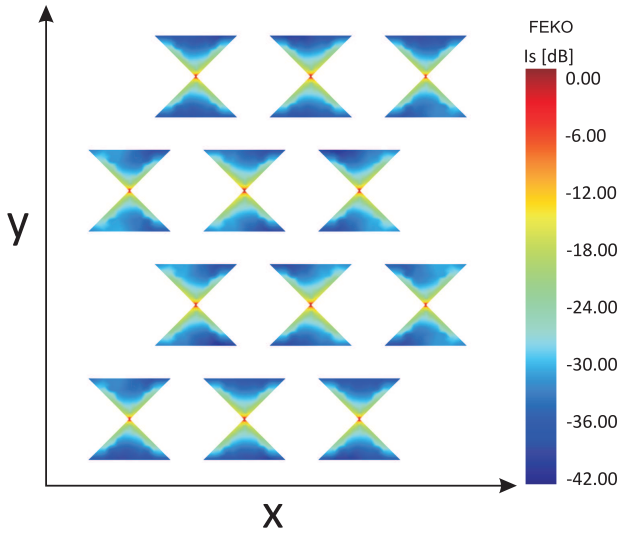
Figure 7.9: PEC Bowtie, Finite Array: Mesh.

Surface currents calculated after the solution of the MoM system of equations

are in Fig. 7.10. Fig. 7.10a is relative to the MoM code, while Fig. 7.10b to FEKO: surface currents plots indicates a very good matching between the two solutions also in the array case. 3D gain and pattern for the proposed MoM code



(a) MoM code



(b) FEKO

Figure 7.10: PEC Bowtie, Finite Array: Surface Currents.

is illustrated in Fig. 7.11. 2D cuts comparison in Fig. 7.12 shows an excellent

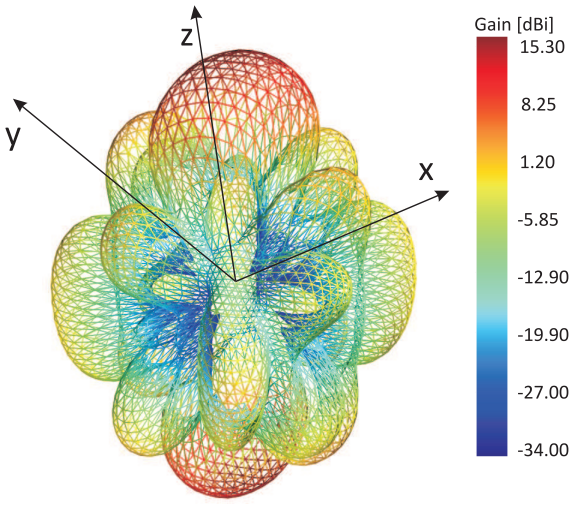


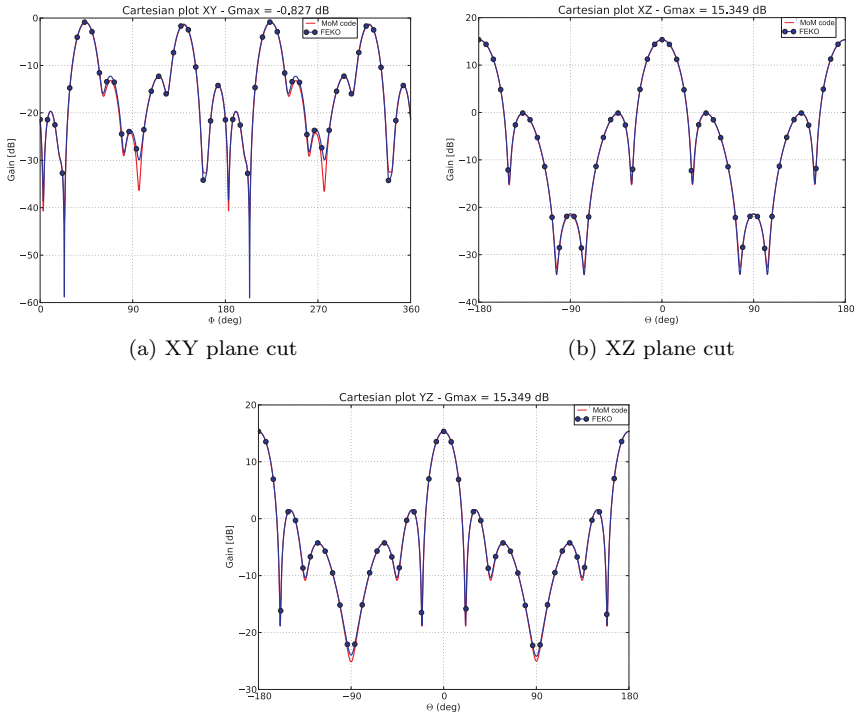
Figure 7.11: PEC Bowtie, Finite Array: 3D pattern and gain.

agreement between the proposed solution and the commercial one.
Results in Table 7.3 show that for single process, MoM code's execution time is way faster than FEKO's. For parallel execution, both computations are sped up, and performances are equivalent.

Table 7.3: PEC Bowtie, Finite Array: execution time comparison

		MoM Code	FEKO
Elements N_{el}		12	
Triangles per el.		256	
Edges per el. N_a		349	
Triangles tot.		3072	
Edges tot. N		4188	
Matrix Size		4188 × 4188	
Solution time	1 process	7.8 s	18.9 s
	4 process	6.7 s	6.7 s

Finally, results in Table 7.4 are relative to ASM-MBF acceleration. At first, a $N_\psi = 4$ solution has been exploited: SVD decomposition has not reduced the



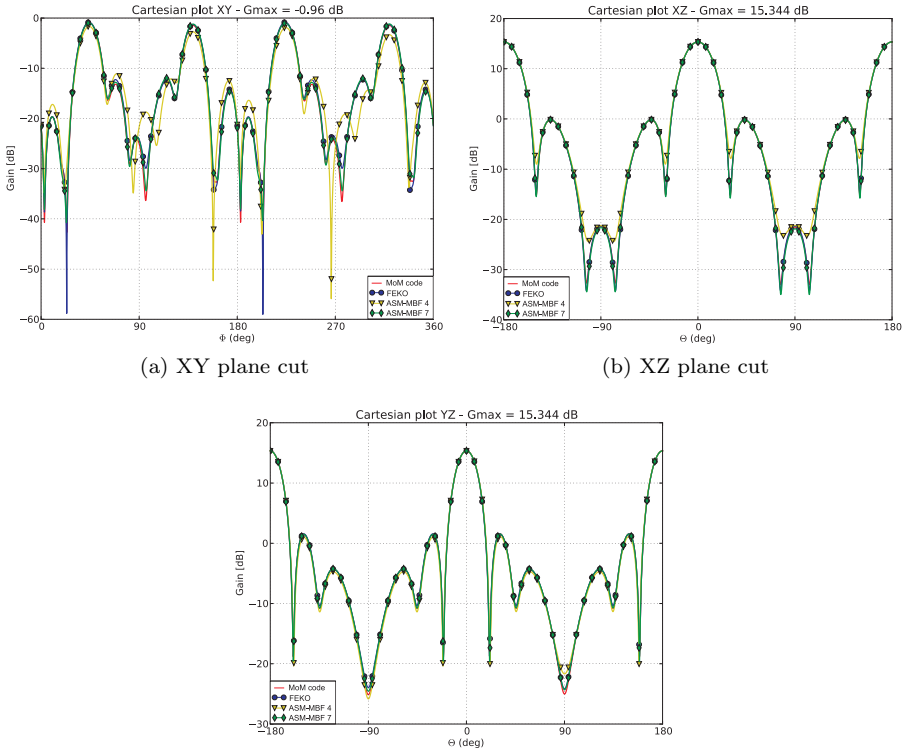
(a) XY plane cut

(b) XZ plane cut

(c) YZ plane cut

Figure 7.12: PEC Bowtie, Finite Array: 2D pattern cuts.

number of MBF, hence $N_{mbf} = N_\psi = 4$. In this way the original 4188×4188 system of equations has been reduced to a 48×48 size, hence a 87 times reduction in problem complexity. Total solution time is then the sum of the solution of 4 ASM problems, plus the solution of the reduced system of equations, hence $4 \times 0.023 + 0.0007 = 0.0927$ s, for an 84 times reduction in computational time. The $N_{mbf} = 4$ solution is depicted in Fig. 7.13 as a yellow curve: it is shown that this solution is not accurate with respect to the non-accelerated MoM approach or to FEKO.
Hence a more suitable a $N_\psi = 9$ solution has been exploited; after a reduction to $N_{mbf} = 7$ via SVD decomposition, the original 4188×4188 system of equations has been reduced to a 84×84 size, hence a 50 times reduction in problem complexity. Total solution time is then the sum of the solution of 9 ASM problems, plus the solution of the reduced system of equations, hence $9 \times 0.023 + 0.0018 = 0.2088$ s, for a 37 times reduction in computational time. The $N_{mbf} = 7$ solution is depicted in Fig. 7.13 as a green curve: accuracy is improved, and this solution is shown to have an excellent agreement with the non-accelerated MoM approach and FEKO.



(a) XY plane cut

(b) XZ plane cut

(c) YZ plane cut

Figure 7.13: PEC Bowtie, Finite Array accelerated via ASM-MBF: 2D pattern cuts.

Table 7.4: PEC Bowtie, Finite Array accelerated via ASM-MBF: execution time comparison

		ASM-MBF 4	ASM-MBF 7
Elements N_{el}		12	
ASM sampling pts. N_{ψ}^2		4	9
MBF per el. N_{mbf}		4	7
MBF tot. P		48	84
Reduced Matrix Size		48×48	84×84
ASM Sol. time	1 process	4×0.023 s	9×0.023 s
Reduced Matrix Sol. time		0.0007 s	0.0018 s
Total Solution time		0.0927 s	0.2088 s

7.2 Tapered Slot Antenna (TSA) simulations

7.2.1 PEC TSA, Coarse Mesh

A Tapered Slot antenna, whose dimensions are have been chosen to deliver a good feed matching and radiation performances, is shown in Fig. 7.14. Mesh is finer in regions where currents are supposed to be stronger, hence where better accuracy is needed.

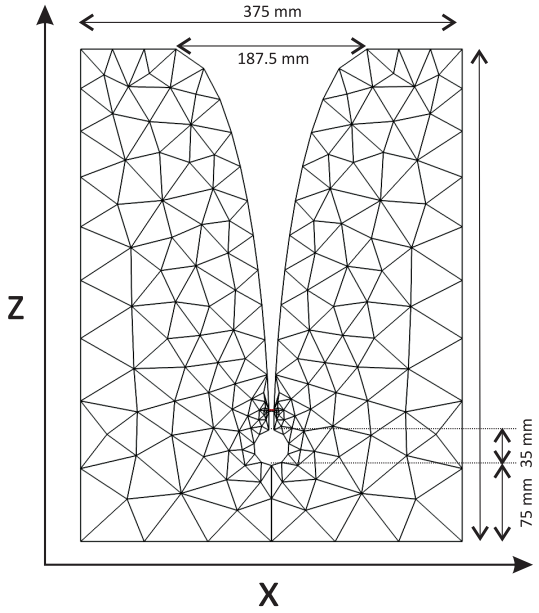
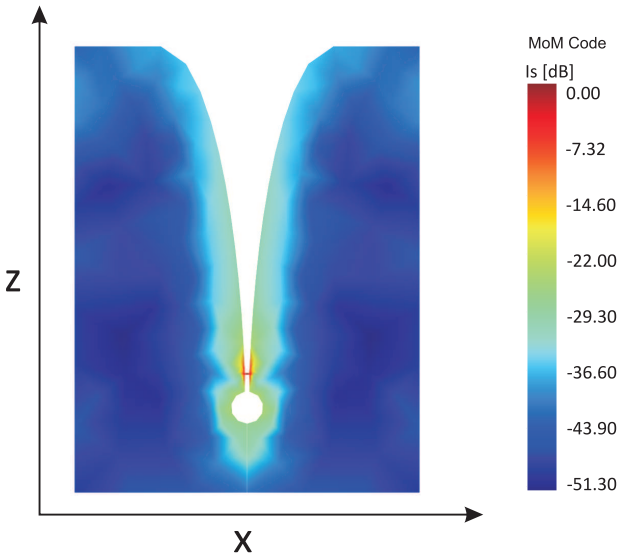
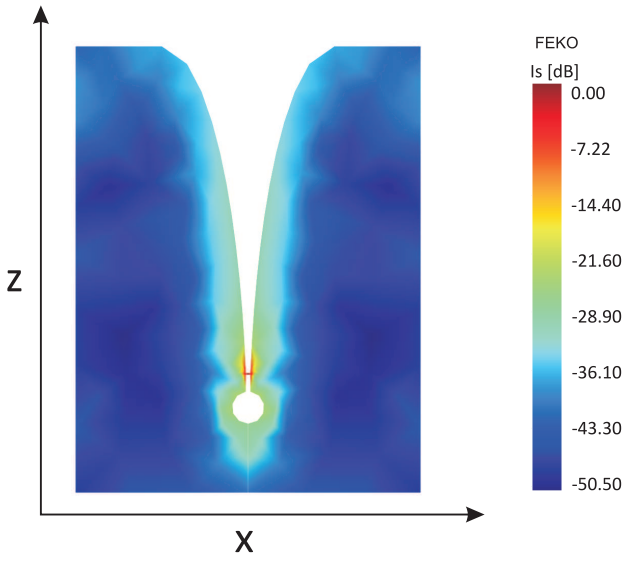


Figure 7.14: PEC TSA, Coarse Mesh.

Surface currents calculated after the solution of the MoM system of equations are in Fig. 7.15. Fig. 7.15a is relative to the MoM code, while Fig. 7.15b to FEKO: surface currents plots indicates a very good matching between the two solutions. 3D gain and pattern for the proposed MoM code is illustrated in Fig. 7.16. 2D cuts comparison in Fig. 7.17 shows an excellent agreement between the proposed solution and the commercial one.
Finally, results in Table 7.5 show that computational burden and execution time are very similar between the two solutions. Also for the TSA case, for this coarsely meshed geometry, MPI solution with 4 process is slower than the single process one: this is due to the slight computational overhead introduced by a parallel solution, which in this case is comparable to the execution time itself.



(a) MoM code



(b) FEKO

Figure 7.15: PEC TSA, Coarse Mesh: Surface Currents.

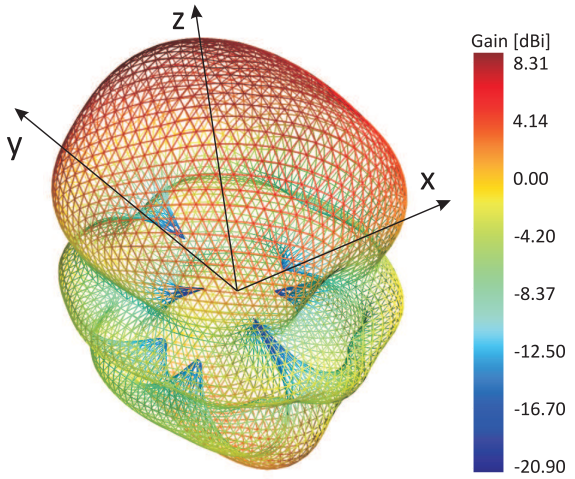
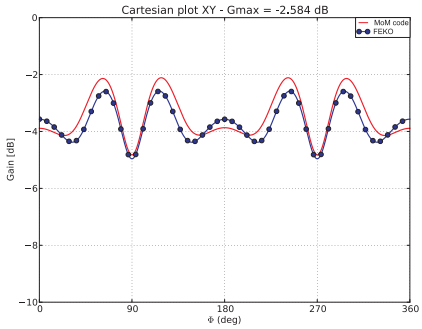
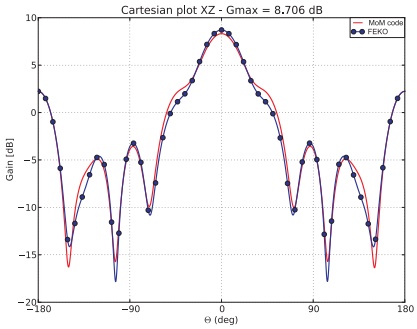


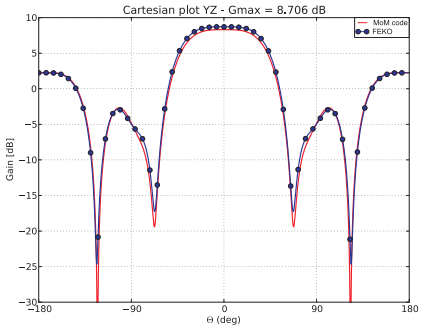
Figure 7.16: PEC TSA, Coarse Mesh: 3D pattern and gain.



(a) XY plane cut



(b) XZ plane cut



(c) YZ plane cut

Figure 7.17: PEC TSA, Coarse Mesh: 2D pattern cuts.

Table 7.5: PEC TSA, Coarse Mesh: execution time comparison

		MoM Code	FEKO
Triangles		338	
Edges		471	
Matrix Size		471×471	
Solution time	1 process	0.047 s	0.060 s
	4 process	0.128 s	0.235 s

7.2.2 PEC TSA, Fine Mesh

The same TSA antenna of Fig. 7.14 is in Fig. 7.18: a finer mesh is exploited in this case, to achieve better accuracy is needed.

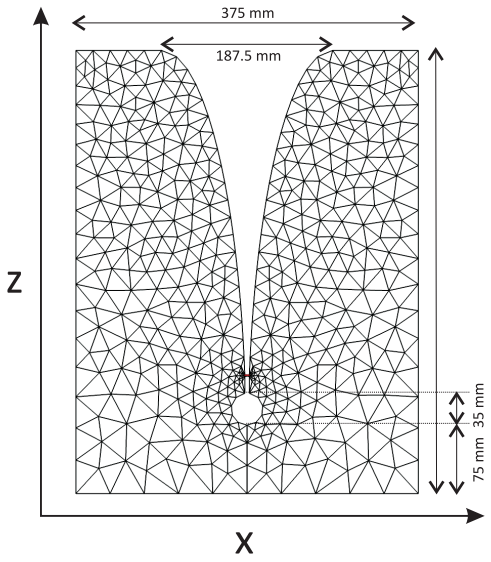
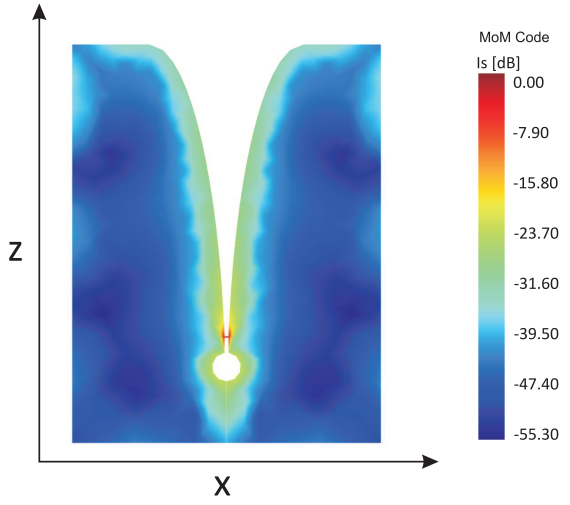
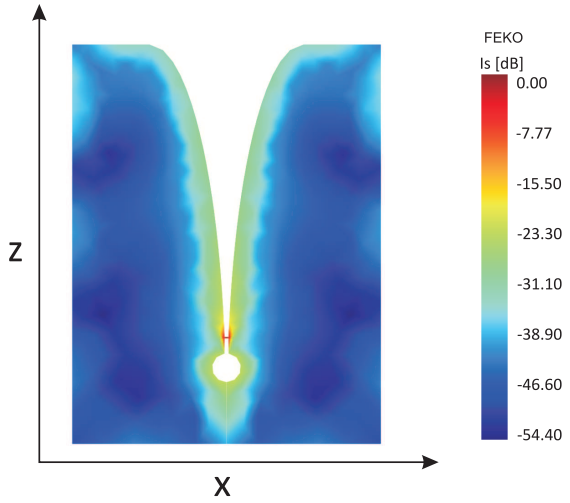


Figure 7.18: PEC TSA, Fine Mesh.

Surface currents calculated after the solution of the MoM system of equations are in Fig. 7.19. Fig. 7.19a is relative to the MoM code, while Fig. 7.19b to FEKO: surface currents plots indicates a very good matching between the two solutions also in this case. 3D gain and pattern for the proposed MoM code is illustrated in Fig. 7.20. 2D cuts comparison in Fig. 7.21 shows an excellent agreement between the proposed solution and the commercial one.



(a) MoM code



(b) FEKO

Figure 7.19: PEC TSA, Fine Mesh: Surface Currents.

Finally, results in Table 7.6 show that for single process, MoM code's execution time is faster than FEKO's. For parallel execution, MoM code and FEKO have similar execution time; for both solutions computational overhead is significant, hence MPI solution with 4 process is slower than the single process one.

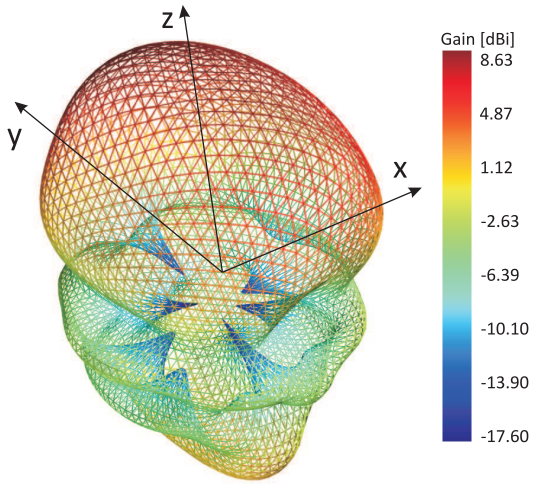
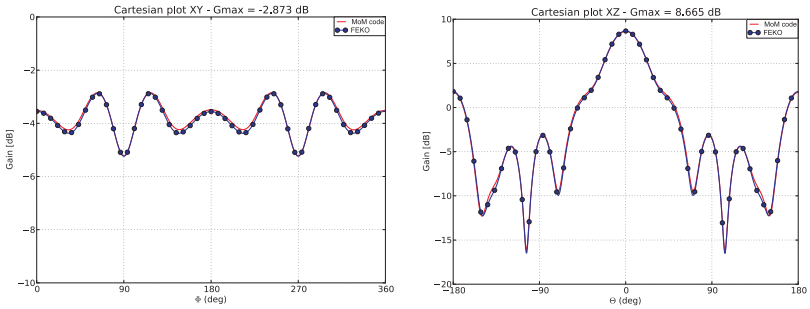
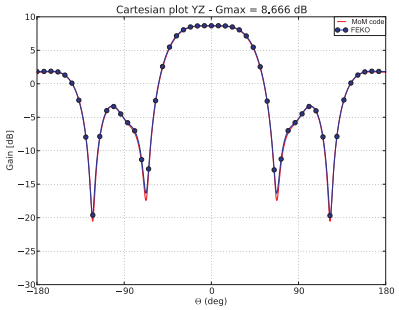


Figure 7.20: PEC TSA, Fine Mesh: 3D pattern and gain.



(a) XY plane cut

(b) XZ plane cut



(c) YZ plane cut

Figure 7.21: PEC TSA, Fine Mesh: 2D pattern cuts.

Table 7.6: PEC TSA, Fine Mesh: execution time comparison

		MoM Code	FEKO
Triangles		920	
Edges		1321	
Matrix Size		1321×1321	
Solution time	1 process	0.46 s	0.83 s
	4 process	1.17 s	1.16 s

7.2.3 PEC TSA, Finite Array

The TSA antenna meshed as in Fig. 7.14 is replicated in an $N_x = 3$, $N_y = 4$ elements array, placed on a rectangular grid. Spacings are $d_x = 1.3\lambda_0$ and $d_y = 0.5\lambda_0$ (Fig. 7.22), and elements are fed with constant amplitude and phase.

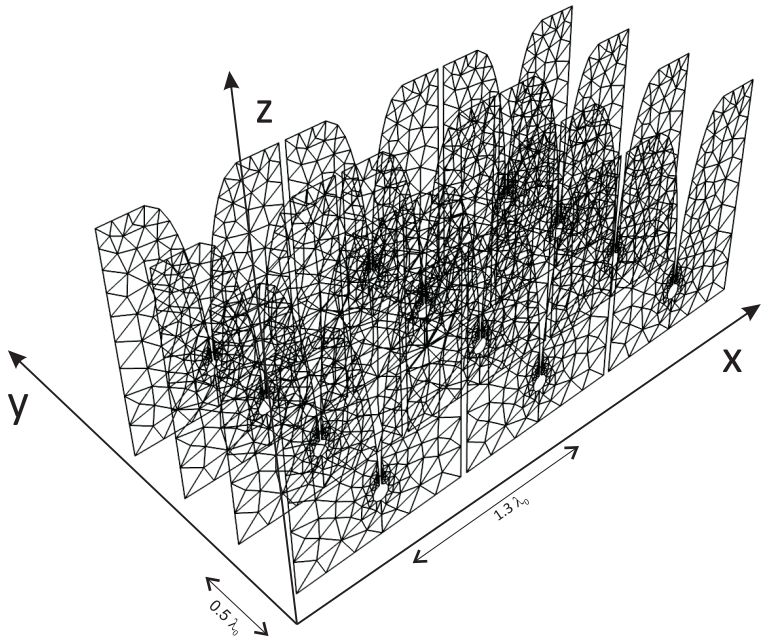


Figure 7.22: PEC TSA, Finite Array: Mesh.

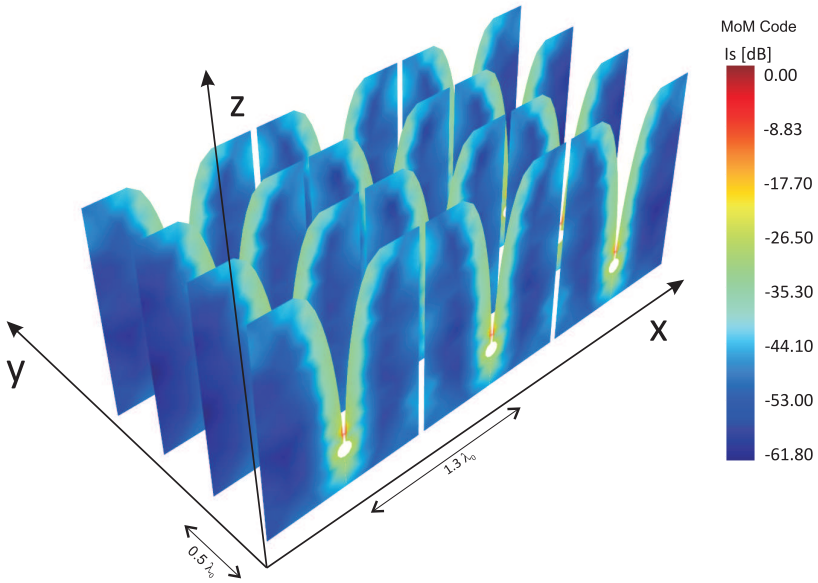
Surface currents calculated after the solution of the MoM system of equations

are in Fig. 7.23. Fig. 7.23a is relative to the MoM code, while Fig. 7.23b to FEKO: surface currents plots indicates a very good matching between the two solutions also in the array case. 3D gain and pattern for the proposed MoM code is illustrated in Fig. 7.24. 2D cuts comparison in Fig. 7.25 shows an excellent agreement between the proposed solution and the commercial one.
Results in Table 7.7 show that for single process, MoM code's execution time is way faster than FEKO's. For parallel execution, both computations are sped up, and performances are almost equivalent, with MoM code still faster than FEKO.

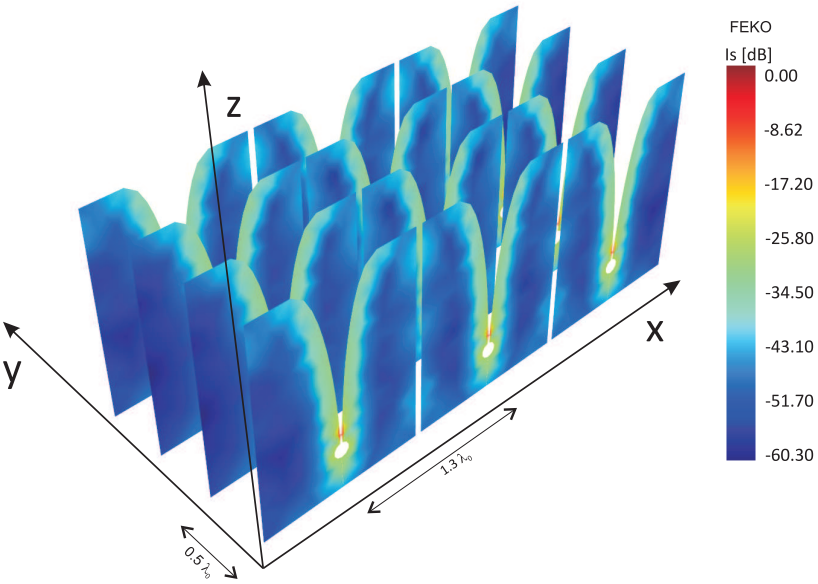
Table 7.7: PEC TSA, Finite Array: execution time comparison

		MoM Code	FEKO
Elements N_{el}		12	
Triangles per el.		338	
Edges per el. N_a		471	
Triangles tot.		4056	
Edges tot. N		5652	
Matrix Size		5652×5652	
Solution time	1 process	17.4 s	45.5 s
	4 process	14.8 s	17.2 s

Finally, results in Table 7.8 are relative to ASM-MBF acceleration. At first, a $N_\psi = 4$ solution has been exploited: SVD decomposition has not reduced the number of MBF, hence $N_{mbf} = N_\psi = 4$. In this way the original 5652×5652 system of equations has been reduced to a 48×48 size, hence a 118 times reduction in problem complexity. Total solution time is then the sum of the solution of 4 ASM problems, plus the solution of the reduced system of equations, hence $4 \times 0.047 + 0.0008 = 0.1888$ s, for an 92 times reduction in computational time. The $N_{mbf} = 4$ solution is depicted in Fig. 7.26 as a yellow curve: it is shown that this solution is not accurate with respect to the non-accelerated MoM approach or to FEKO.
Hence a more suitable a $N_\psi = 9$ solution has been exploited; after a reduction to $N_{mbf} = 5$ via SVD decomposition, the original 5652×5652 system of equations has been reduced to a 60×60 size, hence a 94 times reduction in problem complexity. Total solution time is then the sum of the solution of 9 ASM problems, plus the solution of the reduced system of equations, hence $9 \times 0.047 + 0.0011 = 0.4241$ s, for a 41 times reduction in computational time. The $N_{mbf} = 5$ solution is depicted in Fig. 7.26 as a green curve: accuracy is improved, and this solution is shown to have an excellent agreement with the non-accelerated MoM approach and FEKO.



(a) MoM code



(b) FEKO

Figure 7.23: PEC TSA, Finite Array: Surface Currents.

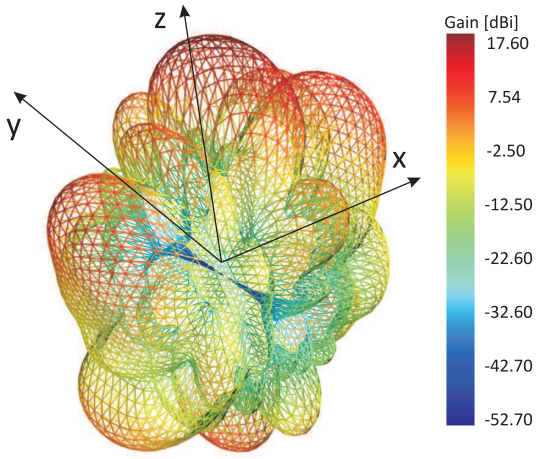
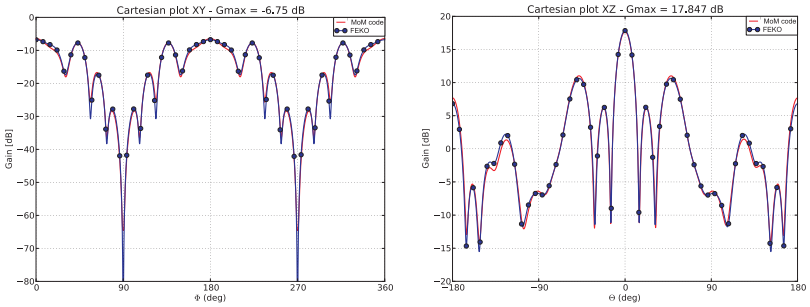
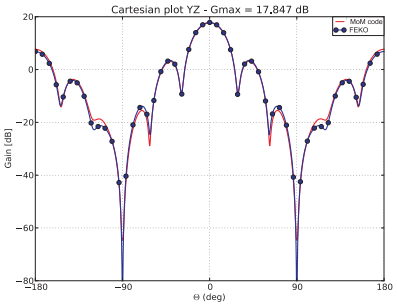


Figure 7.24: PEC TSA, Finite Array: 3D pattern and gain.



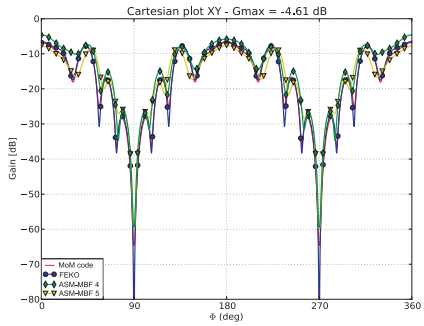
(a) XY plane cut

(b) XZ plane cut

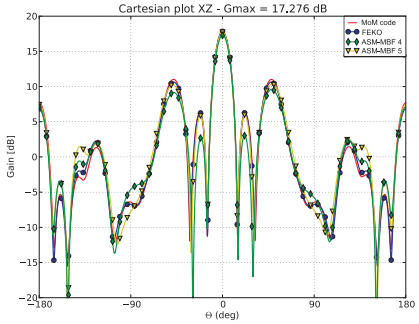


(c) YZ plane cut

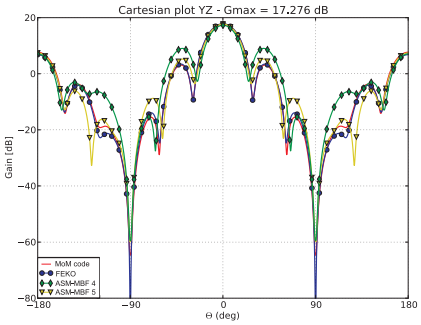
Figure 7.25: PEC TSA, Finite Array: 2D pattern cuts.



(a) XY plane cut



(b) XZ plane cut



(c) YZ plane cut

Figure 7.26: PEC TSA, Finite Array accelerated via ASM-MBF: 2D pattern cuts.

Table 7.8: PEC TSA, Finite Array accelerated via ASM-MBF: execution time comparison

		ASM-MBF 4	ASM-MBF 5
Elements N_{el}		12	
ASM sampling pts. N_{ψ}^2		4	9
MBF per el. N_{mbf}		4	5
MBF tot. P		48	60
Reduced Matrix Size		48×48	60×60
ASM Sol. time	1 process	4×0.047 s	9×0.047 s
Reduced Matrix Sol. time		0.0008 s	0.0011 s
Total Solution time		0.1888 s	0.4241 s

Bibliography for Method of Moments

[1] S. Ratnajeevan and H. Hoole. *Computer-aided Analysis and Design of Electromagnetic Devices*. Prentice Hall, 1988.

[2] M. N.O. Sadiku. *Numerical Techniques in Electromagnetics*. CRC Press, 2000.

[3] P. Silvester. "Finite-Element Solution of Homogeneous Waveguide Problems". In: *Alta Frequenza* 38 (1969).

[4] R. F. Harrington. *Field Computations by Moment Methods*. IEEE Press, New York, 1993.

[5] S.E. Forsythe and W.R. Wasow. *Finite Difference Methods for Partial Differential Equations*. Wiley, 1960.

[6] S. H. Gould. *Variational Methods for Eigenvalue Problems; An Introduction to the Methods of Rayleigh, Ritz, Weinstein, and Aronszajn*. Toronto: University of Toronto Press, 1957.

[7] C.A. Brebbia. *The Boundary Element Method for Engineers*. Pentech Press, 1978.

[8] G. Jeng and A. Wexler. "Isoparametric, Finite Element, Variational Solution of Integral Equations for Three-Dimensional Fields". In: *International Journal for Numerical Methods in Engineering* 11 (1977), 14551471.

[9] S. L. Ray Peterson A. F. and R. Mittra. *Computational Methods for Electromagnetics*. IEEE Press, New York, 1998.

[10] W.C. Gibson. *The Method of Moments in Electromagnetics*. Chapman and Hall/ CRC, 2008.

[11] W. Chew, M.S. Tong, and B. Hu. *Integral Equation Methods for Electromagnetic and Elastic Waves*. Synthesis Lectures on Computational Electromagnetics. Morgan & Claypool Publishers, 2008.

[12] *FEKO Suite 7.0, EM Software and Systems*. URL: http://www.feko.info.

[13] D. B. Davidson. *Computational Electromagnetics For RF and Microwave Engineering*. Cambridge University Press, 2005.

[14] S. Makarov. *Antenna and EM Modeling with MATLAB*. Wiley, 2002.

[15] J. Yeo, V. V. S. Prakash, and R. Mittra. "Efficient Analysis of a Class of Microstrip Antennas using the Characteristic Basis Function Method (CBFM)". In: *Microwave and Optical Technology Letters* 39.6 (2003), pp. 456–464. ISSN: 1098-2760. DOI: 10.1002/mop.11247.

[16] V. V. S. Prakash and Raj Mittra. "Characteristic Basis Function Method: A New Technique for Efficient Solution of Method of Moments Matrix Equations". In: *Microwave and Optical Technology Letters* 36.2 (2003), pp. 95–100. ISSN: 1098-2760. DOI: 10.1002/mop.10685.

[17] C. Craeye, B. A. Garca, E. Garca Muoz, and R. Sarkis. "An Open-Source Code for the Calculation of the Effects of Mutual Coupling in Arrays of Wires and for the ASM-MBF Method". In: *International Journal of Antennas and Propagation* (2010). DOI: 10.1155/2010/137903.

[18] C. Craeye and D. Gonzlez-Ovejero. "A Review on Array Mutual Coupling Analysis". In: *Radio Science* 46.2 (2011). DOI: 10.1029/2010RS004518.

[19] C. Craeye. "A Fast Impedance and Pattern Computation Scheme for Finite Antenna Arrays". In: *Antennas and Propagation, IEEE Transactions on* 54.10 (2006), pp. 3030–3034. ISSN: 0018-926X. DOI: 10.1109/TAP.2006.882202.

[20] B.A. Munk and G. Burrell. "Plane-wave Expansion for Arrays of Arbitrarily Oriented Piecewise Linear Elements and its Application in Determining the Impedance of a Single Linear Antenna in a Lossy Half-space". In: *Antennas and Propagation, IEEE Transactions on* 27.3 (1979), pp. 331–343. ISSN: 0018-926X. DOI: 10.1109/TAP.1979.1142089.

[21] S.M. Rao, D. Wilton, and A.W. Glisson. "Electromagnetic Scattering by Surfaces of Arbitrary Shape". In: *Antennas and Propagation, IEEE Transactions on* 30.3 (1982), pp. 409–418. ISSN: 0018-926X.

[22] R. F. Harrington. *Time-Harmonic Electromagnetic Fields*. McGraw-Hill, New York, 1961.

[23] T.K. Sarkar, S.M. Rao, and A.R. Djordjevic. "Electromagnetic Scattering and Radiation from Finite Microstrip Structures". In: *Microwave Theory and Techniques, IEEE Transactions on* 38.11 (1990), pp. 1568–1575. ISSN: 0018-9480. DOI: 10.1109/22.60001.

[24] K. Umashankar, Allen Taflove, and S.M. Rao. "Electromagnetic Scattering by Arbitrary Shaped Three-dimensional Homogeneous Lossy Dielectric Objects". In: *Antennas and Propagation, IEEE Transactions on* 34.6 (1986), pp. 758–766. ISSN: 0018-926X. DOI: 10.1109/TAP.1986.1143894.

[25] A.J. Poggio and E.K. Miller. "Integral Equation Solutions of Three-Dimensional Scattering Problems". In: *Computer Techniques for Electromagnetics*. CRC Press, 1973.

[26] E. Arvas, A. Rahhal Arabi, A. Sadigh, and S.M. Rao. "Scattering from Multiple Conducting and Dielectric Bodies of Arbitrary Shape". In: *Antennas and Propagation Magazine, IEEE* 33.2 (1991), pp. 29–36. ISSN: 1045-9243. DOI: 10.1109/74.88184.

[27] S.M. Rao, Chung-Chi Cha, R.L. Cravey, and D.L. Wilkes. "Electromagnetic Scattering from Arbitrary Shaped Conducting Bodies Coated with Lossy Materials of Arbitrary Thickness". In: *Antennas and Propagation, IEEE Transactions on* 39.5 (1991), pp. 627–631.

[28] A.W. Glisson, D. Kajfez, and J. James. "Evaluation of Modes in Dielectric Resonators Using a Surface Integral Equation Formulation". In: *Microwave Theory and Techniques, IEEE Transactions on* 31.12 (1983), pp. 1023–1029.

[29] P. Yla-Oijala, M. Taskinen, and J. Sarvas. "Surface Integral Equation Method for General Composite Metallic and Dielectric Structures with Junctions". In: *Progress In Electromagnetics Research* 52 (2005), pp. 81–108. DOI: 10.2528/PIER04071301.

[30] B. M. KolundZija. "Electromagnetic Modeling of Composite Metallic and Dielectric Structures". In: *Microwave Theory and Techniques, IEEE Transactions on* 47.7 (1999), pp. 1021–1032. ISSN: 0018-9480. DOI: 10.1109/22.775434.

[31] J. Shin, A. W. Glisson, and A. A. Kishk. "Modeling of General Surface Junctions of Composite Objects in an SIE/MoM Formulation". In: *ACES Conference Proceedings*. 2000, pp. 683–690.

[32] J. R. Mautz and R. F. Harrington. "Electromagnetic scattering from a homogeneous material body of revolution". In: *Arch. Elektron. Uebertragungstechn. (Electron. Commun.)* 33 (1979), 7180.

[33] R.J. Mailloux. *Phased Array Antenna Handbook*. Artech House, Incorporated, 2005.

[34] G. Valerio, P. Baccarelli, P. Burghignoli, and A. Galli. "Comparative Analysis of Acceleration Techniques for 2-D and 3-D Green's Functions in Periodic Structures Along One and Two Directions". In: *Antennas and Propagation, IEEE Transactions on* 55.6 (2007), pp. 1630–1643. ISSN: 0018-926X.

[35] E. Garcia, C. Delgado, L. Lozano, and F. Catedra. "Analysis of the Parameters of an Approach that Combines the Characteristic Basis Function Method and the Multilevel Fast Multipole". In: *Microwaves, Antennas Propagation, IET* 5.4 (2011), pp. 419–425. ISSN: 1751-8725. DOI: 10.1049/iet-map.2010.0404.

[36] N. Kinayman and M.I. Aksun. "Comparative Study of Acceleration Techniques for Integrals and Series in Electromagnetic Problems". In: *Antennas and Propagation Society International Symposium, 1995. AP-S. Digest*. Vol. 2. 1995, 1037–1040 vol.2. DOI: 10.1109/APS.1995.530194.

[37] C. Craeye, X. Radu, F. Capolino, and A. Schuchinsky. "Fundamentals of Method of Moments for Artificial Materials". In: *Handbook of Artificial Materials, vol I: Phenomena and Theory*. Taylor and Francis, 2009.

[38] S. Singh, W.F. Richards, J.R. Zinecker, and D.R. Wilton. "Accelerating the Convergence of Series Representing the Free Space Periodic Green's Function". In: *Antennas and Propagation, IEEE Transactions on* 38.12 (1990), pp. 1958–1962. ISSN: 0018-926X. DOI: 10.1109/8.60985.

[39] F. Capolino, D.R. Wilton, and W.A. Johnson. "Efficient Computation of the 2-D Green's Function for 1-D Periodic Structures Using the Ewald Method". In: *Antennas and Propagation, IEEE Transactions on* 53.9 (2005), pp. 2977–2984. ISSN: 0018-926X.

[40] F. Capolino, D.R. Wilton, and W.A. Johnson. "Efficient Computation of the 3D Green's Function with One Dimensional Periodicity using the Ewald Method". In: *Antennas and Propagation Society International Symposium 2006, IEEE*. 2006, pp. 2847–2850. DOI: 10.1109/APS.2006.1711199.

[41] S. Oroskar, D. R. Jackson, and D. R. Wilton. "Efficient Computation of the 2D Periodic Green's Function Using the Ewald Method". In: *J. Comput. Phys.* 219.2 (2006), pp. 899–911. DOI: 10.1016/j.jcp.2006.06.050.

[42] G. Kobidze, B. Shanker, and D. P. Nyquist. "Efficient Integral-Equation-based Method for Accurate Analysis of Scattering from Periodically Arranged Nanostructures". In: *Physical Review E* 72 (5 2005), p. 056702. DOI: 10.1103/PhysRevE.72.056702.

[43] B. Stroustrup. *The C++ Programming Language*. 3rd. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000. ISBN: 0201700735.

[44] S. Balay, J. Brown, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H." Zhang. *PETSc Users Manual*. Tech. rep. ANL-95/11 - Revision 3.4. Argonne National Laboratory, 2013.

[45] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries". In: *Modern Software Tools in Scientific Computing*. Ed. by E. Arge, A. M. Bruaset, and H. P. Langtangen. Birkhäuser Press, 1997, pp. 163–202.

[46] C. Sanderson. *Armadillo: An Open Source C++ Linear Algebra Library for Fast Prototyping and Computationally Intensive Experiments*. Tech. rep. NICTA, 2010.

[47] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide*. 1992.

[48] *LAPACK*. URL: http://www.netlib.org/lapack.

[49] C. Lawson, R. Hanson, D. Kincaid, and F. Krogh. "Basic Linear Algebra Subprograms for FORTRAN usage". In: *Transactions on Mathematical Software* 5 (1979), 308323.

[50] *OpenBLAS*. URL: http://xianyi.github.com/OpenBLAS.

[51] *MPIch2*. URL: http://www.mpich.org.

[52] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing In Science & Engineering* 9.3 (2007), pp. 90–95.

[53] P. Ramachandran and G. Varoquaux. "Mayavi: 3D Visualization of Scientific Data". In: *Computing in Science & Engineering* 13.2 (2011), pp. 40–51. ISSN: 1521-9615.

[54] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Baltimore, MD: Johns Hopkins University Press, 1983.

[55] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, Second Edition*. Philadelphia, PA: SIAM, 1994.

[56] *SCALAPACK*. URL: http://www.netlib.org/scalapack/.

[57] *BLACS*. URL: http://www.netlib.org/blacs/.

[58] *PLAPACK*. URL: http://www.cs.utexas.edu/users/plapack/.

[59] C. Geuzaine and J.-F Remacle. "GMSH: a three-dimensional Finite Element Mesh Generator with built-in pre- and post- processing facilities." In: *International Journal for Numerical Methods in Engineering* 79.11 (2009), pp. 1309–1331.

[60] Y. Kamen and L. Shirman. "Triangle Rendering using Adaptive Subdivision". In: *IEEE Computer Graphics and Applications* (1998), pp. 95–103.

[61] Z. Chen and H. Xiao. "Preconditioned Krylov Subspace Methods Solving Dense Nonsymmetric Linear Systems Arising from BEM". In: *Computational Science ICCS 2007*. Ed. by Yong Shi, GeertDick Albada, Jack Dongarra, and PeterM.A. Sloot. Vol. 4489. Springer Berlin Heidelberg, 2007, pp. 113–116. DOI: 10.1007/978-3-540-72588-6_16.

[62] C. Craeye and R. Sarkis. "Finite Array Analysis through Combination of Macro Basis Functions and Array Scanning Methods". In: *Applied Computational Electromagnetics Society Journal* 23 (2008).

[63] S. Amaya Maldonado. "Efficient Analysis of 2D Antenna Arrays using the ASM-MBF Method". MA thesis. Universitat Politcnica de Catalunya, 2011.

Publications

[P1] Elson Agastra, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "Multi-Objective Optimization Techniques". In: *Wiley Enciclopedia of Electrical and Electronics Engineering*. John Wiley & Sons, Inc., 2014. DOI: 10.1002/047134608X.

[P2] Elson Agastra, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "Taguchi's Method for Multi-Objective Optimization Problems". In: *International Journal of RF and Microwave Computer-Aided Engineering* 23.3 (2013), pp. 357–366. ISSN: 1099-047X. DOI: 10.1002/mmce.20680.

[P3] Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "A Novel Multiobjective Taguchi's Optimization Technique for Multibeam Array Synthesis". In: *Microwave and Optical Technology Letters* 55.8 (2013), pp. 1836–1840. ISSN: 1098-2760. DOI: 10.1002/mop.27705.

[P4] Ruggero Taddei, Giacomo Guarnieri, Giuseppe Mauriello, Giuseppe Pelosi, and Stefano Selleri. "A fast MoM Code for Finite Arrays". European Microwave Conference, EuMW, Rome. 2014.

[P5] Giacomo Guarnieri, Ruggero Taddei, and Giuseppe Mauriello. "Slot Analysis Techniques for the Synthesis of Waveguide Slot Array Antennas". Submitted to IET Microwaves, Antennas and Propagation. 2014.

[P6] Ruggero Taddei, Giacomo Guarnieri, and Giuseppe Mauriello. "Automated Waveguide Array - Integrated Design Environment (AWA-IDE): An Automated and Modular Tool for Planar Waveguide Array Synthesis and Analysis". In: *Antennas and Propagation Magazine, IEEE* 55.4 (2013), pp. 204–216. DOI: 10.1109/MAP.2013.6645180.

[P7] Ruggero Taddei and Giacomo Guarnieri. "Automated Waveguide Array Design with Ansys HFSS 15 IronPython scripting". In: *ANSYS User Meeting/High Frequency Simulation Conference*. Salsomaggiore Terme, Italy. 2013.

[P8] Ugo d'Elia, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "Finite Element Design of CNT-based Multilayer Absorbers". In: *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering* 32.6 (2013), pp. 1929 –1942. DOI: 10.1108/COMPEL-09-2012-0187.

[P9] Niccoló Breda, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "Finite Element Analysis of Wideband Nanostructures for Photovoltaic Applications". In: *11th International Workshop on Finite Elements for Microwave Engineering-FEM2012*. Estes Park, Colorado, USA. 2012.

[P10] Ugo d'Elia, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "A Carbon Nanotube Based Frequency-Selective Absorber". In: *International Journal of Microwave and Wireless Technologies, Cambridge University Press* 2.5 (2010), pp. 479–485. DOI: 10.1017/S1759078710000693.

[P11] Ugo d'Elia, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "A Carbon Nanotubes-Based Material for High Absorption FSS Layers". In: *10th International Workshop on Finite Elements for Microwave Engineering-FEM2010*. Meredith (NH). 2010.

[P12] Ugo d'Elia, Giuseppe Pelosi, Stefano Selleri, and Ruggero Taddei. "Finite Element Design of CNT-based Multilayer Absorbers". In: *IV Italian Workshop The Finite Element Method Applied to Electrical and Information Engineering*. Roma Tre, Rome. 2010.