Computation of Relative 1-Cohomology Generators From a 1-Homology Basis for Eddy Currents Boundary Integral Formulations

Bernard Kapidani[1], Paweł Dłotko[2], Piergiorgio Alotto[3], Paolo Bettini[3], and Ruben Specogna[1]

[1]Dipartimento Politecnico di Ingegneria ed Architettura, Università di Udine, Udine 33100, Italy
[2]DataShape Team, Inria Saclay, Palaiseau 91120, France
[3]Dipartimento di Ingegneria Industriale, Università di Padova, Padua 35131, Italy

Efficient boundary integral formulations based on stream functions for solving eddy current problems in thin conductors, which are modeled by the orientable combinatorial two-manifold with boundary, need generators of the first relative cohomology group to make the problem well defined. The state-of-the-art technique is to compute directly the relative cohomology generators with a combinatorial algorithm having linear worst-case complexity. In this paper, we propose to compute the relative cohomology generators from the homology generators, introducing a novel and general algorithm whose running time is again linear in the worst case. The advantage is that one may use an off-the-shelf software to compute the homology generators and implement only a simple and cheap procedure to obtain the required relative cohomology generators. Although the presented applications relate to ac power systems, the proposed technique is of general interest, and may be used for other applications in computational science and engineering.

Index Terms— **Boundary integral (BI) formulation, eddy currents, homology, relative cohomology generators, stream function.**

I. Introduction

EDDY current problems in thin conducting sheets can be solved by employing efficient boundary integral (BI) formulations based on stream functions [1]. By modeling the conducting sheet as an orientable combinatorial two-manifold $\mathcal{K}$ with boundary $\partial\mathcal{K}$ [2], such BI formulations require the relative cohomology generators $H^1(\mathcal{K}, \partial\mathcal{K})$ to make the problem well defined [3], [5]. For a formal introduction of algebraic topology, see [2]. Informal presentations for electrical engineers may be found, for example, in [3] and [4]. How to compute the required relative cohomology generators for a general two-manifold is non-trivial and remained an open issue for a long time [6]–[8]. Nowadays, the state-of-the-art technique in the context of BI formulations is to compute directly the relative cohomology generators, e.g., the combinatorial algorithm introduced in [5], which is general and exhibits a linear worst case complexity. Yet, a different approach has also been considered in literature: the relative cohomology generators may be obtained from the homology generators in post-processing.

Let us always assume that the two-manifold is oriented, and each input representative $\mathbf{h}$ of the homology generator is not self-intersecting, and that its support $|\mathbf{h}|$ is a loop of edges. This is the case for the representatives of the homology generators produced by the algorithms that compute homology with interdigitated spanning trees [9] on the primal and dual complexes. The idea, introduced in [9], is to construct a representative of the cohomology generator by taking as its support all the edges lying on one side of $|\mathbf{h}|$. If $\mathcal{K}$ is without boundary, the algorithm produces absolute cohomology generators $H^1(\mathcal{K})$.

A practical advantage of the homology-based technique with respect to [5] is that one may use the off-the-shelf software to compute homology generators and then build the required relative cohomology generators with a simple and cheap procedure. In fact, there are many codes available for computing homology, whereas computing cohomology—especially the relative one—is much less explored, and as far as we know, no efficient implementation with linear time complexity is available as open-source software, yet.

This is exactly the philosophy followed recently in [10] in which the authors claim to have extended the algorithm presented in [9] to the general case of two-manifolds with boundaries and handles. However, in Section III, we present some examples that show the lack of generality of this approach.

The main aim of this paper is thus to introduce a novel algorithm to produce the relative cohomology generators from the homology generators of the arbitrary orientable two-manifolds that does not suffer from such shortcomings. Apart from the application to the BI formulations for eddy currents, there are many others that would benefit from such an algorithm. For example, it can be used for the computation of the harmonic fields for post-processing and volumetric parametrization, and for the generation of a relative cohomology basis for volumetric integral formulations or for finite element method–boundary element method.

The paper is organized as follows. Section II discusses whether absolute or relative cohomology should be employed in eddy currents BI formulations under standard boundary conditions. Section III presents some critical examples for an algorithm presented in literature to compute the relative cohomology generators from the homology generators.

Manuscript received March 10, 2016; revised June 6, 2016; accepted June 15, 2016. Date of publication June 20, 2016; date of current version September 16, 2016. Corresponding author: R. Specogna (e-mail: ruben.specogna@uniud.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2016.2582471

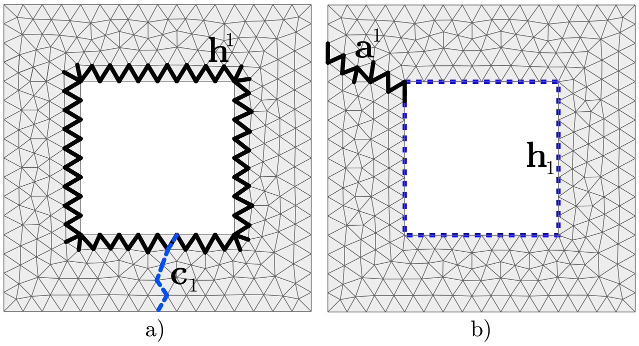

Fig. 1. (a) Support of the representative $\mathbf{h}^1$ of the $H^1(\mathcal{K}, \partial\mathcal{K})$ generator for an annulus. $\mathbf{c}_1$ (dashed edges, blue in the color version of this paper) is the support of a chain non-trivial in $H_1(\mathcal{K}, \partial\mathcal{K})$. (b) Support of the representative $\mathbf{a}^1$ of the absolute $H^1(\mathcal{K})$ generator for the annulus. Dotted loop is the support $|\mathbf{h}_1|$ of a representative $\mathbf{h}_1$ of the $H_1(\mathcal{K})$ generator.

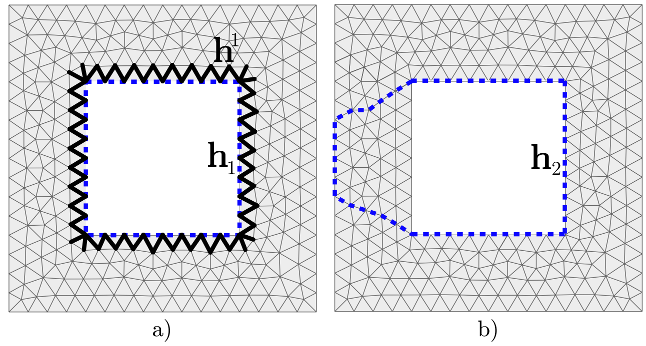

Fig. 2. (a) Thick blue dotted loop represents the support of the representative $\mathbf{h}_1$ of the homology generator. The edges on one side of the loop are the support of the representative $\mathbf{h}^1$ of the relative cohomology generator. (b) It is clear that the technique proposed in [10] cannot work in this case.

Section IV introduces a novel algorithm that does not suffer from this lack of generality. Section V shows the performances of the new algorithm on a few practical benchmarks. Finally, Section VI draws the conclusion.

II. Absolute or Relative Cohomology?

Let us consider a conductive sheet, which is an annulus covered by a triangular mesh [in Fig. 1(a) (gray)] whose topology is encoded in the cell complex $\mathcal{K}$. We denote as $\mathbf{G}$ the incidence matrix between edges and node pairs. To prevent current from flowing out of the sheet, the coefficients of the stream function 0-cochain $\mathbf{\Psi}$ related to the boundary nodes have to be set to zero. This way, the current 1-cocycle $\mathbf{I}$ defined as $\mathbf{G\Psi}$ vanishes on the boundary of the conductor.

A problem in simply defining $\mathbf{I} = \mathbf{G\Psi}$ arises when an independent current i_1 [7] flows around the annulus. This current, in fact, cannot be represented by the stream function alone, given that the evaluation of $\mathbf{G\Psi}$ on any 1-cycle non-trivial in $H_1(\mathcal{K}, \partial\mathcal{K})$ [e.g., $\mathbf{c}_1$ in Fig. 1(a)] vanishes. Indeed, such evaluation is the difference of the scalar potential on the ending and starting node of the one cycle, and these two potentials are both zero in the case of c_1. A fool-proof solution of this issue involves the use of cohomology theory. By definition, a 1-cochain $\mathbf{a}^1$ is non-trivial in the first (absolute) cohomology group $H^1(\mathcal{K})$ if it achieves the following.

1) It is a 1-cocycle, i.e., a curl-free discrete field such that $\mathbf{Ca}^1 = \mathbf{0}$, where $\mathbf{C}$ is the matrix that stores the incidence between polygons and edges.
2) It is not a 1-coboundary, i.e., it cannot be written as $\mathbf{a}^1 = \mathbf{Gp}$, where $\mathbf{p}$ is an arbitrary 0-cochain.

The support of a non-trivial element $\mathbf{a}^1$ of $H^1(\mathcal{K})$ for the annulus is represented in Fig. 1(b).

The absolute cohomology group is not suitable for the BI formulations in general. In fact, the support of 1-cycles non-trivial in $H^1(\mathcal{K})$ may involve edges on the boundary of the conductor, as it happens in $\mathbf{a}^1$ in Fig. 1(b). This would violate the boundary conditions, as it will be evident soon. That is the reason why we need the cocycles to have zero coefficients on $\partial\mathcal{K}$. A 1-cocycle non-trivial in the first relative cohomology basis $H^1(\mathcal{K}, \partial\mathcal{K})$ fulfills the two aforementioned properties, and it also avoids boundary edges in its support. For example, the support of a non-trivial element $\mathbf{h}^1$ of $H^1(\mathcal{K}, \partial\mathcal{K})$ for the annulus is represented in Fig. 1(a).

The *generators* of $H^1(\mathcal{K}, \partial\mathcal{K})$ are by definition a minimal set of non-trivial elements of $H^1(\mathcal{K}, \partial\mathcal{K})$, such that an arbitrary current 1-cocycle $\mathbf{I}$ may be written as a linear combination of them plus the coboundary of the scalar potential

$$\mathbf{I} = \mathbf{G\Psi} + \sum_{k=1}^{\beta_1(\mathcal{K})} i_k \mathbf{h}^k \tag{1}$$

where $\beta_1(\mathcal{K})$ is the first Betti number of $\mathcal{K}$ and $\{i_k\}_{k=1}^{\beta_1(\mathcal{K})}$ is a set of *independent currents* [7].

In the annulus example of Fig. 1(a), it is clear that with definition (1), the current through every relative 1-chain homologous to $\mathbf{c}_1$ is going to be always i_1. It should be now clear why absolute cohomology generators [as the representative $\mathbf{a}^1$ in Fig. 1(b)] are not the right choice in this setting. Note that the generator in [10, Fig. 1] is a representative of a relative cohomology generator, even if in this paper only the absolute cohomology generators are discussed.

III. Cohomology From Homology: Critical Cases

This section discusses the possible issues related to the extension of the algorithm presented in [9] to the case of orientable combinatorial two-manifolds with general topology. The idea behind the algorithm can be seen in Fig. 2(a). The triangles represent the mesh of an annulus. The algorithm starts from the given representative $\mathbf{h}_1$ of the homology generator, whose support $|\mathbf{h}_1|$ is the set of thick blue dotted edges. The support of the representative of the relative cohomology generator may be retrieved as all edges that lie on one side of the loop $|\mathbf{h}_1|$ and have exactly one node in their boundary in common with the support of the loop (black thick edges). This is the procedure performed by the algorithm in [10]. We remark that, thanks to Lefschetz–Poincarè duality [2], the homology generators can be put in one to one correspondence to a relative cohomology generator

$$H_1(\mathcal{K}) \cong H^1(\mathcal{K}, \partial\mathcal{K}). \tag{2}$$

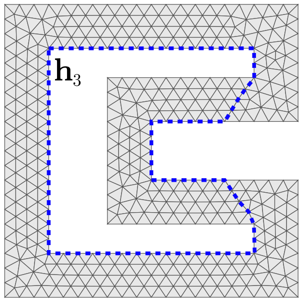

Fig. 3. Mesh of an annulus represented in gray. The thick dotted cycle represents the support of the shortest representative $\mathbf{h}_3$ of the homology generator.

Despite its apparent simplicity, this algorithm may fail. As a first example, consider Fig. 2(b) in which the representative of the homology generator touches both connected components in the boundary of the two-manifold. In this case, the implementation of the algorithm in [10] will fail because there is no room on either side of the support of the representative $\mathbf{h}_2$ of the homology generator to accommodate the support of the representative of the relative cohomology generator. Yet, the representative $\mathbf{h}_2$ is in the same homology equivalence class as $\mathbf{h}_1$, and since the code producing the homology generators is supposed to be a black box, there is nothing one can do to avoid this situation in general. One may try to solve this issue by considering the shortest basis of homology generators. Nonetheless, also this approach will fail in some cases, e.g., the counter-example in Fig. 3. Special techniques can be devised to address these problems; however, the resulting algorithm will not be as simple as in its original spirit.

IV. Novel Two-Sided Algorithm

A general and robust novel algorithm to compute the relative cohomology generators from the homology generators is presented in this section. The proposed two-sided (TS) algorithm is based on the idea of perturbing the loop $|\mathbf{h}|$ into a neighboring dual loop (i.e., a loop made of dual edges). Thus, this dual loop is the support of a 1-cycle $\tilde{\mathbf{c}}$ on the dual complex $\tilde{\mathcal{K}}$. Such perturbation is performed locally for every node of $|\mathbf{h}|$ by adding a suitable topologically trivial surface. This guarantees that the homology class of $\mathbf{h}$ and $\tilde{\mathbf{c}}$ is the same. First, consider the node n_1 in Fig. 4(a). Pick the dual face $\tilde{f}_1$ dual to n_1. Such dual face is always partitioned into two surfaces by cutting it across the edges in $|\mathbf{h}|$. We note that there are always two of such edges, since $|\mathbf{h}|$ is not self-intersecting. Pick the boundary of one of such surfaces (for example, the red one referring to the color version of the paper), and add it to $\mathbf{h}$ to obtain as supporting the thick dotted loop in Fig. 4(a). Not always we can choose between two surfaces. For example, once we consider the local modification of n_2 in Fig. 4(b), due to the presence of the boundary of $\mathcal{K}$, only one of the two surfaces is suitable for our purpose (again, the red one in the color version of the paper). So these are the only two cases that may happen. In fact, there is always at least one of such good surfaces due to the fact that $\mathcal{K}$ is a manifold with boundary.

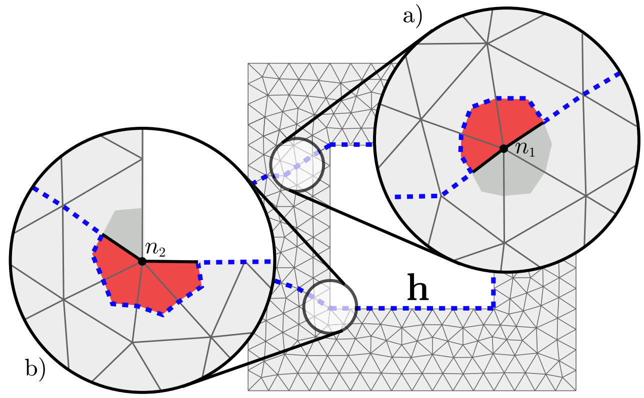

Fig. 4. Mesh of an annulus represented in gray. The thick dotted loop represents the support $|\mathbf{h}|$ of the given representative $\mathbf{h}$ of the homology generator. Example of the local modifications of $|\mathbf{h}|$ for the nodes (a) n_1 and (b) n_2.

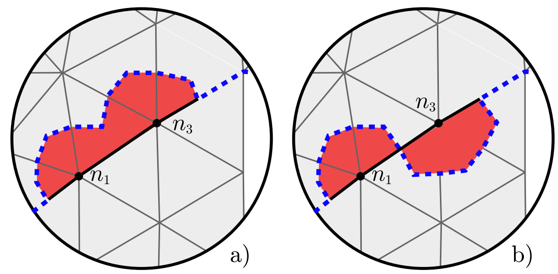

Fig. 5. Two possible configurations in the neighborhood of node n_1. (a) Two adjacent surfaces are chosen on the same side with respect to the $|\mathbf{h}|$ loop. (b) Two adjacent surfaces are chosen on the opposite side with respect to the $|\mathbf{h}|$ loop.

In principle, by choosing the surface to be added randomly, one may treat each node of $|\mathbf{h}|$ in parallel, since each local modification does not rely on what happens in the neighboring nodes. We note that two cases may arise, once we consider the superposition of effects of the two local modifications (see Fig. 5). In the first case [Fig. 5(a)], two adjacent surfaces are chosen on the same side with respect to $|\mathbf{h}|$ loop, whereas in the second case [Fig. 5(b)], the surfaces are chosen on the opposite side. In the first case, the two portions of dual edge cancel out (since the dual loop $|\tilde{\mathbf{c}}|$ goes in the middle point of the edge, and then goes back through the same dual edge) leaving a piece of path on the dual complex. In the second case, the two portions are added to form an entire dual edge that crosses $|\mathbf{h}|$.

Once all the nodes in $|\mathbf{h}|$ have been treated, we obtain a dual 1-cycle $\tilde{\mathbf{c}}$, which is in the same homology class as $\mathbf{h}$ [see Fig. 6(a)]. $\tilde{\mathbf{c}}$ may be represented with an array of integer coefficients, one for each dual edge. Thanks to (2) and the duality between the primal and dual complex, if we interpret the same array as the coefficients of a 1-cochain on the primal complex, we get the representative $\mathbf{c} = D(\tilde{\mathbf{c}})$ of a $H^1(\mathcal{K}, \partial\mathcal{K})$ generator [see Fig. 6(b)], where D is the map between the corresponding elements of the primal and dual complex. The basis property of the obtained $H^1(\mathcal{K}, \partial\mathcal{K})$ generators is inherited from the homology basis provided as input.

In the algorithm that we are about to describe, our aim is to get the representative $\mathbf{c}$ from $\mathbf{h}$. Let us fix two edges $E_1, E_2 \in \mathcal{K}$ such that $E_1 \cap E_2 \neq \emptyset$. Let us define the

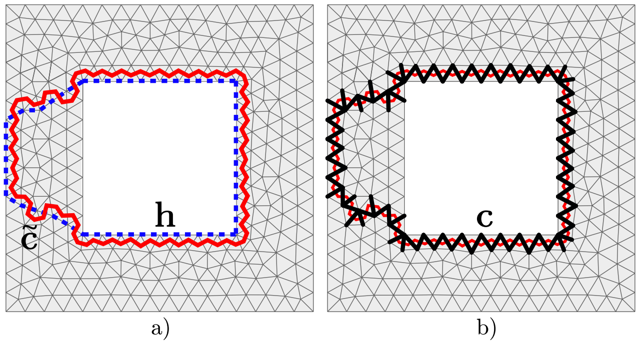

Fig. 6. (a) Once all the nodes are treated, one obtains a dual cycle $\tilde{\mathbf{c}}$ whose support is $|\tilde{\mathbf{c}}|$ (in red in the color version of the paper). (b) Edges of $\mathcal{K}$ dual to the edges in $|\tilde{\mathbf{c}}|$ form the support of the relative cohomology generator $\mathbf{c}$.

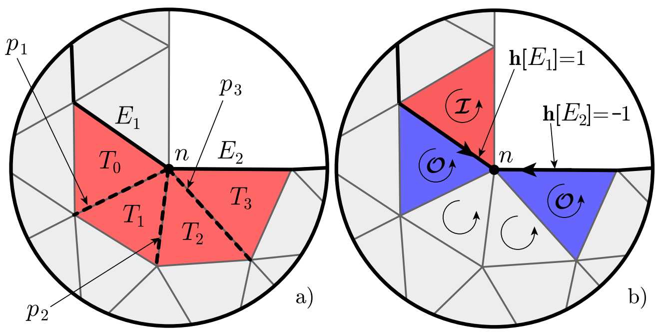

Fig. 7. (a) For the configuration represented in the figure, the propagation edges P_{E_1,E_2} are p_1, p_2, and p_3. The associated collection of 2-cells is T_0, T_1, T_2, and T_3. (b) Sets $\mathcal{I}$ and $\mathcal{O}$ for the same configuration.

propagation edges P_{E_1,E_2} between E_1 and E_2 as any collection of edges $p_1, \ldots, p_m$ (without repetitions), such that there exists a collection of 2-cells $T_0, T_1, \ldots, T_m$ such that E_1, $p_1 \in \partial T_0$, $p_i, p_{i+1} \in \partial T_i$ for $i \in \{1, \ldots, m-1\}$ and $p_m, E_2 \in \partial T_m$ [see Fig. 7(a)]. Note that since $\mathcal{K}$ is a manifold with boundary, for every two 1-cells E_1 and E_2 such that $E_1 \cap E_2 \neq \emptyset$, there is at least one and at most two collections of the edges in P_{E_1,E_2}.

Given $n = E_1 \cap E_2$, let us now introduce the propagation cochain $\mathbf{p}_{E_1,E_2}$ as the 1-cochain whose eth coefficient is defined as follows:

$$\mathbf{p}_{E_1,E_2}[e] = \begin{cases} \mathbf{G}[e,n], & \forall e \in P_{E_1,E_2} \\ \frac{1}{2}\mathbf{G}[e,n], & \text{for } e = E_1, E_2 \\ 0, & \text{elsewhere.} \end{cases} \quad (3)$$

The TS algorithm works as follows. By $\mathbf{h}[e]$, we denote the coefficient of the 1-chain $\mathbf{h}$ related to the edge e. For every node n in the loop $|\mathbf{h}|$ and for every pair of edges E_1 and E_2 in $|\mathbf{h}|$ such that $E_1 \cap E_2 = n$, first, the algorithm finds $\mathbf{p}_{E_1,E_2}$, for example with a standard local breadth-first search. For each computed $\mathbf{p}_{E_1,E_2}$, we also need to know which side of the loop $|\mathbf{h}|$ we are in (see Fig. 5).

To get this information, let us observe that, given a consistent orientation on the top-dimensional cells of $\mathcal{K}$ and an oriented loop $|\mathbf{h}|$, the set of 2-cells (S such that $|\partial S| \cap |\mathbf{h}| \neq \emptyset$) can be partitioned into two sets: the ones where $\partial S[|\partial S| \cap |\mathbf{h}|] = \mathbf{h}[|\partial S| \cap |\mathbf{h}|]$, called $\mathcal{I}$, and the ones where $\partial S[|\partial S| \cap |\mathbf{h}|] \neq \mathbf{h}[|\partial S| \cap |\mathbf{h}|]$, called $\mathcal{O}$ [see Fig. 7(b)].

Algorithm 1 Pseudocode of the TS Algorithm

Require: two-manifold with boundary $\mathcal{K}$, non self-intersecting, 1-cycle $\mathbf{h}$ in $\mathcal{K}$

1: $\mathbf{c}$, 1-cocycle, initially all zeros
2: $N = \bigcup_{E \in |\mathbf{h}|} \partial E$
3: **for all** $n \in N$ **do**
4: Find $\mathbf{p}_{E_1,E_2}$ with $E_1, E_2 \in |\mathbf{h}|$ and $n \in \partial E_1 \cap \partial E_2$
5: $\sigma = \mathbf{C}[T_0, E_1] \cdot \mathbf{h}[E_1]$
6: $\mathbf{c} = \mathbf{c} + \sigma \cdot \mathbf{p}_{E_1,E_2}$
7: **end for**
8: Return $\mathbf{c}$

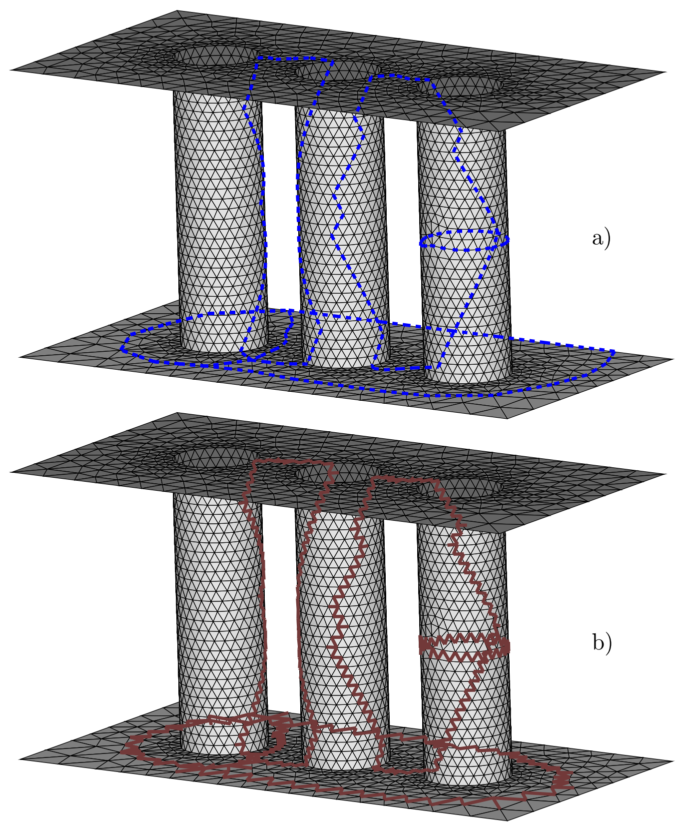

Fig. 8. (a) Input mesh of a shield. Dashed lines: support of the five representatives of a $H_1(\mathcal{K})$ basis. (b) Thick edges are the support of the representatives of the $H^1(\mathcal{K}, \partial\mathcal{K})$ generators.

It is easy to verify that the 2-cells from $\mathcal{I}$ and $\mathcal{O}$ lie on the opposite sides of $|\mathbf{h}|$. The TS algorithm will orient the constructed cocycle toward the 2-cells of $\mathcal{I}$. To do that, we need to keep track of which side of the loop $|\mathbf{h}|$ we are on, and pick the orientation of the propagation cochain accordingly. This information is given by σ, which is computed in line 5 of Algorithm 1, where T_0, as in the definition of the propagation edges, is the first 2-cell in P_{E_1,E_2} and $\mathbf{C}$ is a sparse matrix, which contains the incidence between faces and edge pairs. For example, in the configuration presented in Fig. 7(b), $\sigma = -1$. This way, if the local modifications to adjacent pairs of edges in $|\mathbf{h}|$ are on the opposite sides of $|\mathbf{h}|$ [as in Fig. 5(b)], the two half edges that result from (3) will add up to 1 or -1; otherwise, [if they are, as in Fig. 5(a)] they will cancel each other out.

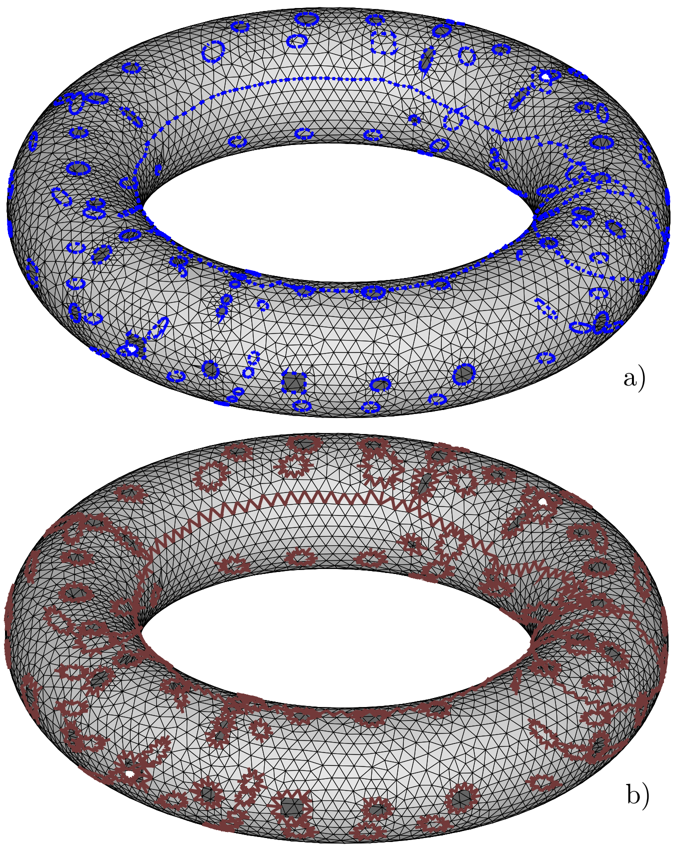

Fig. 9. (a) Input mesh of a shield part of a fusion device. Dashed lines: support of the 103 representatives of a $H_1(\mathcal{K})$ basis. (b) Thick edges are the support of the representatives of the $H^1(\mathcal{K}, \partial\mathcal{K})$ generators.

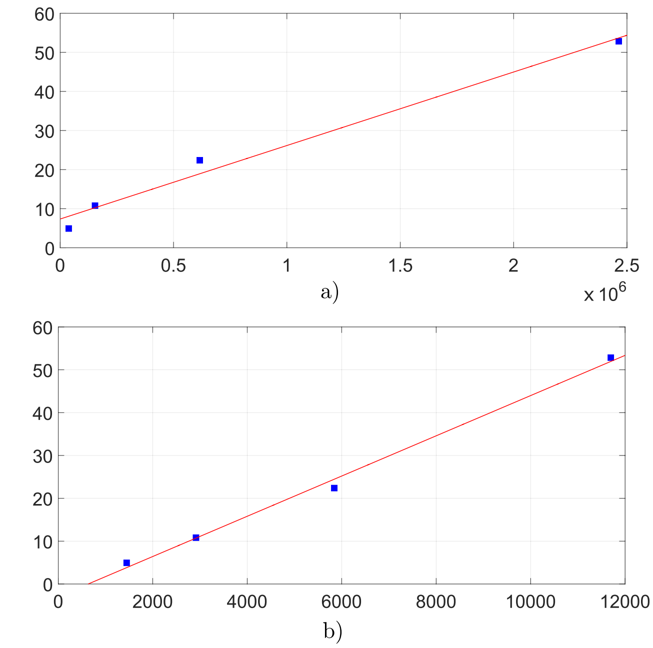

Fig. 10. (a) Computational time, in milliseconds, required by the TS algorithm on the example of Fig. 8, with respect to the number of millions of edges in the mesh. (b) Computational time, in milliseconds, required for the TS algorithm on the same example, but with respect to the size of the support of the 1-homology generators (the number of nodes is equal to the number of edges, since the input is a set of loops).

The 1-cochain $\mathbf{c}$ produced by the TS algorithm is a 1-cocycle, since for every 2-cell $S \in P_{E_1,E_2}$, $|\partial S| \cap |\mathbf{c}|$ consists of two edges that are oriented both as sources or both as sinks with respect to the node n. Therefore, they have opposite incidence with respect to the boundary of S. Moreover, since the 1-cycle dual to $\mathbf{c}$ is in the same homology class as $\mathbf{h}$, $\mathbf{c}$ is the representative of a $H^1(\mathcal{K}, \partial\mathcal{K})$ generator.

V. Numerical Experiments

In this section, we present the validation of the TS algorithm on some benchmarks of practical interest. The algorithm has been implemented in C++, and all computations have been run on a Intel(R) Core(TM) i7-6500U CPU running at 2.50 GHz with 8 GB of RAM memory. For homology computation, we used the dual version of the efficient technique described in [5]. Yet, we remark that one may use any black box software that computes the first homology group generators. The output of this computation is fed to the TS algorithm. We remark that the running time of the implemented TS algorithm is always well under one second, even on the meshes of millions of triangular elements; therefore, the possible parallelism of the algorithm has not been exploited.

Fig. 8(a) shows a mesh of a part of a shield for a three-phase ac component, homeomorphic to a twofold torus with two holes, inspired by [6]. Dashed lines represent the support of the five representatives of a $H_1(\mathcal{K})$ basis, while in Fig. 8(b), the thick edges belong to the support of the correspondent representatives of relative cohomology generators.

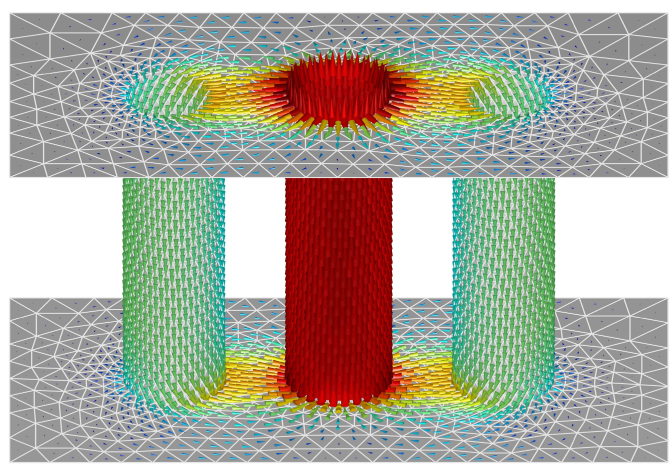

Fig. 11. Eddy currents induced in the first shield by a three-phase busbar system where the source ac currents at 50 Hz flow through the cylinders. The mesh consists of 6500 triangular elements and 3301 nodes.

Fig. 9(a) shows a mesh of another shield, part of a fusion device, i.e., homeomorphic to a torus with 102 holes. Dashed lines represent the support of the 103 representatives of a $H_1(\mathcal{K})$ basis, while in Fig. 9(b), the thick edges belong to the correspondent representatives of the $H^1(\mathcal{K}, \partial\mathcal{K})$ generators (101 due to holes and two due to the handle of the torus). For the sake of clarity, we show a coarse mesh with 11 593 triangles and 6187 nodes. We tested the procedure on a finer mesh with 794 802 triangles and 399 589 nodes. The running time of

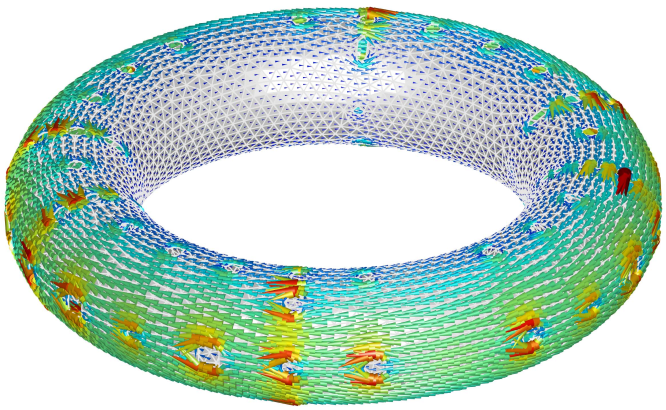

Fig. 12. Eddy currents induced in the second shield by a uniform field ($B = 1$ T and $f = 100$ Hz). The mesh consists of 11 593 triangular elements and 6187 nodes.

the TS algorithm on such refined mesh is still a mere 30 ms. Instead, in Fig. 10, we show how the computational times behave with respect to increasing the number of elements in the input mesh and the consequent increase of the support of the input.

Finally, Figs. 11 and 12 validate the TS algorithm by computing the eddy currents that are induced on the shields and by verifying that they are the same, up to linear solver tolerance, as the ones obtained as described in [5]. To solve the problem with a BI formulation on the refined meshes, full matrices have been sparsified according to the technique described in [11].

VI. Conclusion

This paper introduces a TS algorithm to produce the relative first cohomology group generators from a first homology group basis provided as input that may be obtained with any black box software. The TS algorithm is general, and shows an optimal computational complexity, which should spread its use in industrial applications.

Acknowledgment

The work of P. Dłotko was supported by the European Research Council through the Advanced Grant Project Geometric Understanding in Higher Dimensions.

References

[1] A. Kameari, "Transient eddy current analysis on thin conductors with arbitrary connections and shapes," *J. Comput. Phys.*, vol. 42, no. 1, pp. 124–140, Jul. 1981.

[2] J. R. Munkres, *Elements of Algebraic Topology*. Menlo Park, CA, USA: Addison-Wesley, 1993.

[3] P. W. Gross and P. R. Kotiuga, *Electromagnetic Theory and Computation: A Topological Approach* (MSRI Monograph Series), vol. 48. Cambridge, U.K. Cambridge Univ. Press, 2004.

[4] P. Dłotko and R. Specogna, "Cohomology in 3D magneto-quasistatics modeling," *Commun. Comput. Phys.*, vol. 14, no. 1, pp. 48–76, 2013.

[5] P. Bettini and R. Specogna, "A boundary integral method for computing eddy currents in thin conductors of arbitrary topology," *IEEE Trans. Magn.*, vol. 51, no. 3, Mar. 2015, Art. no. 7203904.

[6] A. Bossavit, "Eddy currents on thin shells," in *Proc. 3rd Int. Workshop Electr. Magn. Fields*, Liège, Belgium, May 1996, pp. 453–458.

[7] P. Dłotko and R. Specogna, "Physics inspired algorithms for (co)homology computations of three-dimensional combinatorial manifolds with boundary," *Comput. Phys. Commun.*, vol. 184, no. 10, pp. 2257–2266, Oct. 2013.

[8] M. Pellikka, S. Suuriniemi, L. Kettunen, and C. Geuzaine, "Homology and cohomology computation in finite element modeling," *SIAM J. Sci. Comput.*, vol. 35, no. 5, pp. B1195–B1214, 2013.

[9] R. Hiptmair and J. Ostrowski, "Generators of $H_1(\Gamma_h, \mathbb{Z})$ for triangulated surfaces: Construction and classification," *SIAM J. Comput.*, vol. 31, no. 5, pp. 1405–1423, 2002.

[10] F. Moro, P. Alotto, A. Stella, and M. Guarnieri, "Solving 3-D eddy currents in thin shells of any shape and topology," *IEEE Trans. Magn.*, vol. 51, no. 3, Mar. 2015, Art. no. 7203104.

[11] P. Alotto, P. Bettini, and R. Specogna, "Sparsification of BEM matrices for large-scale eddy current problems," *IEEE Trans. Magn.*, vol. 52, no. 3, Mar. 2016, Art. no. 7203204.