electronics

MDPI

Article

Curved Domains in Magnetics: A Virtual Element Method Approach for the T.E.A.M. 25 Benchmark Problem

Franco Dassi [1,*], Paolo Di Barba [2] and Alessandro Russo [1]

[1] Dipartimento di Matematica e Applicazioni, Università di Milano–Bicocca, Via Cozzi 55, 20153 Milano, Italy; alessandro.russo@unimib.it
[2] Dipartimento di Ingegneria Industriale e dell'Informazione, Università di Pavia, Via Ferrata 5, 27100 Pavia, Italy; paolo.dibarba@unipv.it
* Correspondence: franco.dassi@unimib.it

Abstract: In this paper, we are interested in solving optimal shape design problems. A critical challenge within this framework is generating the mesh of the computational domain *at each optimisation step* according to the information provided by the minimising functional. To enhance efficiency, we propose a strategy based on the Finite Element Method (FEM) and the Virtual Element Method (VEM). Specifically, we exploit the flexibility of the VEM in dealing with generally shaped polygons, including those with hanging nodes, to update the mesh *solely* in regions where the shape varies. In the remaining parts of the domain, we employ the FEM, known for its robustness and applicability in such scenarios. We numerically validate the proposed approach on the T.E.A.M. 25 benchmark problem and compare the results obtained with this procedure with those proposed in the literature based solely on the FEM. Moreover, since the T.E.A.M. 25 benchmark problem is also characterised by curved shapes, we utilise the VEM to accurately incorporate these "exact" curves into the discrete solution itself.

Keywords: virtual elements; curved meshes; optimisation procedure; magnetic field; T.E.A.M. 25

Citation: Dassi, F.; Di Barba, P.; Russo, A. Curved Domains in Magnetics: A Virtual Element Method Approach for the T.E.A.M. 25 Benchmark Problem. *Electronics* **2024**, *13*, 2053. https://doi.org/10.3390/electronics13112053

Academic Editor: Jianguo Zhu

Received: 28 March 2024
Revised: 17 May 2024
Accepted: 22 May 2024
Published: 24 May 2024

1. Introduction

Shape optimisation is a branch of optimal control theory. One of the typical problems is to find the shape that is optimal, i.e., the shape that minimises a prescribed cost functional under specific constraints. Such a functional is usually computed from the discrete solution of a Partial Differential Equation (PDE) defined on a *variable* domain that is often obtained via a Finite Element Method (FEM).

In computational electromagnetism, there is a wide variety of shape optimisation problems based on the resolution of a PDE. For instance, one would like to find the shape of a magnetic pole in order to have a prescribed magnetic field in a specific region of interest. One of the most famous benchmark optimisation problems of this kind is the so-called T.E.A.M. 25 problem [1]. In this case, the device is characterised by a two-pole electromagnet used for the magnetisation of the ferromagnetic powder inserted in the two cavities and the optimisation consists of shaping the die molds so that the ferromagnetic powder is subject to a radial magnetisation in a given region.

From this example, it becomes clear that one of the main issues related to these shape optimisation processes is handling domain variations. Indeed, since the cost functional depends on the solution of a PDE, each time that the domain changes, it is necessary to compute a new discretisation and solve the PDE again. This fact can be a bottleneck for the entire optimisation procedure. In fact, although mesh generation software is able to discretise involved domains without high computational effort, one would prefer to avoid the re-meshing procedure and simply modify the actual mesh. To achieve this goal, there are several strategies that we will briefly describe in the following paragraphs.

Given a mesh of the domain, one possible way to avoid re-meshing is to slightly move mesh nodes according to the information provided by the cost functional. This procedure is fast and avoids re-meshing but it could produce discretisations characterised by zero-area triangles or inverted elements.

Another strategy consists of keeping the mesh fixed and changing the "properties" of the FEM elements [2–6]. In the electromagnetism framework, the "property" can be the presence or absence of the ferromagnetic material. For instance, in [3], the authors apply such a strategy for the T.E.A.M. 25 problem. They discretise the region with a binary bitmap where the die molds are placed; see Figure 6 of [3]. In particular, black squares (pixel 1) represent a portion of the domain filled with ferromagnetic material, while white squares (pixel 0) represent air. Furthermore, the constraints on the geometry are converted to colour constraints, i.e., there are pixels that cannot have a value equal to 1. Such a strategy clearly does not require any mesh generation during the optimisation process; different shapes will be directly discretised during the assembly of the FEM matrix. However, although this strategy avoids generating a new mesh at each optimisation step, the result of the shape optimisation is characterised by an "echelon structure" that is not so reliable from a technological point of view to construct a prototype device.

Another way to drastically reduce the computational effort in mesh generation during an optimisation process is the one proposed in [7]. The idea behind this method is to combine the FEM with the Virtual Element Method (VEM) [8–13]. The VEM is an extension of the FEM to meshes composed of generally shaped polygons. These two methods share the same degrees of freedom on edges, so they can coexist on the same mesh. One can take advantage of this fact in te shape optimisation process. Consider, for instance, a uniform quadrilateral mesh in the region where the shape is varying. Here, the unknown optimal shape is the piecewise straight line; see Figure 1 (left). Then, one can generate a mesh conforming to this line by cutting the elements, as shown in Figure 1 (right).

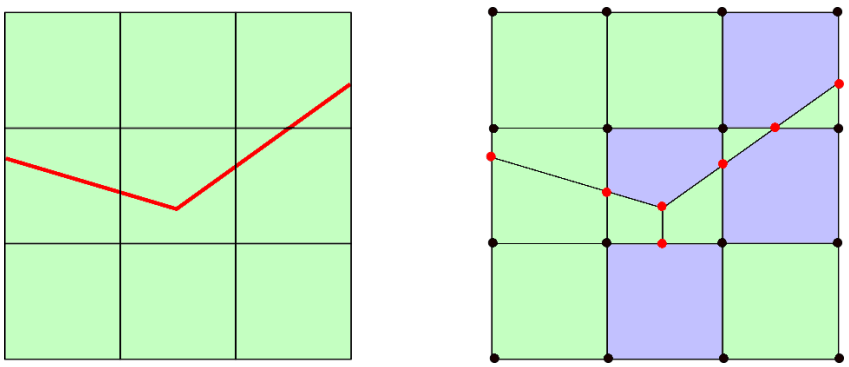

Figure 1. Example of cutting squares where we highlight the elements for which we use the FEM (light green), and the ones for which we use the VEM (light blue). Note that a VEM element can be fully surrounded by FEM elements. The initial mesh with the cutting line in red (**left**) the resulting cut mesh (**right**).

As a consequence, the resulting mesh will be composed of squares, triangles, and other polygons with more than four edges. A standard FEM approach cannot handle such a mesh but, if we combine the FEM with the VEM, we can compute the discrete solution. More specifically, referring to Figure 1 (right), one can use the FEM on triangles and squares (light green elements) and the VEM on all the other polygons that have more than four edges (light blue elements). In general, one could use the VEM for *all* the elements but it is advisable to combine the FEM and the VEM since for the VEM, the computation of local stiffness matrices for triangles and quadrilaterals would be more expensive with respect to the FEM.

The last cutting procedure is clearly faster than generating a new mesh at each iteration step, and the computational effort required to obtain the discrete solution also benefits from it. In [7], we have used such a strategy and we noted that the number of mesh elements

remains relatively stable throughout the optimisation steps; see Figure 10c of [7]. As a consequence, the size of the linear system to be solved to obtain the discrete solution is approximately constant during the entire optimisation process.

In this paper, we use the last approach and we apply it to the T.E.A.M. 25 benchmark problem. More specifically, the paper is organised as follows. In Section 2, we give a short survey of the VEM. Then, in Section 3, we give a brief description of the procedure used to minimise the cost functional. Finally, in Section 4, we show the entire optimisation process applied to the T.E.A.M. 25 benchmark problem and we compare our results with the ones presented in the literature.

2. The Virtual Element Method: A Short Survey

The Potential formulation for Magnetostatics consists of a single Poisson equation with variable diffusion. For simplicity, we describe the VEM in the case of a Poisson equation with constant diffusion. The construction can be easily extended to a piecewise constant diffusion, variable coefficients, or to the case of a non-linear diffusion, i.e., a diffusion coefficient that depends on the potential field itself.

From a mathematical point of view, solving a Poisson equation with constant diffusion and homogeneous Dirichlet boundary condition amounts to finding the solution of the following problem:

$$\begin{cases} -\Delta u = f & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{1}$$

which can be written in variational form as follows:

$$\begin{cases} \text{find } u \in H_0^1(\Omega) \text{ such that} \\ \displaystyle\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\mathbf{x} = \int_\Omega f v \, \mathrm{d}\mathbf{x} \quad \text{for all } v \in H_0^1(\Omega). \end{cases} \tag{2}$$

The VEM is an extension of the FEM to generally shaped polygons and maintains the structure of a Galërkin method. Hence, to compute the discrete solution, we follow exactly the same steps of the FEM. Specifically, given an integer k, which defines the polynomial accuracy of the method, we proceed as follow:

`step 1:` discretise the domain Ω with non-overlapping elements $\{E\}$;

`step 2:` construct in each element a local space $V_h^k(E)$;

`step 3:` assemble the global space $V_h^k(\Omega) \subseteq H_0^1(\Omega)$ by gluing together in a continuous fashion the local spaces, respecting global boundary conditions;

`step 4:` solve the discrete variational problem associated with the equation at hand.

The key point of the VEM stays in the definition of the discrete local spaces $V_h^k(E)$. To better understand such a construction, we start with $k = 1$ and compare the VEM and the FEM spaces.

Given a triangle T, the local FEM space of degree one on T is the well-known Courant element, which corresponds to the space of polynomials of degree 1. Such a polynomial can be uniquely defined by the values at the vertices of the triangle so the *local degrees of freedom* of a function $v_h \in V_h^1(T)$ are

$$\mathrm{dof}_i(v_h) := v_h(\boldsymbol{V}_i), \tag{3}$$

where $\boldsymbol{V}_i$ is the i-th triangle vertex. The local basis functions $\varphi_i \in V_h^1(T)$ are defined in terms of the local degrees of freedom by the following condition:

$$\mathrm{dof}_i(\varphi_j) = \delta_{ij}. \tag{4}$$

The previous equation simply means that φ_i is the only linear function that has value one on vertex $\boldsymbol{V}_i$ and zero on the others. Then, the set of functions $\{\varphi_1, \varphi_2, \varphi_3\}$ is a basis for the space $V_h^1(T)$.

Let us now characterise the space $V_h^1(T)$ in a different way that will be crucial to understand the VEM local spaces for $k = 1$. Since there exists a unique harmonic function with assigned value at the boundary, the local space $V_h^1(T)$ can also be identified by the following properties:

- for each edge e of T, $v_{h|e}$ is linear on e;
- the linear functions $v_{h|e}$ on the edges match at the vertices;
- v_h is harmonic inside, i.e., $\Delta v_h = 0$ in T.

The idea of the VEM is to use this characterisation in order to define the local spaces on a *generally* shaped polygon (including the case of hanging nodes). Specifically, given a polygon P, we define the VEM space $V_h^1(P)$ in the following way:

$$V_h^1(P) := \left\{ v_h \in H^1(P) \,:\, \Delta v_h = 0,\ v_h|_e \in \mathbb{P}_1(e)\ \forall e \in \partial P,\ v_h|_{\partial P} \text{ continuous} \right\}.$$

From this definition, it is clear that, if P is a triangle, we recover *exactly* the Courant element. The i-th local degree of freedom is still defined as the value of the function at vertex i. Then, a function in $V_h^1(P)$ is completely determined by its values at the vertices, so that

$$\text{dimension of } V_h^1(P) = \#\text{ of vertices of } P := N^V.$$

Also, in this case, the local basis function φ_i takes the value one at vertex i and is zero at the other vertices.

The global VEM space is defined exactly as for the classical FEM; the local spaces on adjacent polygons sharing an edge are glued together continuously through the common edge, producing globally continuous approximation functions. Moreover, it is clear that this gluing process can also be performed between FEM and VEM local spaces since, for both of them, on the common edge we have linear polynomials which share the same degrees of freedom.

It is worth noting that $V_h^1(P)$ has a key property; it contains linear polynomials. Indeed, if $p_1 \in \mathbb{P}_1(P)$, p_1 is linear on each edge, is continuous on the boundary of the polygon, and is harmonic, i.e., $\Delta p_1 \equiv 0$. This is an important observation that ensures the same good approximation properties of the classical FEM; if we were able to compute exactly the local stiffness matrices, the method would converge at the expected rates.

However, since the basis functions are not explicitly known inside the polygon (we only know that they are harmonic), we cannot directly use them to compute such a matrix. To avoid this issue, in principle, one might compute such basis functions by solving in each element a Laplace equation with given boundary data, but the computational cost would be huge.

Since it is not feasible to compute the local stiffness matrices directly with the virtual basis functions φ_i as in FEM, the idea of the VEM is to use a linear polynomial $\Pi_1^\nabla \varphi_i$ that approximates φ_i in the following integral sense:

$$\int_P \nabla(\Pi_1^\nabla \varphi_i - \varphi_i) \cdot \nabla q_1 \, \mathrm{d}\mathbf{x} = 0 \quad \text{for all linear polynomial } q_1, \tag{5}$$

$$\sum_{j=1}^{N^V} \Pi_1^\nabla \varphi_i(V_j) = \sum_{j=1}^{N^V} \varphi_i(V_j). \tag{6}$$

Even if we do not know the virtual basis function φ_i inside P, it is possible to compute $\Pi_1^\nabla \varphi_i$ using only the degrees of freedom of φ_i (see, e.g., [8,14]).

At this point, it seems to be a good idea to use the following approximation of the "true" local stiffness matrix:

$$\int_P \nabla \varphi_j \cdot \nabla \varphi_i \, \mathrm{d}\mathbf{x} \approx \int_P \nabla \Pi_1^\nabla \varphi_j \cdot \nabla \Pi_1^\nabla \varphi_i \, \mathrm{d}\mathbf{x}. \tag{7}$$

However, using Equation (7) is not appropriate because the $N^V \times N^V$ local stiffness matrix

$$M_{ij}^{\text{loc}} = \int_P \nabla \Pi_1^\nabla \varphi_j \cdot \nabla \Pi_1^\nabla \varphi_i \, \mathrm{d}\mathbf{x},$$

is rank-deficient, and it could give rise to a singular global matrix.

The right rank for the local matrix M_{ij}^{loc} should be $N^V - 1$, because the constant functions are clearly in the kernel, but they are ruled out by the global boundary conditions, giving at the end an invertible matrix.

Instead, it can be easily shown that the rank of M_{ij}^{loc} is exactly 2 which is strictly less than $N^V - 1$, unless P is a triangle. In the global space, the constant functions are ruled out by the boundary condition, but the global stiffness matrix in general remains singular.

To fix the rank of M_{ij}^{loc}, we need an additional term that:

- guarantees existence and uniqueness;
- does not spoil consistency;
- is defined element by element and computable.

It turns out that we can add to M_{ij}^{loc} a term of the form

$$S\big((\mathrm{I} - \Pi_1^\nabla)u_h, (\mathrm{I} - \Pi_1^\nabla)v_h\big) = \sum_T S_T\big((\mathrm{I} - \Pi_1^\nabla)u_h, (\mathrm{I} - \Pi_1^\nabla)v_h\big), \tag{8}$$

where $S_T(\cdot,\cdot)$ is a symmetric coercive bilinear form that scales in the right way. Note that, if one of the entries is a linear polynomial p_1, the term (8) is *exactly* zero so consistency with polynomials is preserved. There is some freedom in the choice of such local stability bilinear form; one possibility is the so-called D-recipe, or the more common `dofi-dofi` [8].

For the treatment of the right-hand side, we need again a suitable approximation of the basis functions φ_i's; we refer the reader to [14].

We briefly consider the case $k = 2$. As for the FEM on triangles, we need to add the middle point of each edge in order to have a polynomial of degree 2 on each edge. Then, since the Laplacian of a polynomial of degree 2 is constant, we substitute the condition $\Delta v_h \equiv 0$ with $\Delta v_h = \text{constant}$:

$$V_h^2(P) := \Big\{ v_h \in H^1(P) \,:\, \Delta v_h \in \mathbb{P}_0(E),\ v_h|_e \in \mathbb{P}_2(e)\ \forall e \in \partial P,\ v_h|_{\partial P} \text{ continuous} \Big\}.$$

In this case, the degrees of freedom of a function $v_h \in V_h^2(P)$ are:

- boundary degrees of freedom: as before, the pointwise values at the vertices and at the middle point of the edges;
- internal degrees of freedom: the mean value on P, i.e., $\dfrac{1}{|P|}\displaystyle\int_P v_h \, \mathrm{d}\mathbf{x}$.

Hence, we have $\dim V_h^2(P) = 2N^V + 1$. Note that in this case, if P is a triangle, we *do not* recover the classical finite element of order two but a larger space. It is worth noting that in this case, we also have $\mathbb{P}_2(P) \subseteq V_h^2(P)$, and the projection Π_2^∇ onto polynomials of degree 2 is computable knowing only the degrees of freedom [14].

In the general case, we have

$$V_h^k(P) := \Big\{ v_h \in H^1(P) \,:\, \Delta v_h \in \mathbb{P}_{k-2}(E),\ v_h|_e \in \mathbb{P}_k(e)\ \forall e \in \partial P,\ v_h|_{\partial P} \text{ continuous} \Big\}.$$

This space still contains polynomials of degree k. The boundary degrees of freedom are the point-wise values at the vertices and at $k-1$ internal nodes on each edge, which can be taken uniformly spaced as in the FEM, and the internal degrees of freedom are moments up to order $k-2$ in P, i.e.,

$$\frac{1}{|P|}\int_P v_h m \, \mathrm{d}\mathbf{x}, \quad m \text{ monomial up to degree } k-2. \tag{9}$$

In [14], it is shown that in this case, the projection Π_k^∇ onto polynomials of degree k is also computable.

After this brief introduction, it is clear that the VEM has two main drawbacks. First, in the computation of the local matrices used to assemble the global one, it is necessary to build a suitable polynomial projection of the virtual basis functions. Such a projection is obtained by solving small linear systems for each mesh element. As a consequence, the computational effort is slightly increased with respect the FEM where there is a reference element. The last drawback is the presence of a stabilisation term that is necessary to guarantee existence and uniqueness of the solution of the final linear system [8].

Despite these two drawbacks, there are several advantages in using the VEM that give more flexibility in the approximation of a PDE.

In `step 1`, there is more flexibility in the mesh generation. Indeed, in an FEM framework, the "non-overlapping elements" are only triangles, squares, tetrahedrons, and hexahedrons, while in the VEM, such elements can be a generally shaped polygon (also non-convex); see Figure 2. It is worth noting that in Figure 2, there is also an element with hanging nodes; the VEM can naturally handle such an element and, in the specific case depicted in Figure 2, it is treated as a "normal" pentagon, despite the fact that two edges are collinear.

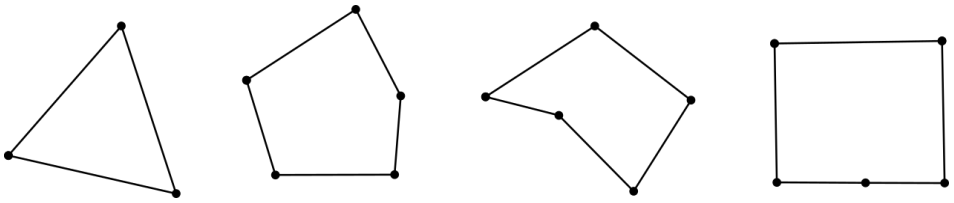

Figure 2. A sequence of elements that can be handled by the VEM.

Furthermore, the usage of polygons can be exploited to reduce the computational cost. Specifically, there will be less degrees of freedom with respect to an FEM approach. In Figure 3, we show the degrees of freedom associated with a pentagonal region for $k = 2$, meshed by only one virtual element and the same region that needs at least three (triangular) finite elements.

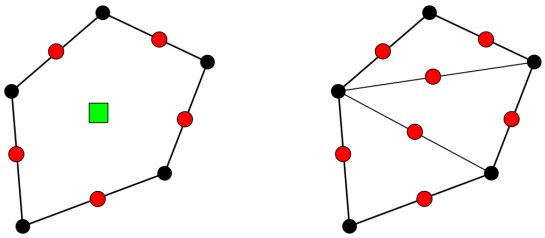

Figure 3. Virtual element (**left**) and finite elements (**right**) for a pentagonal region, degree 2. In both figures we highlight the degrees of freedom. Specifically, back and red circles are points evaluations, while the green square is a moment.

In `step 2`, there is much freedom in the definition of the local spaces. Consequently, it is possible to plug in useful properties which can be essential in the physics of the global solution. For instance, in [15], the authors propose a VE space where the discrete velocity vector field of a Stokes or Navier–Stokes problem is solenoidal. Such a property is not so easy to guarantee with the FEM and it is usually achieved only in a weak sense. One possibility to obtain a pointwise divergence-free velocity field is using the Scott–Vogelius elements, but such a choice will increase the computational effort in obtaining the discrete solution. Furthermore, it is also possible to incorporate the curved geometry of the computational domain inside the VEM space [16]. Such an addition can be made by using the *exact* parameterisation of the curve. For instance, one can incorporate the description of a circle and plug the geometry exactly in the VEM space.

Then, as already mentioned in the introduction, the FEM and the VEM can coexist in the same mesh. Indeed, in `step 2`, it is possible to construct FEM spaces on triangles and squares, employing VEM spaces on the other elements; see Figure 1. Then, since the local virtual element spaces are polynomials on the element's boundary, they can be glued with continuity with the FEM local spaces in `step 3`. The final global space will result in a "patchwork" of finite and virtual element spaces. This property also offers a possible solution of one of the main drawbacks of the VEM. Specifically, since the computational effort of assembling local matrices in the VEM is slightly higher than the FEM due to the computation of the projection operator Π_k^∇, one can use VEM spaces *only* where necessary. In this way, one gains in terms of computational effort and exploits the flexibility of the VEM for both element shapes and space properties.

Another advantage is that one can glue two meshes together and solve a PDE on the resulting mesh. Consider, for instance, the mesh depicted in Figure 4. Such a mesh is the result of gluing a structured quadrilateral mesh with a triangular one. It is not possible to proceed with `step 2,3` and `4` with a standard FEM approach. However, if we use the FEM local spaces for triangles and squares, and VEM spaces for the middle elements that have hanging nodes, we can define and obtain the solution of a PDE on such a mesh.

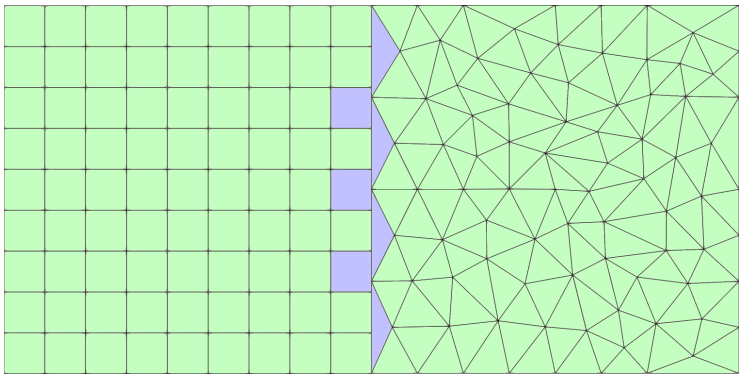

Figure 4. Example of "patchwork" mesh where the VEM spaces are used *only* where it is necessary; we highlight the elements where we use the FEM (light green), and the ones where we use the VEM (light blue). In this specific case, we use the VEM when an element has at least one hanging node.

We conclude this brief introduction on the VEM with Table 1, which will summarise the analogies/differences between the FEM and the VEM. However, if the reader is interested in a more detailed description of the VEM, we refer to [14], where it is possible to find a more practical description of the method for the resolution of a Poisson equation.

Table 1. Summary of the analogies/differences between the FEM and the VEM.

	FEM	**VEM**
shapes	triangles and quadrilaterals	*all* polygons, see Figure 2
local spaces	polynomials	polynomials plus other functions
knowledge	pointwise everywhere	degrees of freedom only

3. Minimising the Cost Functional: The Estra Algorithm

In the literature, there are many algorithms for local or global optimisation:; generally speaking, they can be categorised as belonging to the class of deterministic computing or evolutionary computing or nature-inspired computing. Moreover, a few of them were specifically designed for multi-objective optimisation purposes. In fact, the algorithmic paradigm offered by evolutionary computing is particularly suited for identifying the global minimum of a non-convex cost functional. On the other hand, however, the major drawback of every algorithm of evolutionary computing is the substantial computational

burden which is required. The cost issue is particularly important when the computation of the cost functionals and constraint functions imply the solution of a field analysis problem by means of conforming finite elements or virtual elements, like it was in our case.

Moving from this background, a cost-effective algorithm based on a simple evolution strategy (EStra) was utilised. The key concept is twofold: conceiving the design variables as random numbers characterised by mean values and standard deviations, on the one hand, and considering a solution that could temporarily deteriorate the cost functional, on the other hand.

To this end, the proposed algorithm is implemented in such a way that a new design vector $x = m + ud$ (offspring) is accepted if it improves the current design vector m (parent) according to a user-defined cost functional, subject to problem constraints. This means that a (1+1) strategy is utilised, which originates the lowest computational cost. In turn, d is the standard deviation vector associated to m, while $u \in [-1, 1]$ is a normally distributed perturbation. Vector d is initialised as d_0, and the value of its elements is proportional to the feasible range of the corresponding design variable. Vector d, which drives the search, is updated according to the prescribed rate of success in improving the cost functional; that is where the self-adaptation of the strategy parameter comes in. d itself undergoes a modification, which is ruled by a randomised process. In fact, given the correction rate $q \in (0, 1)$, considering the k-th iteration, $d_{k+1} = q^{-1} d_k$ (or $d_{k+1} = q\, d_k$) is set in order to force a larger (or smaller) standard deviation of Gaussian distribution associated with x in the next iteration, respectively. If the current rate of success is higher than a threshold, the standard deviation is enlarged and vice versa; in other words, the solution vector x and the standard deviation vector d are both subject to random mutation. It is interesting to note that the strategy ensures a broad exploration of the feasible search space, considering also regions where the cost functional could deteriorate, in the search for the global minimum region.

In a basic, cost-effective (1+1) implementation, the operator of selection allows for the best individual, out of parent m and offspring x, to survive to the next generation. In other words, an offspring individual is selected to survive if and only if it is better, or at least not worse, than the parent individual against the cost functional. This way, given an initial point, there is a non-zero probability that the optimisation trajectory eventually leads to a point belonging to the region which incorporates the global minimum. The algorithm converges when the ratio of the largest value of d vector elements to the corresponding element of the initial standard deviation vector d_0 is smaller than a user-prescribed search tolerance.

The basic computational cost c of the EStra algorithm can be a priori estimated as

$$c \approx c_0\, n_i\, n_p\,, \tag{10}$$

where c_0 is the hardware-dependent time necessary to run a single solution of the direct problem associated to the optimisation problem, n_i is the number of convergence iterations for a prescribed search accuracy, and n_p is the number of evolving solutions (in our case, $n_p = 1$). Indeed, in our case, the associated direct problem is the field analysis given a tentative geometry of the magnetiser device.

In principle, there is no upper limitation to the number of design variables the algorithm can process, and this is a potential advantage for a high-dimensionality problem. Summing up, the evolutionary algorithm starts from a guess solution, which can be either user-supplied or randomly generated, iteratively originates a search trajectory driven by the concept of evolution, and eventually converges to an approximation of the global minimum of the cost functional.

In view of a more detailed description of the algorithm, reference is made to the flowchart shown in Figure 5 and subsequent explanations and remarks.

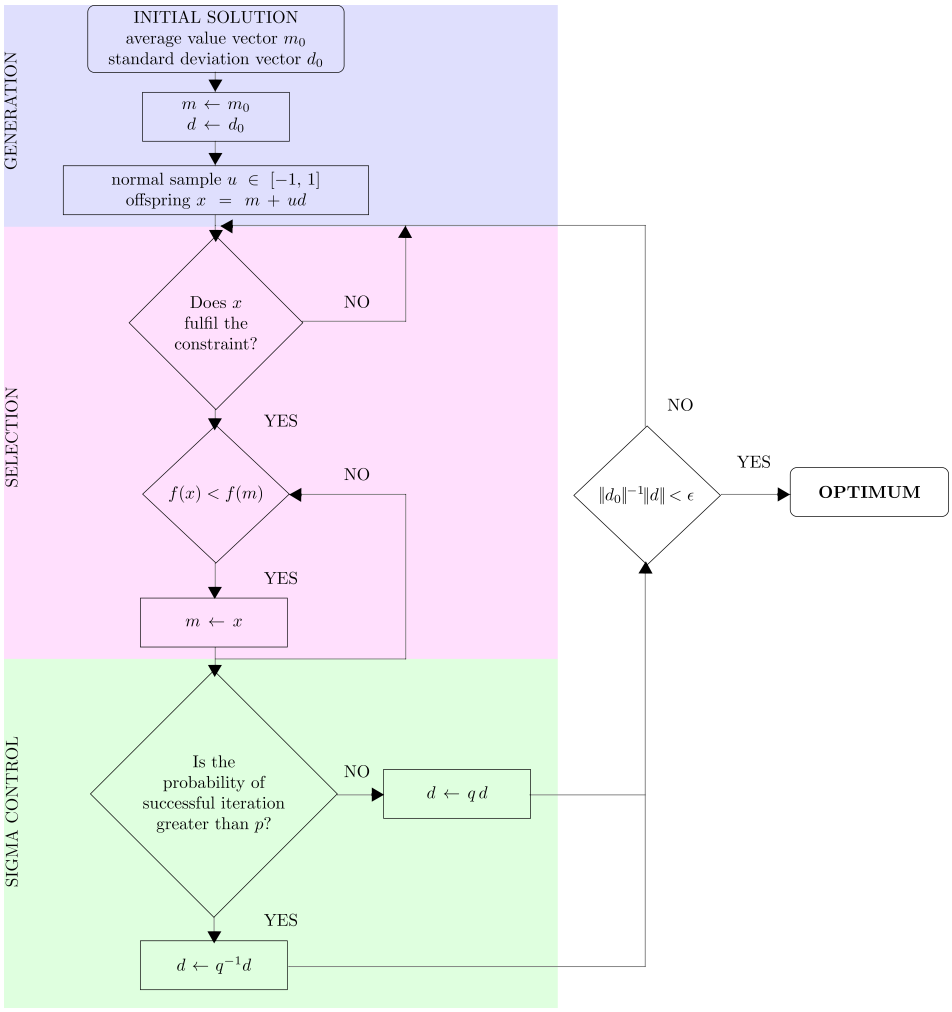

Figure 5. The optimisation algorithm flow chart.

Setting $m = m_0$ and $d = d_0$ at the initial iteration, the generation of a design vector (offspring) $x = m + ud$ takes place, resorting to a scalar stochastic sample u distributed in the $[-1, 1]$ interval; generally, u is a normally distributed sample. Specifically, vector m_0 is assigned by the user and is equal to the initial, or guess, solution, while vector d_0 is computed by the algorithm itself, proportionally to the admissible range of design variables. It is then verified that x fulfils bounds and constraints, i.e., x represents a feasible solution; otherwise, a new design vector is generated until it falls inside the feasible region (generation process). It can be noted that x falls inside the interval $(m - d, m + d)$ according to the following rationale. Small and continuous variations are more likely to be observed than abrupt variations in natural systems subject to an evolution process ruled by a Gaussian-like statistics.

The objective function values $f(m)$ and $f(x)$ are subsequently computed and the test $f(x) < f(m)$ is performed; if the test is successful, then m is replaced by x (selection process); otherwise, m is retained.

The next step is focused on the radius of the search region that will be used for the subsequent iteration. The underlying rationale is that when a point better than the current one is found (i.e., a solution improving the objective function), the radius of the search region is increased around the new point to search for further improvements; if no improvement is found, then the radius of the search region is gradually decreased up to the convergence annealing process. In this respect, it means that the search radius is proportional to the norm of d vector. From this viewpoint, the proposed evolutionary algorithm substantially differs from a deterministic one in which the search region would

be monotonically narrowed around a better point in order to converge towards the nearest minimum. The drawback, in the case of a non-convex function, is that the minimum found might be a local one. In contrast, the evolutionary algorithm, if successful in finding a point better than the current one, spans a larger region of search in order to see if there is a better candidate in the neighbourhood and then does the opposite when this is not possible. This way, there is a concrete chance to find the region where the global minimum is located.

Accordingly, a historical horizon of $n_h > 1$ iterations is defined in order to control the value of standard deviation (the so-called sigma); if at least a fraction p of the last n_h iterations was successful in finding a better point, then the trend is said to be positive, while it is negative otherwise. In particular, an iteration is successful if x is feasible and improves the objective function value with respect to m; therefore, successful iterations occurring within the given historical horizon are counted and their frequency is compared to threshold p. It the trend is positive, then the standard deviation vector is set to $q^{-1}d$; otherwise, it is set to qd. In particular, during the first n_h iterations, when the historical horizon has not yet been formed, d remains unchanged (sigma control).

The procedure stops when the prescribed accuracy ϵ, sometimes called search tolerance, is achieved. Quantities p and q are named probability of success and rate of sigma correction, respectively. They are the "tuning knobs" of the algorithm; typical values, heuristically found, of n_h, p, and q are 30, 0.2, and 0.8, respectively.

4. Test Case: T.E.A.M. 25

We consider as a test case the T.E.A.M. 25 benchmark problem [1] which is a typical example of an optimal shape design problem characterised by curved edges.

In the T.E.A.M. 25 benchmark problem, the device is characterised by a two-pole electromagnet used for the magnetisation of the ferromagnetic powder inserted in the two cavities shown in Figure 6. The cavities are shaped as circular sectors and each cavity is placed between two die molds; one is shaped as a circle, $\mathcal{C}$, and the other ones are arc of ellipse, $\mathcal{E}_1$ and $\mathcal{E}_2$. The optimisation consists of shaping these die molds so that the magnetic field is radial in the region highlighted in Figure 6.

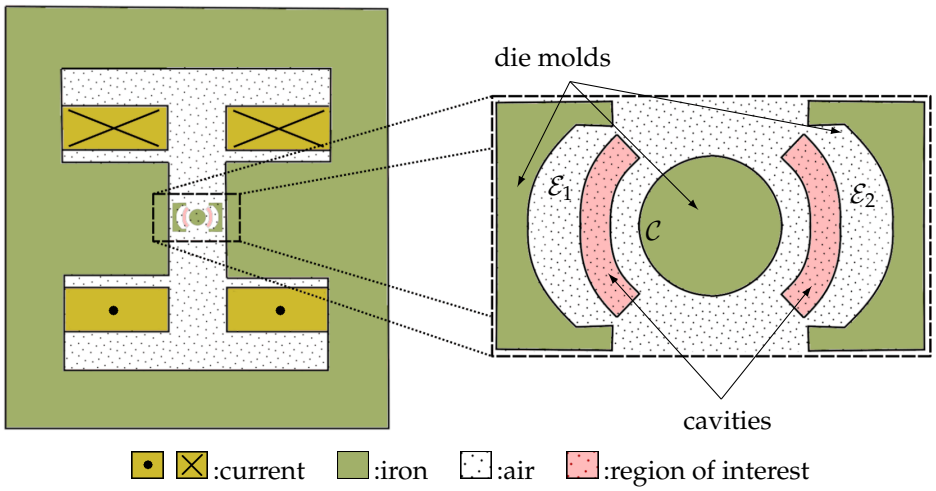

Figure 6. T.E.A.M. 25: the scheme of the electric device/the detail of the die molds.

4.1. Device Modelling

The device of the T.E.A.M. 25 problem is symmetric with respect to the vertical and horizontal axes. Consequently, if we apply Dirichlet and Neumann boundary conditions according to this problem symmetry, it is possible to reduce the computational domain and consider *only* one-quarter of the domain.

The magnetic core and the die molds are made of steel and we will consider a non-linear relation between the induction field **B** and the magnetic field **H**. More specifically,

we consider that the ferromagnetic material is modelled as a *iron–silicon alloy with a width of 0.5* mm [17].

4.2. Design Variables

In this section, we describe the design variables of the T.E.A.M. 25 problem [1]. As already said, the goal of this optimisation problem is to find the circle, $\mathcal{C}$, and the ellipse, $\mathcal{E}$, such that the resulting magnetic field is radial in the region of interest. To achieve this goal, as dictated by the original paper of Takahashi [1], we consider the following design variables:

- $R1$, the radius of $\mathcal{C}$;
- $L2$ and $L3$, the semi-axes of $\mathcal{E}$;
- $L4$, a proper length that determines the position of the point A (see Figure 7). The point A has a fixed y coordinate, 10.5 mm, while its x coordinate is determined by $(20 - L4)$ mm.

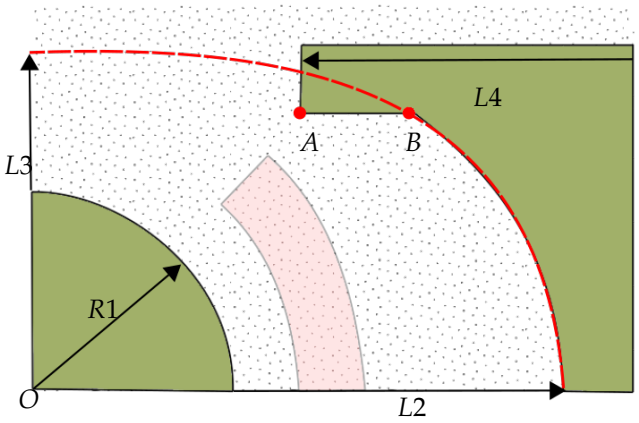

Figure 7. Details of the computational domain where we highlight the design variables $R1$, $L2$, $L3$, and $L4$.

Each of these quantities can vary in a specific range of values; in Table 2, we collect the ranges of each design variable.

Table 2. Range of each design variable.

	R1	***L2***	***L3***	***L4***
min	5.0	12.6	14.0	4.0
max	9.4	18.0	45.0	19.0

Since the algorithm to find the minimum is based on a random walk, the values of the design variables can be associated with invalid configurations. As a consequence, we add further constraints among the variables $L2$, $L3$, and $L4$. To assemble the device of the T.E.A.M. 25 problem, it is necessary that the points $A(x_A, y_A)$ and $B(x_B, y_B)$ do not coincide and that their abscissas verify the relation $x_A < x_B$. Furthermore, we put a constraint on the eccentricity of the resulting ellipse $\mathcal{E}$. More specifically, we require that the eccentricity, e, be lower than 0.75. In this way, the resulting ellipse is far from being a straight line (which corresponds to the case $e \approx 1$).

4.3. Objective Functional

The goal of the optimisation procedure is to identify the design variables $R1$, $L2$, $L3$, and $L4$ so that the magnetic field $\mathbf{B}$ has radial distribution in the region of interest. In the formula, we would like to have the following target field in the region of interest:

$$\mathbf{B}_0 = 1.4\,\mathbf{u}_r + 0.0\,\mathbf{u}_\theta\,, \tag{11}$$

where we have rewritten the vector $\mathbf{B}_0$ in a polar coordinate system, (r, θ), and $\mathbf{u}_r$ and $\mathbf{u}_\theta$ are the unit vectors associated with the r and θ coordinate, respectively.

In order to estimate how far the actual magnetic field is from the target one, as in Equation (11), we define an appropriate objective functional, F; the lower F is, the closer the resulting magnetic field will be to the target one. In the numerical experiments of Section 4.5, we make a different choice of such a functional:

$$\begin{aligned} F_1(R1,\, L2,\, L3,\, L4) &= \max_{P_i \in \Gamma} |\mathbf{B}(P_i) \cdot \mathbf{n}(P_i) - 1.4|\,, \\ F_2(R1,\, L2,\, L3,\, L4) &= \max_{P_i \in \Gamma} \{|\mathbf{B}(P_i) \cdot \mathbf{n}(P_i) - 1.4| + |\mathbf{B}(P_i) \cdot \mathbf{t}(P_i)|\}\,, \\ F_3(R1,\, L2,\, L3,\, L4) &= \sum_{P_i \in \Gamma} \|\mathbf{B}(P_i) - \mathbf{B}_0\|^2\,, \end{aligned} \tag{12}$$

where $\{P_i\}_{i=1}^{100}$ are 100 points placed on a circle arc Γ represented in Figure 8, then $\mathbf{n}(P_i)$ and $\mathbf{t}(P_i)$ are the normal and tangential unit vector of the curve Γ at P_i, respectively.

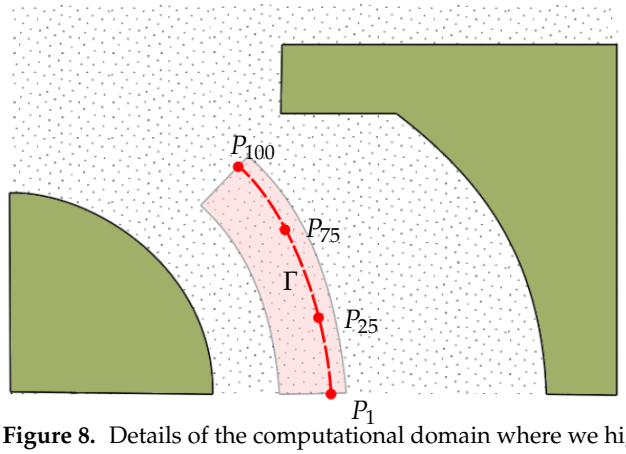

Figure 8. Details of the computational domain where we highlighted the line used to judge different configurations with a dashed red line and we also report the position of the points P_1, P_{25}, P_{75} and P_{100}.

4.4. The VEM Discretisation

In this section, we describe one possible way of using the VEM for the T.E.A.M. 25 optimisation problem. The proposed strategy will exploit the following features of the VEM:

1. its flexibility in incorporating the geometrical information of the domain;
2. the possibility of gluing meshes;
3. the presence of hanging nodes;
4. the coupling between the VEM and the FEM.

As was already mentioned, the VEM is able to incorporate the curved geometry within the functional spaces [16] so both $\mathcal{C}$ and $\mathcal{E}$ are exactly a circle and an ellipse.

Moreover, we take advantage of gluing meshes during the optimisation process following the strategy proposed in [7]. Indeed, we discretise the part of the domain far from the circle and ellipse; see Figure 9 (left). Then, starting from a uniform structured mesh composed by a square, we create the components of the device according to the design variables $R1$, $L2$, $L3$, and $L4$; see Figure 9 (middle). Finally, we glue these two pieces of mesh together and we obtain the computational domain to run the simulation; see Figure 9 (right).

It is worth noting that this procedure will speed up the entire optimisation problem. Indeed, given the values of the design variables $R1$, $L2$, $L3$, and $L4$, we do not need to regenerate the entire mesh, but we have to simply draw on a small mesh the magnet boundaries and merge two meshes, i.e., we jump from the middle of Figure 9 to the right-hand side of Figure 9 during the optimisation process, until we reach a minimum of the

objective functional. Moreover, since such a cutting process is based on a search algorithm, we further speed it up by using a background structured quadrilateral mesh.

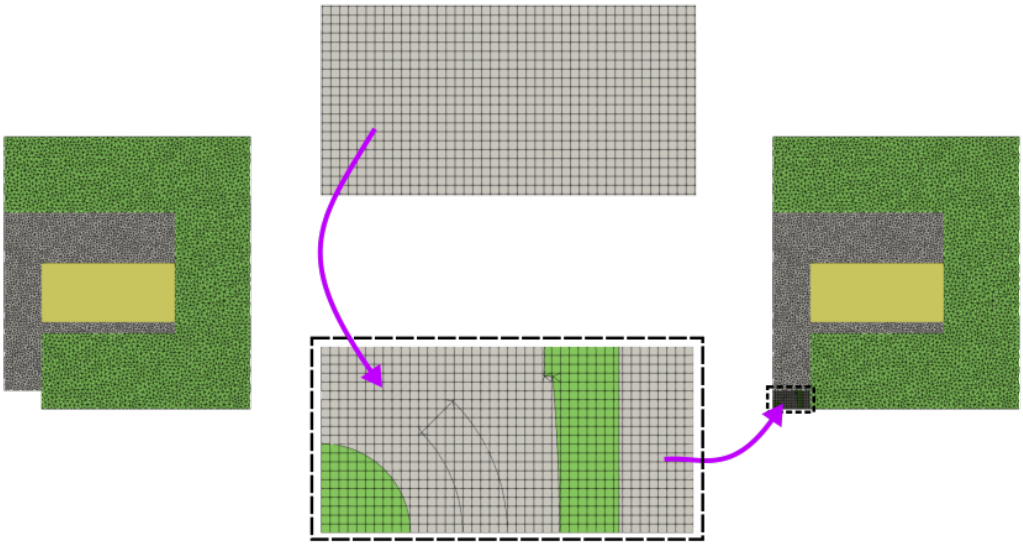

Figure 9. Meshing procedure proposed for the optimisation process. (**Left**) the mesh of the device that is not involved in the "moving" parts. This mesh is computed one time for all. (**Middle**) the structured quadrilateral mesh that is updated at each optimisation step. (**Right**) the final mesh obtained by gluing the middle mesh with the left side mesh.

We further observe that we take advantage of mesh gluing also in the generation of the mesh in Figure 9 (left). Here, the part of the domain which represents the source current is discretised by a rectangle with a lot of hanging nodes on its sides. Indeed, since, in this region, there is the given constant current source, we do not want or, rather, we do not need such a high resolution, i.e., the mesh size in this region can be as large as possible. It is really hard to obtain such a coarse mesh with a standard FEM approach. Indeed, since the FEM triangles are conforming, passing from a region where we have a fine mesh to a region with a coarser one requires a grading region to preserve the conformity; see Figure 10 (left). In the VEM, it is possible to avoid this issue by adding hanging nodes; see Figure 10 (right).

Finally, we further speed up the assembly of the global matrix defined by the PDE at hand using the VEM spaces *only* where it is necessary. Indeed, the resulting mesh is composed of triangles, squares, polygons with hanging nodes, and polygons with curved edges; see Figure 9 (right). For the previous two types of polygon, we use the FEM local spaces, while for the last ones, we use the VEM.

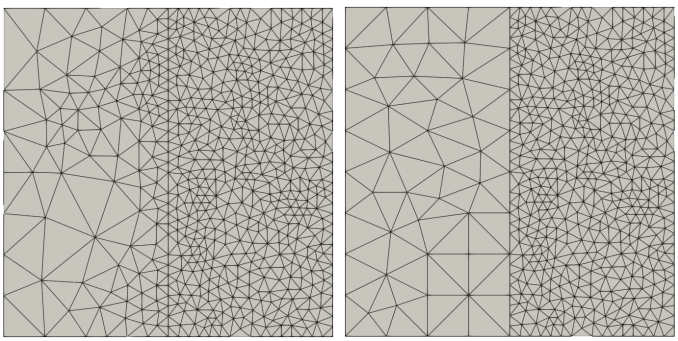

Figure 10. Two regions with different mesh size and a grading between these two regions (**left**). The same regions discretised with the same mesh size constraints but now we allow the presence of hanging nodes so there is the mesh grading effect (**right**).

4.5. Optimisation Results

In this section, we show the results of the optimisation process based on each functional described in Section 4.3 subject to the constraints described in Section 4.2. Both the VEM discretisation and the meshing procedure analysed in Section 4.4 were implemented in the `c++` library `vem++` [18] while, to find the minimum of the objective functional, we use the Matlab© code `runextra` [19].

For each functional in Equation (13), we run the optimisation procedure thirty times. In Figure 11, we collect all the optimisation steps of each run. More specifically, each run is associated with a specific symbol and its x and y coordinates are defined as the mean of discrepancy between the normal and tangential target magnetic field, i.e., given vector field $\mathbf{B}$, we define

$$\begin{aligned} x &= \frac{1}{100} \sum_{P_i \in \Gamma} |\mathbf{B}(P_i) \cdot \mathbf{n}(P_i) - 1.4| \,, \\ y &= \frac{1}{100} \sum_{P_i \in \Gamma} |\mathbf{B}(P_i) \cdot \mathbf{t}(P_i)| \,. \end{aligned}$$

The last step of each run is highlighted with a rectangle that contains the number associated with the run and the colour which represents the functional used during the run itself. Clearly, the closer the last step is to the origin of this Cartesian plane, the closer the magnetic field is to the target one.

In all cases, the number of optimisation steps required to find the minimum was between 30 and 45 iterations. From these data, the best configurations are the couple `run 11` and `run 28` associated with functional F_2 and the couple `run 2` and `run 18` associated with functional F_3. In Figure 12, we present the details of varying regions associated with the best design variables. Another interesting aspect is that the runs associated with F_1 are far away from the origin so they do not give a distribution of the magnetic field that is close to the target one.

Furthermore, if we look at the distribution of the data shown in Figure 11, we observe that these data suggest the presence of three Pareto walls. We highlight such walls with three magenta curves in Figure 13.

In Figure 14, we show the value of the normal and tangential components of $\mathbf{B}$ along the line Γ in order to have a more quantitative analysis on the results of the optimisation problem. In this graph, the best configuration would be the one characterised by tow flat lines at 1.4 T and 0.0 T for the normal and tangential components, respectively. We observe that the graphs associated with `run 11`, `run 28`, `run 2`, and `run 18` are flat from the beginning to almost the end of Γ, i.e., from P_0 to P_{80}. In the last part of the curve (form P_{80} to P_{100}), both the normal and the tangential component of the magnetic field deviate from the target value. The lowest discrepancy is provided by `run 18`. Consequently, the best configuration among *all* runs analysed in this paper is `run 18`.

Finally, to validate the proposed VEM approach, in Table 3, we compare the design variables obtained with this approach and the ones coming from a standard FEM analysis and other optimisation processes [20,21]. From these data, we observe that the best design variables obtained by this procedure are in accordance with the ones in the literature. In this table, we report also the number of iterations performed to reach the minimum.

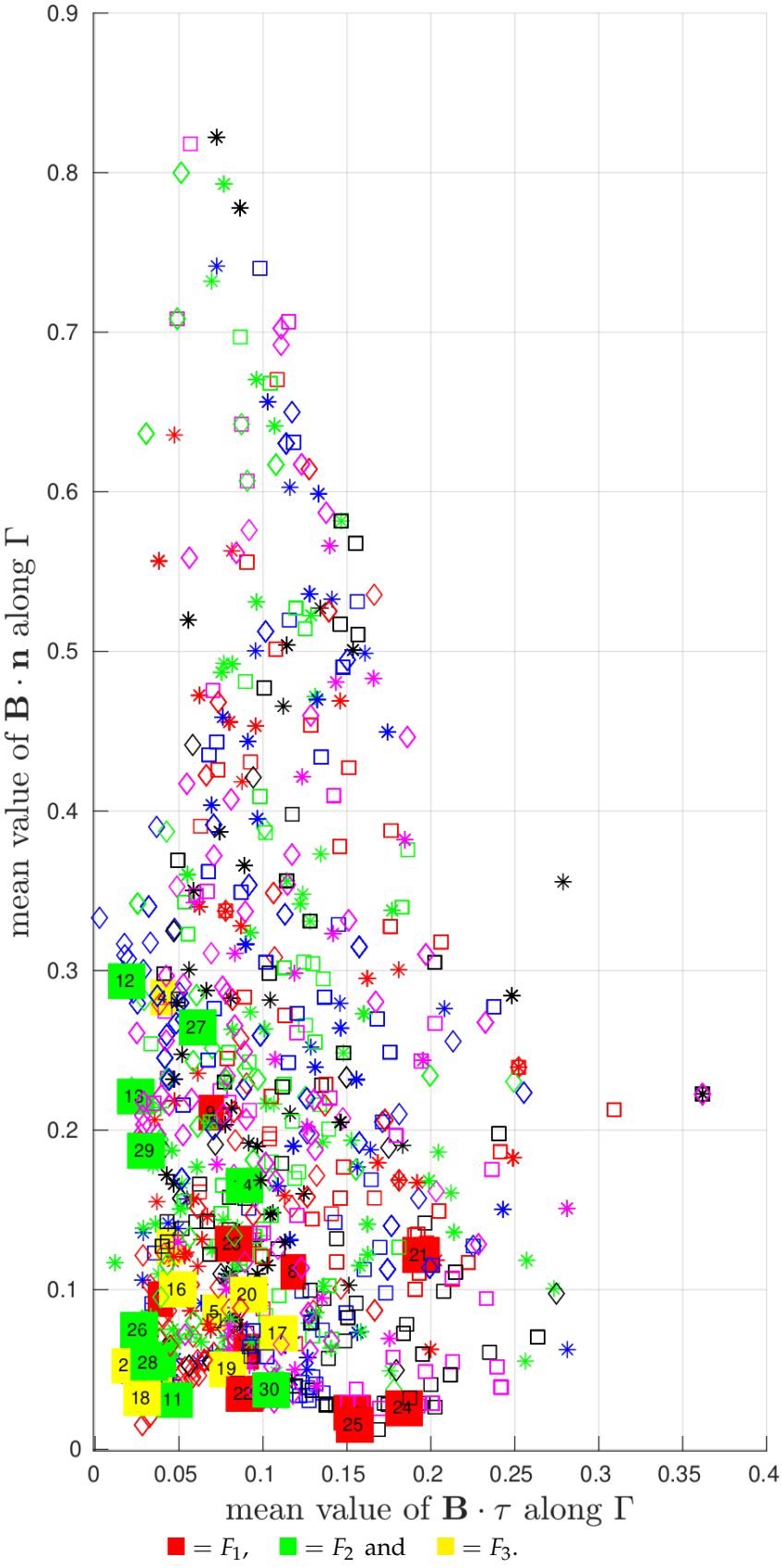

Figure 11. Cartesian plane that collects all the runs carried out for each functional. On the right, a zoom of this graph close to the origin.

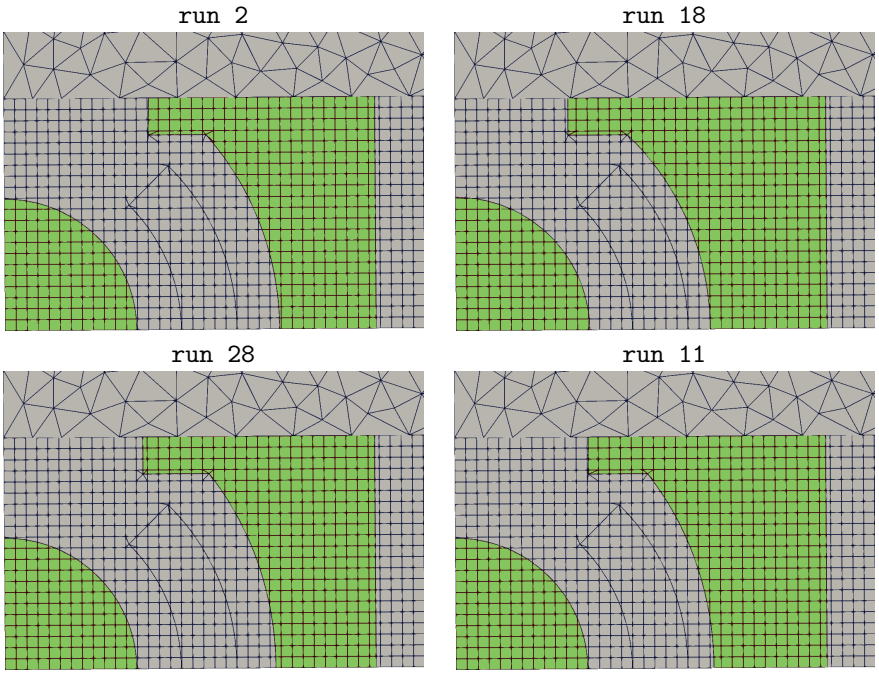

Figure 12. The final configurations associated with a magnetic field that is as radial as possible in the region of interest.

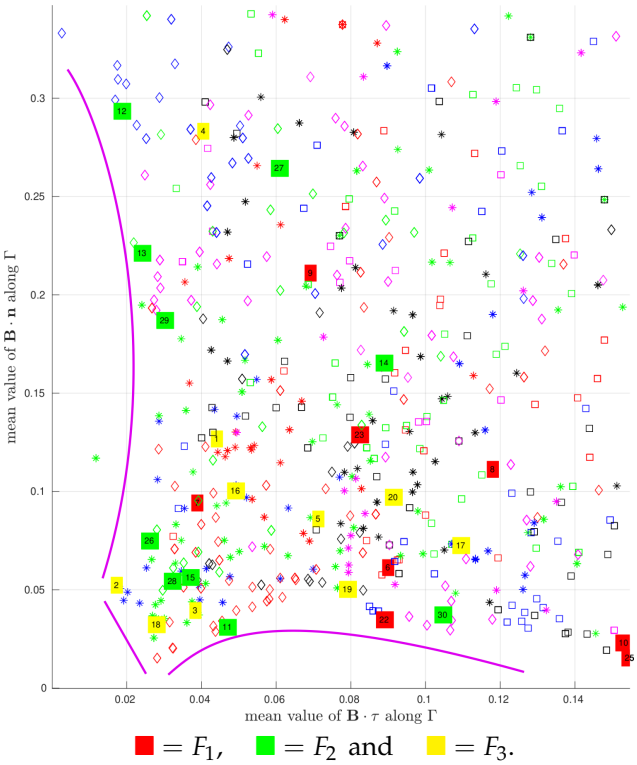

Figure 13. Zoom of this graph in Figure 11 close to the origin.

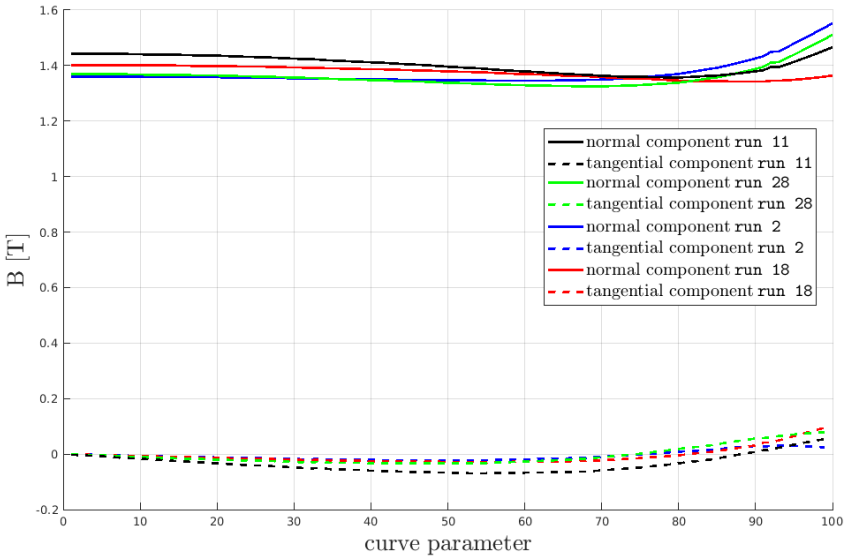

Figure 14. Plot of $\mathbf{B} \cdot \mathbf{n}_\Gamma$ and $\mathbf{B} \cdot \mathbf{t}_\Gamma$ along the line Γ for each run shown in Figure 12.

Table 3. Comparison among design variables obtained from different approaches.

	*R*1	*L*2	*L*3	*L*4	Iterations
`Evolution strategy run 2`	7.1	14.8	15.5	12.2	31
`Evolution strategy run 18`	7.2	13.7	14.3	13.9	40
`Evolution strategy run 28`	7.1	14.6	16.1	12.5	39
`Evolution strategy run 11`	7.1	13.9	15.8	12.8	31
`COMSOL Monte Carlo`	7.0	14.3	17.2	12.9	300
`COMSOL Nelder-Mead`	7.1	14.2	15.0	12.9	255
`COMSOL BOBYQA`	7.1	13.4	14.0	14.4	45
`Agros Suite BayesOpt`	6.9	13.5	14.0	13.6	300
`Agros Suite Nelder-Mead`	6.6	13.8	14.9	12.3	160
`Agros Suite BOBYQA`	7.1	14.8	22.4	11.6	30
`ES algorithm G. Alotto et al.[21]`	7.1	13.4	13.4	15.0	N.A.
`GA algorithm G. Alotto et al.[21]`	7.2	14.0	14.0	14.6	N.A.
`Sa algorithm G. Alotto et al.[21]`	7.2	13.9	14.0	14.6	N.A.

5. Conclusions

In this paper, as far as we know, we have applied for the first time an FEM/VEM strategy to the T.E.A.M. 25 benchmark problem. The usage of the VEM spaces on some mesh elements has allowed us to both speed up the meshing procedure and consider *exactly* circle and ellipse geometries in the resolution of the problem.

To find the best configuration of this device, we define three objective functionals that are able to estimate how far the computed magnetic field is from the target one. We numerically showed that the best configuration was obtained with `run 18` of the functional F_3. Moreover, we have also observed that the data here proposed are in accordance with the ones in the literature obtained with an FEM approach.

The strategy proposed here in two dimensions can be extended to the three-dimensional case. Specifically, the FEM and the VEM can also be combined in a three-dimensional grid composed of tetrahedra, hexahedra, and generally shaped polyhedra. However, this extension presents two primary challenges. First, from a mesh generation point of view, the cut operations become more involved. Secondly, if the computational domain has smooth surfaces, the theory behind the resolution of a magnetostatic problem with the VEM is still under development.

Author Contributions: Conceptualization, F.D. and A.R.; Methodology, F.D., P.D.B. and A.R.; Software, F.D. and A.R.; Validation, F.D., P.D.B. and A.R.; Investigation, F.D. and P.D.B.; Resources, F.D., P.D.B. and A.R.; Data curation, F.D., P.D.B. and A.R.; Writing—original draft, F.D., P.D.B. and A.R.; Writing—review & editing, F.D., P.D.B. and A.R.; Visualization, P.D.B. and A.R.; Supervision, P.D.B. and A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Takahashi, N. Optimization of Die Press Model (TEAM Workshop Problem 25). In *TEAM Workshop*; CiNii: Okayama, Japan, 1996; Volume 61.
2. Li, Y.; Lei, G.; Bramerdorfer, G.; Peng, S.; Sun, X.; Zhu, J. Machine learning for design optimization of electromagnetic devices: Recent developments and future directions. *Appl. Sci.* **2021**, *11*, 1627. [CrossRef]
3. Tucci, M.; Barmada, S.; Formisano, A.; Thomopulos, D. A regularized procedure to generate a deep learning model for topology optimization of electromagnetic devices. *Electronics* **2021**, *10*, 2185. [CrossRef]
4. Brescia, E.; Costantino, D.; Massenio, P.R.; Monopoli, V.G.; Cupertino, F.; Cascella, G.L. A design method for the cogging torque minimization of permanent magnet machines with a segmented stator core based on ANN surrogate models. *Energies* **2021**, *14*, 1880. [CrossRef]
5. Barmada, S.; Fontana, N.; Thomopulos, D.; Tucci, M. Autoencoder based optimization for electromagnetics problems. *Appl. Comput. Electromagn. Soc. J.* **2019**, *34*, 1875–1880.
6. Asanuma, J.; Doi, S.; Igarashi, H. Transfer learning through deep learning: Application to topology optimization of electric motor. *IEEE Trans. Mag.* **2020**, *56*, 1–4. [CrossRef]
7. Dassi, F.; Di Barba, P.; Russo, A. A free-cutting mesh strategy for optimal shape synthesis in magnetics. *IET Sci. Meas. Technol.* **2022**, *16*, 337–352. [CrossRef]
8. Beirão da Veiga, L.; Brezzi, F.; Cangiani, A.; Manzini, G.; Marini, L.D.; Russo, A. Basic principles of Virtual Element Methods. *Math. Model. Methods Appl. Sci.* **2013**, *23*, 199–214. [CrossRef]
9. Vacca, G.; Beirão da Veiga, L. Virtual element methods for parabolic problems on polygonal meshes. *Numer. Meth. PDEs* **2015**, *31*, 2110–2134. [CrossRef]
10. Aldakheel, F.; Hudobivnik, B.; Artioli, E.; Beirão da Veiga, L.; Wriggers, P. Curvilinear virtual elements for contact mechanics. *Comput. Meth. Appl. Mech. Eng.* **2020**, *372*, 113394. [CrossRef]
11. Mascotto, L.; Perugia, I.; Pichler, A. A nonconforming Trefftz virtual element method for the Helmholtz problem. *Math. Model. Meth. Appl. Sci.* **2019**, *29*, 1619–1656. [CrossRef]
12. Cangiani, A.; Gyrya, V.; Manzini, G. The nonconforming virtual element method for the Stokes equations. *SIAM J. Numer. Anal.* **2016**, *54*, 3411–3435. [CrossRef]
13. Benedetto, M.F.; Berrone, S.; Pieraccini, S.; Scialò, S. The virtual element method for discrete fracture network simulations. *Comput. Meth. Appl. Mech. Eng.* **2014**, *280*, 135–156. [CrossRef]
14. Beirão da Veiga, L.; Brezzi, F.; Marini, L.D.; Russo, A. The hitchhiker's guide to the virtual element method. *Math. Model. Methods Appl. Sci.* **2014**, *24*, 1541–1573. [CrossRef]
15. Beirão da Veiga, L.; Lovadina, C.; Vacca, G. Divergence free virtual elements for the Stokes problem on polygonal meshes. *ESAIM Math. Model Numer. Anal.* **2017**, *51*, 509–535. [CrossRef]
16. Beirão da Veiga, L.; Russo, A.; Vacca, G. The virtual element method with curved edges. *ESAIM Math. Model. Numer. Anal.* **2019**, *53*, 375–404. [CrossRef]
17. Bianchi, N. *Electrical Machine Analysis Using Finite Elements*; CRC Press: Boca Raton, FL, USA, 2017.
18. Dassi, F. `Vem++`, a `c++` library to handle and play with the Virtual Element Method. *arXiv* **2023**, arXiv:2310.05748.
19. Di Barba, P.; Mognaschi, M.E.; Nowak, T.; Blaszczyk, A. Free-form optimisation in industrial dielectric design: A comparative approach. *Int. J. Appl. Electromagn.* **2019**, *60*, S101–S113. [CrossRef]
20. Karban, P.; Kropík, P.; Kotlan, V.; Doležel, I. Bayes approach to solving TEAM benchmark problems 22 and 25 and its comparison with other optimization techniques. *Appl. Math. Comput.* **2018**, *319*, 681–692. [CrossRef]
21. Alotto, P.; Eranda, C.; Brandstatter, B.; Furntratt, G.; Magele, C.; Molinari, G.; Nervi, M.; Preis, K.; Repetto, M.; Richter, K. Stochastic algorithms in electromagnetic optimization. *IEEE Trans. Magn.* **1998**, *34*, 3674–3684. [CrossRef]