Received: 10 June 2021 | Revised: 21 September 2021 | Accepted: 26 October 2021

DOI: 10.1002/nme.6863

RESEARCH ARTICLE

WILEY

Density gradient-based adaptive refinement of analysis mesh for efficient multiresolution topology optimization

Francesco Mezzadri[1] | **Xiaoping Qian**[2]

[1]Department of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia, Modena, Italy

[2]Department of Mechanical Engineering, University of Wisconsin-Madison, Madison, Wisconsin, USA

Correspondence
Francesco Mezzadri, Department of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia, Via P. Vivarelli 10/1, Modena, I-41125, Italy.
Email: francesco.mezzadri@unimore.it

Funding information
National Science Foundation, Grant/Award Number: 1941206; Office of Naval Research, Grant/Award Number: N00014-18-1-2685; Università Degli Studi di Modena e Reggio Emila, Grant/Award Number: A.004@ECDL@GALLIGANI

Abstract

In topology optimization, the finite-element analysis of the problem is generally the most computationally demanding task of the solution process. In order to improve the efficiency of this phase, in this article we propose to represent regions with zero density gradient by a coarser analysis mesh. The design is instead represented in a uniform mesh. We motivate the density gradient-based adaptive refinement by discussing the topological meaning of the density gradient and how it can help avoid loss of information during projections or interpolations between design and analysis meshes. We also study the adaptiveness of the mesh and its ability to detect the topology change of the design. An a posteriori error analysis is performed as well. Furthermore, we provide theoretical and numerical considerations on the reduction of the number of degrees of freedoms of the adaptive analysis mesh with respect to the uniform case. This translates into a faster solution of the analysis, as we show numerically. Finally, we solve several test problems, including large 3D problems that we solve in parallel on computer cluster, demonstrating the applicability of our procedure in large scale computing and with iterative solvers.

KEYWORDS

adaptive analysis mesh, density gradient, topology optimization

Topology optimization (TO) allows to compute optimized designs in a variety of different situations.[1] Thanks to its versatility, TO is used also in applied contexts and to solve large problems. This makes the efficiency of the computation of the solution particularly important, especially when large-scale problems need to be solved.

In this regard, many ways exist to reduce the computational cost, including the use of parallel computing,[2,3] the employment of fast iterative solvers,[4,5] and approximate reanalysis procedures.[6] Another popular approach consists in using adaptive meshes. For instance, adaptive mesh refinement has been used to solve problems in stress-constrained optimization,[7,8] and anisotropic adaptive meshes can also allow to reduce or even skip the postprocessing phase.[9,10]

A common feature to many adaptive meshing approaches is that void regions are generally represented by a coarser mesh than other zones, in order to save as many elements as possible. For instance,[11] refines zones at the interface between solid and void regions, while[12] refines solid regions and interfaces with the void, while simultaneously de-refining void regions. In both these cases, a single mesh is used, so that the adaptivity is present in the representation of the design, in the optimization, and in the finite element analysis. However, a typical TO process can be seen as made by at least

two phases: one where the finite element analysis is performed and the physical response is computed, and one where the optimization is conducted. This latter phase is generally less expensive, but it requires a fine discretization whenever high-resolution designs are required. Thus, while typical TO approaches employ the same discretization in both phases of the procedure, various methods have been proposed in recent years that decouple the discretizations of analysis and design in order to improve the efficiency. In particular, while the idea of separating the grids that are used to model the geometrical and physical response can be traced back to articles like,[13-15] one of the first works primarily devoted to the study of procedures where analysis and design meshes are decoupled is.[16] In particular, in Reference 16 the computational cost is reduced by adaptively regulating the number of design variables, while the analysis mesh is kept fixed. Thus, while the cost of the finite element analysis is fixed, the number of KKT equations and all computations linked to the optimization are reduced. Reference[17] focused, instead, on reducing the number of analysis variables. In particular,[17] considered three different meshes: a *displacement mesh* (coarser, where the analysis is performed), a *design variable mesh* (fine, where the optimization is conducted) and a *density mesh* (the finest mesh, where the densities are represented). In Reference,[17] the name of *multiresolution topology optimization* (MTO) was also introduced to name these approaches that employ more than one mesh. As the finite element analysis is the most computationally onerous step, many other works followed the idea of reducing the resolution of the analysis mesh with respect to the design, including, for instance,.[18-23] The time saving can be remarkable: for instance, in Reference 23, it is remarked that MTO can be approximately up to 3 times faster than traditional TO in 2D settings and up to 30 times faster in 3D problems. Multiresolution topology optimization has been employed also in applied problems.[24,25]

In this context, efficient analyses meshes are often adaptive, and can be built by using a variety of refinement criteria that identify which elements need to be refined. Refinements based on a posteriori evaluation of the error are common (both in MTO and in "standard" adaptive mesh refinement strategies), like in References 20 and 23. For instance, in the recent work,[23] three indicators are used to define which elements to refine: an analysis-based refinement indicator (based on the Kelly a posteriori error estimator[26]), a density-based refinement indicator (which aims at reducing intermediate densities) and a QR-error indicator. Finally, geometry criteria are also possible, as, for instance, has been recently done in TO based on hierarchical B-splines.[27]

In this article, we propose an adaptive analysis mesh that is built on the basis of the gradient of the density function (hereafter named, for compactness, density gradient). This refinement criterion is fundamentally geometry-based, but it inherently contains important physical information. For instance, the density gradient has been recently used to impose design-dependent boundary loadings[28] and manufacturability constraints.[29,30] Furthermore, it allows to easily detect the solid-void interfaces of the design, which are the regions where the topological change occurs at every iteration. Finally, it is easy to compute. We further improve the efficiency of the proposed density gradient-based mesh by introducing a measure of the change of density distribution, which we use to decide if we can skip the remeshing at a given iteration. The construction of the adaptive analysis mesh is, therefore, simple and inexpensive with respect to the finite-element analysis. Furthermore, we provide practical and mathematical justifications to the use of the density gradient as a refinement indicator, and we show numerically, by an a-posteriori error analysis, that we can obtain sufficiently accurate results of the finite-element analysis in our density gradient-based analysis mesh. We also provide theoretical and practical estimates of the expected savings and solve various numerical experiments, including large-scale 3D problems with several millions of design variables.

More precisely, in our approach we use a fine and uniform design mesh, while the analysis mesh is fine only where the density gradient is nonzero, that is, along solid-void interfaces and in gray regions. In all other regions, the analysis mesh is coarser. In this, our choice is similar to Reference 27, where the user-defined refinement criterion was based on refining the interfaces between solid and void regions. However, beyond the obvious differences in the general setting (Reference 27 uses the extended finite-element method and hierarchical B-splines), the refinement in Reference 27 is based on evaluating whether the level set function in an element corresponds to solid, void, or solid-void interface. All three instances are treated differently, and the criterion is geometrical. Furthermore, in practice, the maximum refinement level for interface and solid elements is equal in the numerical experiments, and the solid has a uniform mesh. We, on the other hand, consider a traditional nodal-based finite-element setting, and base our refinement uniquely on the density gradient. Thus, no additional distinction between solid and void is performed, and the density gradient conveys also a physical meaning. Furthermore, we avoid remeshing at each iteration by a measure of the change of density distribution. Finally, using the density gradient, we provide estimates of the expected computational saving for a given design.

To build this adaptive analysis mesh, we start from a uniform, coarse grid. Then, solid-void interfaces are progressively refined by a sequence of density gradient-based refinements, until they have the same resolution as the design mesh.

This process is illustrated in Figure 1 for a 2D design. The figure also reports the final data of the analysis and design grids.

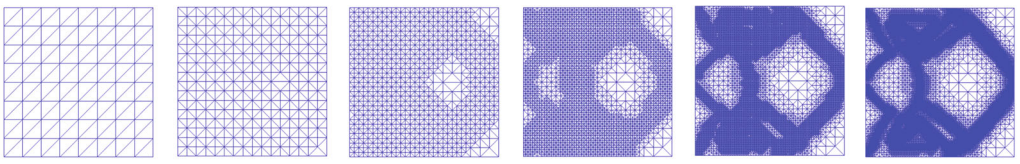
The main novelties and peculiarities of our approach are, then, the following.

- The construction of our analysis mesh is directly based on the design, and evolves with it, possibly at each iteration. This is done avoiding an explicit error analysis, but by employing directly the density gradient as a refinement indicator;
- The analysis resolution is regulated by a sequence of density gradient-based refinements, and the zones where the analysis mesh is fine are thus governed uniquely by the "topological" importance of the region. Although we here set the finest analysis resolution to be the same as the design, the procedure is general and other choices can be made, if needed. For instance, it is possible to require that the analysis mesh is finer than the design in regions with nonzero density gradient if a high-resolution analysis is necessary.
- By its reliance only on density gradient information, the analysis mesh is simple and relatively inexpensive to compute, also for large 3D problems. Moreover, we further improve the efficiency by avoiding to remesh at every iteration, on the basis of a measure of the change of density distribution.
- Our procedure can be easily adapted to different contexts. For instance, we use it also in combination of parallel computing and of iterative solvers. Furthermore, when the elements to be refined have been chosen by our density gradient-based strategy, any refinement algorithm can be used to actually build the mesh, with possible additional efficiency improvements.

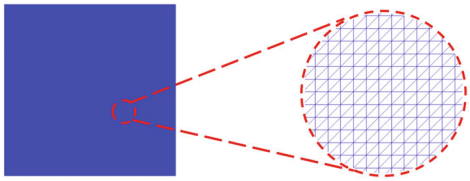
Our mesh is competitive with respect to recently proposed approaches. For instance, in the recent paper,[23] a 2D cantilever beam problem at the last refinement cycle allowed to reduce the number of design variables by over 50%. In our 2D experiments, we achieve a saving in terms of analysis variables between 50% and 70% in most cases, depending on the problem. Smaller efficiency (around 30%) is registered only for small test problems, while larger problems are associated to higher efficiency (exceeding 80% in the largest 2D problem we solved). The saving in the 3D experiments is even more significant: the estimates of the achieved speed-up suggest a 10 times improvement of the computational cost for analysis when the density gradient-based mesh is used in the largest problem, with a peak of over 20 times improvement at some iterations.

The article is structured as follows. In Section 1 we describe the general idea behind the proposed density gradient-based analysis mesh, and motivate it especially by topological considerations. In Section 2 we then describe how our mesh is built. In this regard, we here formulate also the above-mentioned measure of the change of density distribution and we provide an algorithmic form of the mesh construction. In Section 3 we analyze the proposed analysis mesh, devoting particular attention to its ability to detect the topology change and to nucleate all the features of the design. We also discuss the advantages that our mesh has for the projections and interpolations between design and analysis meshes, and provide an a posteriori error analysis that confirms the good behavior of the density gradient-based mesh. In Section 4

(A) Optimized design



(B) Computation of density gradient-based analysis mesh through five refinements. Final size 58,015 cells and 29,176 nodes.



(C) Uniform design mesh. Size 131,072 cells and 66,049 nodes.

FIGURE 1 Progressive local refinement of the analysis mesh based on the gradient of the density distribution

we analyze the computational saving. We especially focus on the reduction of the variables of the analysis problems, but we also provide various remarks on computational times. All considerations are extended to 3D problems as well. In Section 5 we then solve several numerical experiments, including large 3D problems. Finally, Section 6 concludes this work.

1 | GENERAL IDEA OF DENSITY GRADIENT-BASED ANALYSIS MESH

We work with two meshes throughout the optimization process:

- a uniform design mesh, whose resolution is denoted by r_d. This is the mesh where the density "lives".
- an adaptive analysis mesh, where the analysis (i.e., the solution of the primal and adjoint problems) is performed and whose resolution is denoted by r_a. The analysis mesh is computed starting from an initial uniform mesh of resolution r_{a0}.

Throughout the article, the resolutions refer globally to the entire mesh when we refer to a uniform mesh. When, instead, we refer to an adaptive mesh, the actual resolution changes depending on the considered region of the domain. Therefore, when we discuss r_a, we refer to specific regions of a domain, and not to a global, constant resolution.

Employing two meshes is motivated by the fact that analysis is, generally, the most computationally expensive task. Thus, we will try to coarsen the analysis mesh as much as possible. On the other hand, a high-resolution design mesh is needed to represent the finer features of the topology. Therefore, we may want to use a fine design mesh. The aim is to obtain a final design with the high resolution of the design mesh, but with a smaller cost to solve analysis.

In order to do this, at every iteration or when a suitable condition is satisfied, the analysis mesh is created in accordance with the following criteria:

1. the analysis mesh must satisfy $r_a = r_d$ in every region where some topology change is happening;
2. to improve the efficiency, the analysis mesh must satisfy $r_a < r_d$ in regions that are not topologically active;
3. multiple local refinements (or coarsenings, depending on the implementation) are employed, with the aim to coarsen r_a as much as possible in zones that are less topologically active.

For instance, an example of our adaptive analysis mesh over a design is reported in Figure 2. In topologically active zones, the mesh has been refined up to r_d by four successive local refinements, starting from a uniform mesh of resolution r_{a0}. These multiple refinements are visible in the zoom in reported in Figure 2C. In larger void and solid regions, the resolution remains, instead, unaltered, and equal to r_{a0}.

1.1 | Topological relevance of the different parts of the design

In order to satisfy the above-described requirements, we need to evaluate how "topologically important" the different parts of the design are. In this regard, we distinguish among solid-void interfaces, zones involved in the nucleation of holes, and solid and void regions.

1.1.1 | Solid-void interfaces

The regions between solid and void parts of the design (henceforth called "solid-void interfaces") are extremely important in forming the final design. Indeed, they can be seen as the regions where most of the topology change actually happens: evidently, if the solid-void interfaces do not change, the design does not change and the optimization process has stopped. If, instead, solid-void interfaces are evolving, they guide the evolution of the design. Thus, it is crucial that these zones are characterized by a high-resolution analysis mesh in order to correctly detect the topology change.

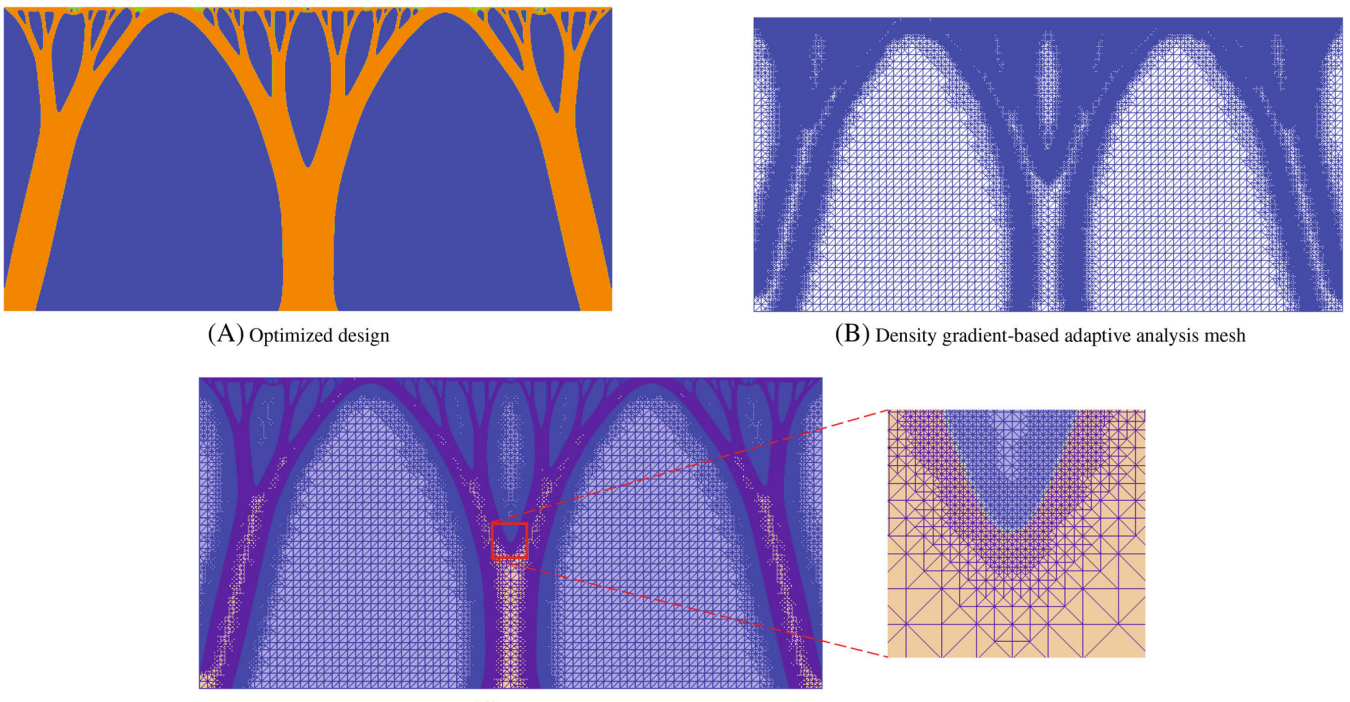
1.1.2 | Zones involved in hole formation

Evidently, if a hole is immediately nucleated, we just have a new solid-void interface and fall in the previous case. However, holes may also nucleate over the course of various iterations: first, a zone with intermediate densities may appear in a solid region and, then, progressively nucleate a hole. During this hole formation, there is not an actual solid-void interface, but the zone is evidently topologically active.

1.1.3 | Solid and void zones

The solid and void regions are less critical. In particular, the analysis mesh can be coarsened significantly in void regions, as is done also in other approaches in the literature and not only in multiresolution TO (like, for instance, in Reference 12).

Also solid regions, although more important, can be viewed as topologically inactive and can have a coarser analysis mesh. Indeed, solid regions that are far from a solid-void interface can, arguably, become topologically active only if a hole is nucleated. In this case, however, either a new solid-void interface or a new gray region appears. As soon as this happens, we fall into one of the previously analyzed topologically relevant cases and the analysis mesh is refined. Therefore, it is enough that the analysis mesh is sufficiently accurate to start the hole formation (i.e., to detect a local increase of intermediate densities). Regarding solid regions that are close to a solid-void interface, instead, they are already characterized by a fine analysis mesh, as long as the analysis mesh is coarsened gradually as we get farther from a solid-void interface. This can be observed also in Figure 2. Thus, if the solid-void interface gets closer to a solid (or void) node, the analysis mesh will be gradually refined, ensuring a sufficient accuracy. Conversely, if the solid-void interface gets farther from a solid (or void) node, the analysis mesh will be coarsened, in order to improve the efficiency of the procedure. In this regard, it is worth noticing that, in other contexts, also some error-based meshes for topology optimization exhibit quite coarse mesh in solid regions, as it happens, for instance, in some anisotropic meshes (e.g., see References 9 and 10).



(A) Optimized design

(B) Density gradient-based adaptive analysis mesh

(C) Overlay of the analysis mesh and the design

FIGURE 2 A design and the corresponding density gradient-based analysis mesh

1.2 | Identifying topologically relevant regions: a density gradient-based approach

Throughout the article, for simplicity and with no loss of generality, we assume to use nodal-based densities and material interpolation. It is easy to notice that the previously described topologically important zones are generally characterized by some degree of intermediate densities. This is particularly evident at earlier iterations, where the grayness between solid and void can involve great part of the domain. Then, during the iterations, the grayness is gradually reduced (possibly, also by the use of suitable filters), but it remains, generally, present, at least on a numerical level. In the limit case where intermediate densities are completely suppressed, we nonetheless have elements where some nodes have void densities and others have solid densities. Analogously to intermediate densities, this can be used to detect topologically relevant zones.

Indeed, in all these cases, we have elements that are characterized by a nonzero gradient of the density function. In fact, in the solid and void regions, the density is evidently constant and its gradient is zero. The regions which are topologically active are, instead, characterized by a nonconstant density. The gradient is, thus, nonzero, as illustrated in Figure 3.

Then, denoting by γ the density function, we can use its gradient as a measure which defines the zones where the analysis mesh should be refined:

$$\begin{aligned} \nabla\gamma \neq \mathbf{0} \quad &\rightarrow \quad \text{topologically active region,} \\ \nabla\gamma = \mathbf{0} \quad &\rightarrow \quad \text{not topologically active region.} \end{aligned} \tag{1}$$

Furthermore, setting $r_a = r_d$ at solid-void interfaces gives an additional advantage. Indeed, if we use two different meshes for design and for analysis, at some point we are going to have to project or interpolate the density from the design onto the analysis mesh, in order to perform the analysis. Zones with intermediate densities are very sensitive to such operations. Indeed, if $r_a < r_d$ where intermediate densities are present, we are going to lose information, as we later describe in Section 3.3. However, regions characterized by intermediate densities will have, in general, a nonzero density gradient. Therefore, our density gradient-based analysis mesh avoids loss of information in projections and interpolations between meshes. Instead, in solid and void regions, the density is exactly one or zero, respectively. We can then afford for a coarser mesh without loss of information.

2 | DESCRIPTION OF THE DENSITY GRADIENT-BASED ANALYSIS MESH

Let us now formalize our approach more precisely. In this regard, let us consider a topology optimization problem in some domain Ω of boundary Γ. As density filtering or additional constraints would not affect the following analysis, let us focus on a simplified form of the problem, where we only have the objective functional and the state equation. Thus, in variational form, we consider the problem

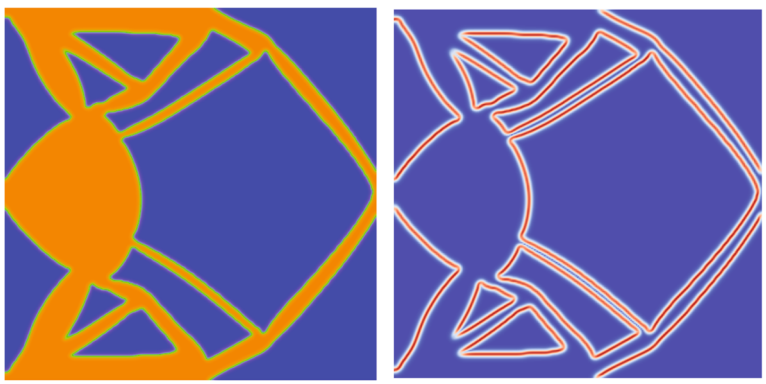


FIGURE 3 A design and the corresponding gradient of density. The gradient is zero in solid and void regions, while it is nonzero in topologically active regions

$$\text{minimize} \quad J(\boldsymbol{u}), \tag{2}$$

$$\text{s.t.} \quad a(\boldsymbol{u}, \tilde{\boldsymbol{u}}) = l(\tilde{\boldsymbol{u}}), \quad \forall \tilde{\boldsymbol{u}} \in \tilde{\mathcal{V}}^d, \tag{3}$$

which is often supplied with also a volume constraint and where $\gamma \in [0, 1]$ is the design variable (possibly filtered), $J(\boldsymbol{u})$ is the objective functional, $a(\boldsymbol{u}, \tilde{\boldsymbol{u}})$ is a bilinear form, $l(\tilde{\boldsymbol{u}})$ is the linear form, and $\tilde{\mathcal{V}}^d$ denotes the space of the test functions for a d-dimensional domain. More precisely, we define the following trial and test spaces.

$$\text{Scalar trial space:} \quad Q = \{v \in H^1(\Omega) : v = v_0 \ \text{ on } \ \Gamma_D\}, \tag{4}$$

$$\text{Scalar test space:} \quad \tilde{Q} = \{\tilde{v} \in H^1(\Omega) : \tilde{v} = 0 \ \text{ on } \ \Gamma_D\}, \tag{5}$$

$$\text{Vector trial space:} \quad \mathcal{V}^d = \{\boldsymbol{v} \in [H^1(\Omega)]^d : \boldsymbol{v} = \boldsymbol{v}_0 \ \text{ on } \ \Gamma_D\}, \tag{6}$$

$$\text{Vector test space:} \quad \tilde{\mathcal{V}}^d = \{\tilde{\boldsymbol{v}} \in [H^1(\Omega)]^d : \tilde{\boldsymbol{v}} = \boldsymbol{0} \ \text{ on } \ \Gamma_D\}, \tag{7}$$

where d is the dimension of the domain, Γ_D denotes the parts of the boundary where Dirichlet boundary conditions hold, and v_0 and $\boldsymbol{v}_0$ are some given boundary values.

Typically, a topology optimization problem of this kind is first discretized by finite element schemes and is then solved iteratively. The iterative solution of the problem consists, at each iteration, in computing the solution of the primal problem, which allows to compute the cost. The adjoint problem is then solved, and this, together with the primal solution, allows to compute the cost sensitivity. The sensitivity can finally be used to compute the new density distribution by some optimizer, such as optimality criteria or the method of moving asymptotes (MMA).[31] We here assume that the representation of densities is nodal based and that the discretization is performed with elements of order, at least, one.

In this procedure, we can identify two different parts:

- a design step, which involves the computation of the sensitivity and of the new density γ. If also other constraints are present, their sensitivities are here computed as well;
- an analysis step, which consists in computing the primal and adjoint solutions of the state equation.

The analysis step, in particular, can be very onerous. Indeed, the solution of the primal and adjoint problems must be carried out at each iteration, and may involve large vectors and matrices, depending on the discretization. Furthermore, while we may need high mesh resolution in the design in order to represent fine features, in the analysis it may be enough to compute an approximation of the primal and adjoint solutions that is sufficiently accurate to allow the correct detection of any topological change. We then aim at increasing the efficiency of the analysis by distinguishing between a fine design mesh and an adaptive analysis mesh, which we build as described in the following.

2.1 | Building the analysis mesh

At any iteration it of the topology optimization procedure, let the design be given in a uniform design mesh of resolution r_d. Furthermore, let it_{ref} be a parameter that represents the iteration after which the adaptive mesh is started*. When $it \geq it_{ref}$, we build the density gradient-based analysis mesh by first initializing it as a uniform mesh of constant resolution r_{a0}. We are going to apply successive density gradient-based refinements to this mesh, and, as discussed above, we want that the final resolution of the analysis mesh is the same as that of the uniform design mesh. Therefore, denoting by n_{ref} the number of adaptive refinements that we want to perform and by k the value by which the resolution increases in the elements that are refined after one refinement, we define r_{a0} as

$$r_{a0} = \frac{r_d}{k^{n_{ref}}}, \tag{8}$$

For instance, if, by the chosen refinement technique, the resolution is doubled in an element that is refined, (8) holds with $k = 2$.

For discrete variational problems in each of these meshes, we formulate finite-dimensional test and trial spaces that are contained in the spaces in (4)–(7). In particular, we define

$$\begin{aligned} \mathcal{Q}_{a0} \subset \mathcal{Q} \quad &\rightarrow \quad \text{Discrete trial space in the starting analysis mesh} \\ \tilde{\mathcal{Q}}_{a0} \subset \tilde{\mathcal{Q}} \quad &\rightarrow \quad \text{Discrete test space in the starting analysis mesh} \\ \mathcal{Q}_{d} \subset \mathcal{Q} \quad &\rightarrow \quad \text{Discrete trial space in the design mesh} \\ \tilde{\mathcal{Q}}_{d} \subset \tilde{\mathcal{Q}} \quad &\rightarrow \quad \text{Discrete test space in the design mesh} \end{aligned} \tag{9}$$

and we denote analogously the corresponding vector spaces $\mathcal{V}^d_{a0}, \tilde{\mathcal{V}}^d_{a0}, \mathcal{V}^d_d, \tilde{\mathcal{V}}^d_d$.

Next, we use (1) to determine whether an element belongs to a topologically active region and we refine only these elements in the analysis mesh. Thus, in order to compute $\nabla\gamma$ in the analysis mesh, we first need to either project or interpolate the density from the design onto the initial analysis mesh. In particular, if we use a projection, we look for a $\gamma_{a0} \in \mathcal{Q}_{a0}$ such that

$$a(\gamma_{a0}, \tilde{\gamma}_a) = \ell(\tilde{\gamma}_a) \quad \forall \tilde{\gamma}_a \in \tilde{\mathcal{Q}}_{a0}, \tag{10}$$

where

$$a(\gamma_{a0}, \tilde{\gamma}_a) = \int_\Omega \gamma_{a0} \cdot \tilde{\gamma}_a \, \mathrm{d}x \quad \text{and} \quad \ell(\tilde{\gamma}_a) = \int_\Omega \gamma \cdot \tilde{\gamma}_a \, \mathrm{d}x. \tag{11}$$

For projections, there are no essential boundary conditions, and $\mathcal{Q}_{a0}, \tilde{\mathcal{Q}}_{a0}$ are defined without any Dirichlet boundary Γ_D. For interpolations, we would proceed similarly and interpolate γ onto the initial analysis mesh.

We can now compute the gradient of γ_{a0}, in order to determine which elements of the analysis mesh need to be refined according to the topology change criterion (1). In this regard, in practice, for numerical reasons we may have small, nonzero values also where the gradient should be exactly zero. Therefore, in our implementation we employ a tolerance ϵ_{ref}, which must be selected to be sufficiently close to zero. Thus, we refine the elements in accordance with the following criterion:

$$\begin{aligned} &\text{if} \quad \left|\frac{\partial \gamma_{a0}}{\partial x_i}\right| \geq \epsilon_{ref} \ \text{ in an element for some } \ i = 1, \dots, d \quad \rightarrow \quad \text{refine the analysis mesh in that element,} \\ &\text{if} \quad \left|\frac{\partial \gamma_{a0}}{\partial x_i}\right| < \epsilon_{ref} \ \text{ in an element } \ \forall i = 1, \dots, d \quad \rightarrow \quad \text{do not refine the analysis mesh in that element.} \end{aligned} \tag{12}$$

Evidently, we could equivalently use some norm of $\nabla\gamma_{a0}$ or any other measure that allows to detect elements where $\nabla\gamma_{a0}$ has a non-negligible magnitude.

At this point, we have performed one density gradient-based adaptive refinement, and the resolution of the newly computed mesh is r_{a1}. If $n_{ref} = 1$, we have obtained our final analysis mesh, and we set $r_a = r_{a1}$. If, instead, $n_{ref} > 1$, we proceed with additional refinements. Thus, by (10)–(11), we project/interpolate γ from the design onto the analysis mesh obtained after the first refinement and we perform another gradient-based refinement in accordance with the criterion (12). We then proceed analogously until we have performed a total of n_{ref} refinements.

Notice that (8) implies that, after performing n_{ref} refinements, we have $r_a = r_d$ in all topologically active regions.

Remark 1. It is worth noticing that the magnitude of the gradients tends to change during the optimization process. Indeed, the design is less sharp at earlier iterations and, therefore, the magnitude of the derivatives is generally smaller. Conversely, at later iterations, the design is sharper and the density derivatives are larger. Therefore, it is generally suitable to modify ϵ_{ref} by continuation methods during the optimization process. In this regard, we will use a small ϵ_{ref} at earlier iterations, in order to detect all zones where intermediate densities are present. As the optimization progresses, we will then increase ϵ_{ref} to avoid unnecessary refinements where the gradients are numerically zero . Nonetheless, the selection of ϵ_{ref} is not problematic, as the results are not very sensitive to its choice (provided that ϵ_{ref} is sufficiently small to detect the nonzero gradients).

Remark 2. Provided that the degree of the interpolation is sufficiently high, the results of the procedure arguably do not change significantly depending on whether projections or interpolations are used, or on which interpolation functions are chosen. For instance, in the numerical experiments we are going to use projections or interpolations depending on what is more convenient to implement (see Section 5.1 for more information), without inconsistencies in the results.

Instead, issues may evidently arise if the degree of the interpolation is too small to correctly reproduce the forms that are interpolated.

Remark 3. For simplicity, we have here described an implementation where we start from a coarse initial analysis mesh and we obtain the final analysis mesh by a sequence of adaptive refinements. However, we can also equivalently start from a fine and uniform mesh with $r_{a0} = r_d$ and apply a sequence of adaptive coarsenings. In this case, at each coarsening step, we would coarsen every element where $|\frac{\partial \gamma_a}{\partial x_i}| < \epsilon_{ref}\ \forall i = 1, \dots, d$, while leaving all the other elements at resolution r_d.

Remark 4. Finally, we remark that it is possible to modify the procedure to refine also other elements that are important for the analysis. This could be, for instance, the case of the elements containing the nodes where loads and boundary conditions are applied. In practice, however, in our experiments we did not find it to be necessary. Therefore, when not otherwise specified, we have employed only the density gradient-based criterion to govern the refinements.

2.2 | Solution of the primal and adjoint problems and projection of the results

Once we have obtained the final analysis mesh, we perform a last projection/interpolation of γ by (10)–(11). We call γ_a the representation of the density function in the final analysis mesh. We can then compute the primal and adjoint solutions in the analysis mesh, which we denote, respectively, by $\boldsymbol{u}_a \in \mathcal{V}_a^d$ and $\boldsymbol{w}_a \in \mathcal{V}_a^d$, where the function spaces are defined with the essential boundary conditions of the state problem.

These solutions are needed to compute the sensitivity of the cost function. Nonetheless, they are defined in the analysis mesh, while we need the sensitivity in the design mesh. Indeed, the sensitivity needs to be passed to the optimizer to compute the new density distribution in the design mesh. Therefore, we again need a map between density and analysis meshes (e.g., using projections/interpolations or by applying the chain rule in the computation of sensitivities).

Nonetheless, these passages are generally required by the very idea of multiresolution TO, and are common to several approaches. Thus, contrarily to the projections and interpolations that, in the previous subsection, we need to perform the density gradient-based refinements, they are not characteristic to our mesh and their study is then beyond the scope of this article. Later, we will then limit ourselves to briefly discuss (in Section 4.3) how these projections and interpolations benefit from having $r_a = r_d$ at solid-void interfaces, as ensured by our gradient-based mesh.

2.3 | Avoiding unnecessary remeshing by evaluating the change of density distribution

The previous two sections contain the core of the proposed gradient-based adaptive meshing. Nonetheless, some improvements can be made to optimize its efficiency.

In particular, in the previous setting, the remeshing is done at each iteration, but this is not always necessary. For instance, if the design does not change, proceeding as in Section 2.1 will produce exactly the same analysis mesh. Furthermore, in practice, topologically active regions will still be in a high-resolution mesh also if we do not update the analysis mesh when there are small changes to the design. Indeed, buffer regions are present around the elements flagged for refinement in order to pass gradually from fine to coarse mesh and improve the conditioning. This is particularly significant at the later stages of the optimization process, when the design is sharp and does not change much between one iteration and the other.

In order to exploit this fact, we can evaluate how much the design has changed to decide whether we need to remesh or not. In this regard, we can evaluate the change of density distribution inexpensively by simply considering differences between density distributions. Thus, let $\gamma^{(k)}$ be the density at some iteration k and let $\gamma^{(old)}$ be the density distribution at the last iteration when the analysis mesh was updated. We can evaluate how $\gamma^{(k)}$ differs from $\gamma^{(old)}$ by computing the measure of the change of density distribution

$$M_{cd} := \frac{||\gamma^{(k)} - \gamma^{(old)}||}{||\gamma^{(old)}||}, \tag{13}$$

where $||\cdot||$ denotes the Euclidean norm. The normalization by $\gamma^{(old)}$ is performed to make M_{cd} more independent of the problem. Indeed, in absence of any normalization, the value of M_{cd} would change when the size or other characteristics

of the problem are changed. The normalization with respect to $\gamma^{(old)}$ relates, instead, the change to the old density distribution, making M_{cd} less dependent on factors such as the volume fraction or the dimension of the problem. Nonetheless, if the change of the volume fraction is a concern, a more conservative normalization can be defined by replacing the denominator of (13) by $\| \min\{\gamma^{(old)}, \gamma^{(k)}\} \|$, where the minimum operator is meant to apply component-wise.

If the measure of the change of density distribution satisfies

$$M_{cd} < \epsilon_{cd}, \tag{14}$$

where ϵ_{cd} is a given tolerance, the analysis mesh is not updated and the computations of Section 2.1 need not be performed. If, instead, $M_{cd} \geq \epsilon_{cd}$, we update the analysis mesh and we set $\gamma^{(old)} = \gamma^{(k)}$.

In the following, when not otherwise specified, it is assumed that we are using (14) to regulate remeshing.

2.4 | Algorithm

We finally summarize the above-described procedures by two algorithms. In particular, in Algorithm 1 we describe how the density gradient-based mesh is built at a given iteration. In the algorithm, n_{ai} denotes the number of elements of the analysis mesh after i refinements, $i = 0, \dots, n_{ref}$. In Algorithm 2, we then describe how the computed mesh fits into the iteration of a topology optimization process.

Algorithm 1. Density gradient-based analysis mesh

Require: density γ, number refinements n_{ref}, initial resolution analysis mesh r_{a0}, tolerance ϵ_{ref}

Build uniform mesh of resolution r_{a0}
for $i = 1, \dots, n_{ref}$ **do**
  Compute γ_{ai-1} projecting (or interpolating) γ from the design mesh onto the analysis mesh at the previous refinement
  Compute the gradient of γ_{ai-1}
  for $j = 1, \dots, n_{ai-1}$ **do**
    Compare the derivatives of γ_{ai-1} and ϵ_{ref} according to (12)
    If refinement criterion is satisfied, flag element j for refinement
  end for
  Refine mesh in all flagged elements
end for

Algorithm 2. One iteration of the optimization process with density gradient-based analysis mesh

Require: starting iteration for adaptiveness it_{ref}, tolerance ϵ_{cd}

if $it < it_{ref}$ **then**
  Solve analysis in the design mesh
else if $it \geq it_{ref}$ **then**
  Compute M_{cd} as in (13), or set $M_{cd} = \epsilon_{cd}$ if $it = it_{ref}$
  if $M_{cd} \geq \epsilon_{cd}$ **then**
    Compute density gradient-based analysis mesh in accordance to Algorithm 1
    Project/interpolate the density from the design mesh onto the density gradient-based analysis mesh
    Solve analysis in the adaptive analysis mesh
    Compute sensitivities, mapping them onto the density mesh
  end if
end if
Pass sensitivities to an optimizer and compute the new density distribution

3 | ANALYSIS OF THE DENSITY GRADIENT-BASED ADAPTIVE MESH

3.1 | Adaptiveness

The proposed mesh adapts to the topology both spatially (given a design, it is fine only where it is needed) and temporally (it changes as the design evolves during the iterations of the optimization process).

In particular, Figure 4 shows how the mesh exploits the information of the density gradient, achieving a spatial adaptiveness. In particular, starting from the design and the initial uniform mesh of Figure 4A, we observe, for each refinement $i = 1, \dots, n_{ref}$,

- the density gradient $\nabla\gamma_{ai-1}$;
- the elements that are flagged to be refined (based on $\nabla\gamma_{ai-1}$), which are in red. The elements that will not be refined are in blue. The mesh (in yellow) is still the one at the previous refinement, $i - 1$;
- the resulting density gradient-based analysis mesh, after the marked elements have been refined.

Considering many refinements, at the beginning the analysis mesh is not able to accurately detect the design and the density gradient. In this case, the projected density tends to change in almost all neighboring nodes, and the corresponding gradient is zero only in very large solid and void regions. In all other zones, the mesh is refined. This also ensures that, if the initial analysis mesh is too coarse to represent the design, we will eventually reach a sufficiently fine analysis mesh. At most, the first refinements will be almost useless to the extent of increasing efficiency (but they will also be very inexpensive, as they involve very coarse meshes).

Then, the representations of the density and of the gradient become progressively more precise, as more gradient-based refinements are performed. This eventually leads to a mesh that is fine only in the neighborhood of elements with $\nabla\gamma \neq 0$.

It is also worth noticing that the analysis mesh is not refined symmetrically in Figure 4. This is particularly evident at early refinements and in buffer zones between parts of the mesh characterized by different resolutions. Nonetheless, the proposed adaptive mesh is not expected to introduce significant nonsymmetrical features in the optimized design. Indeed, we notice that the regions containing solid-void interfaces in the design of Figure 4A are refined at all refinements. Therefore, the final analysis mesh in these zones has a resolution $r_a = r_d$, that is, the final resolution remains as

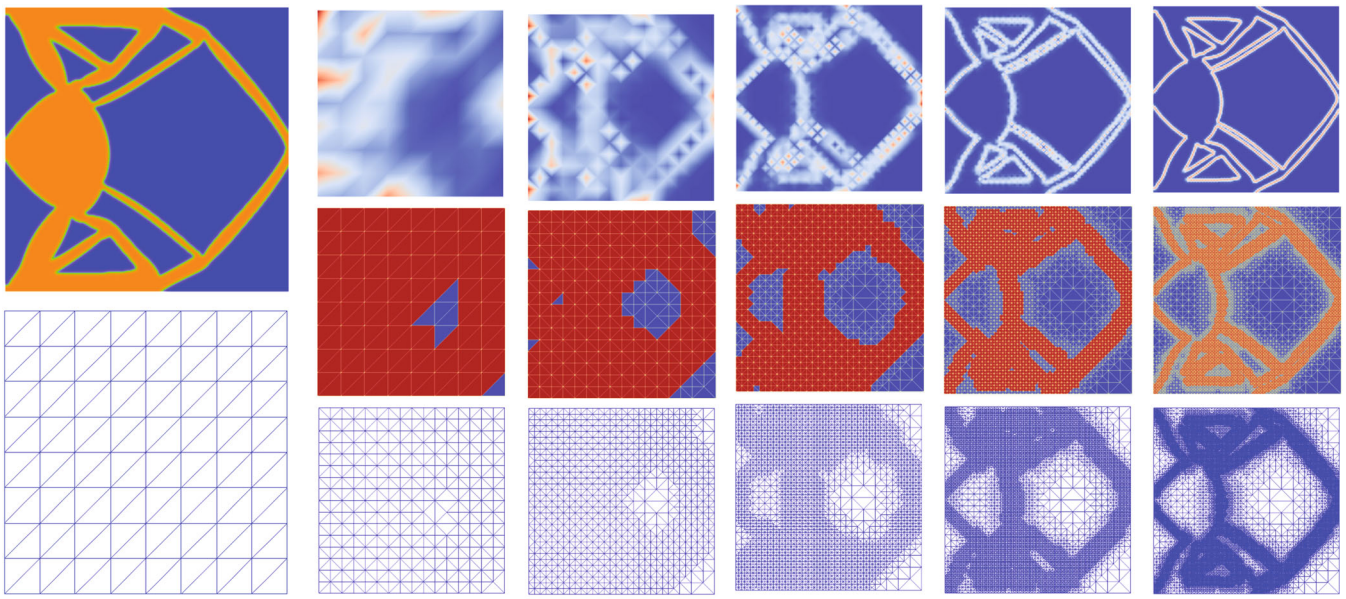


FIGURE 4 Spatial adaptivity: density gradient, marked elements to refine (red), and density gradient-based analysis mesh at various refinements

symmetrical as the design along these interfaces. The regions where the final resolution is not symmetrical are instead characterized by solid or void densities, which, based on the considerations of Section 1, are not topologically active and less critical for the computation of the optimized design. Numerical experiments in later sections confirm that the final result does not change significantly when the adaptive mesh is employed.

Regarding the temporal adaptiveness, the adaptive mesh is updated with the design, which, in turn, is modified by the topology optimization. In particular, this temporal adaptiveness is shown in Figure 5, which shows how the mesh evolves with the design during the optimization.

Figure 5A shows the design and the analysis mesh at an early iteration of the optimization process. The design is still gray, and the mesh is fine in the great part of the domain. Nonetheless, we can already see that the mesh is coarser in the zones where large void and full regions start to stabilize. The fine mesh in gray regions allows a sufficiently accurate computation of the solutions of the analysis problems. Thus, grayness is progressively reduced in the following iterations, and is replaced by finer features that gradually form. Indeed, in Figure 5B–D, we can see that the zones where the analysis mesh is fine become progressively smaller and "pressed" against the solid-void interfaces of the design. During this process, the analysis mesh is updated whenever the design changes enough, in accordance with (14). This is sufficient to detect possible changes of density distribution and to reduce the number of elements in topologically stable zones.

3.2 | Formation of holes and of new solid regions

In this context, it is particularly interesting to focus on the nucleation of holes and of new solid regions. Indeed, it is important to ensure that these features can be created and modified when the analysis is performed in the adaptive mesh.

Figure 6 represents a zoom-in of the same upper part of the structure in Figure 5 at different iterations. Holes are progressively nucleated from a gray region. Simultaneously, new solid regions appear, and the solid-void interface evolves analogously, moving toward previously void or solid zones.

For instance, between Figure 6B,C, on the right of the design a branch of the structure moves leftwards, and previously void regions pass to solid. At the same time, on the left of the domain, holes, and new structures are nucleated from

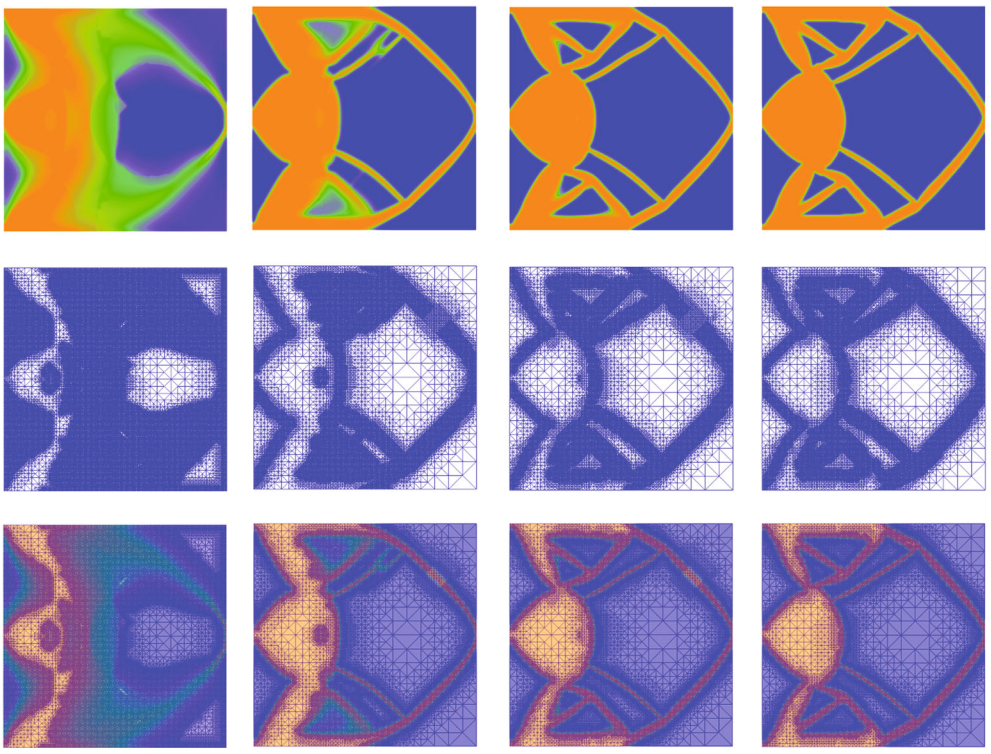


FIGURE 5 Temporal adaptivity: topological evolution of the design and the corresponding adaptive mesh as the optimization progresses

the grayness of Figure 6A. In particular, holes become progressively larger and sharper. Consequently, the analysis mesh changes, following the evolution of the topology. For instance, while, at iteration 25 (Figure 6A), almost the entire region is characterized by grayness and a fine analysis mesh, at iteration 100 (Figure 6C) the mesh coarser meshes in the formed holes and solid areas.

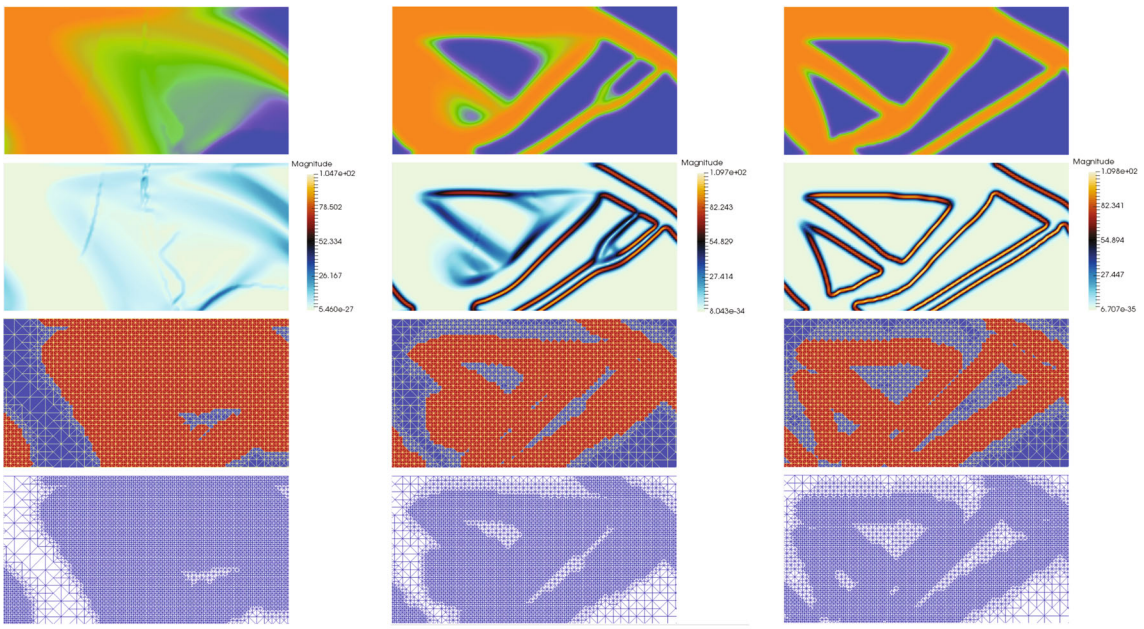
3.3 | Analysis of projection of density in solid and void regions

In addition to the "topological" reason behind the idea of setting $r_a = r_d$ where the gradient of density is nonzero, there is also a mathematical reason related to the necessity to relate the design and analysis meshes by projections or interpolations.

In particular, projections/interpolations of the density onto the analysis mesh are necessary when the analysis is performed on a mesh different from the design. When the projection involves void or solid regions, it is easy to verify that we do not lose any information on the density function even if the analysis mesh is significantly coarser. Indeed, in these regions, the density is constant and equal, in every node, to zero (in void) or to one (in solid). This fact is irrespective of whether the mesh is fine or coarse. Instead, this is not true where the gradient of density is different from zero. Indeed, in these regions, the value of the density function changes and the projection onto a coarser mesh would imply the loss of information on the value of γ.

For instance, Figure 7 reports an example of what happens when a design represented in a uniform mesh of resolution $r = 256$ is interpolated onto a coarser uniform mesh with $r = 32$. We notice that solid and void regions are still represented exactly in the coarser mesh, while the solid-void interfaces appear significantly rougher. It is also interesting to notice the presence of QR-patterns (i.e., numerical artifacts that consist of disconnected features with artificially high stiffness) in several regions of the design represented in a coarser mesh. This issue is common in multiresolution topology optimization and has been recently studied in Reference 32, where it was attributed to the limitations of the used modeling scheme (extending also earlier considerations reported in Reference 22). In particular, in Figure 7 the QR-patterns arise because the mesh is too coarse to represent without disconnections the thinnest features of the design.

In Figure 8 we show that the loss of information at solid-void interfaces is eliminated if the resolution of the two meshes is the same where $\nabla\gamma \neq 0$. Evidently, if the mesh is coarse along the interface, much information on γ is lost along the entire boundary, as in the top-right figure. When, instead, the analysis mesh has the same resolution as the design

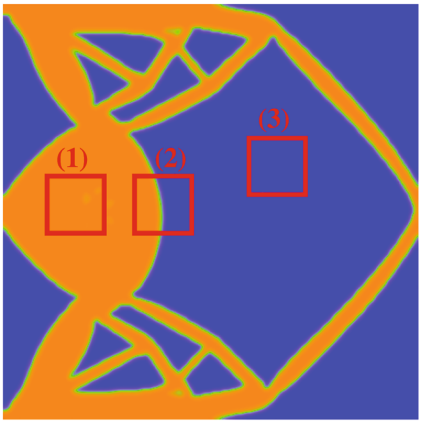


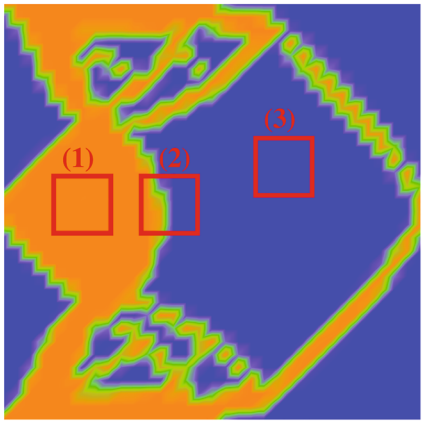
(A) Iteration 25 (B) Iteration 50 (C) Iteration 100

FIGURE 6 Nucleation of holes and of new solid regions from a gray area. From top to bottom: design, density gradient, elements marked to be refined at the last density gradient-based refinement (red), final analysis mesh

(A) Design in uniform mesh, resolution 256

(B) Design interpolated onto uniform mesh of resolution 32





(C) Regions with $\nabla\gamma = 0$ have constant density, and can be represented exactly in the coarse mesh (e.g. solid and void regions marked by (1) and (3)). At solid-void interfaces (e.g. region (2)), coarser mesh implies loss of information in the interpolation

FIGURE 7 Loss of information in projections and interpolations of density onto a coarser mesh

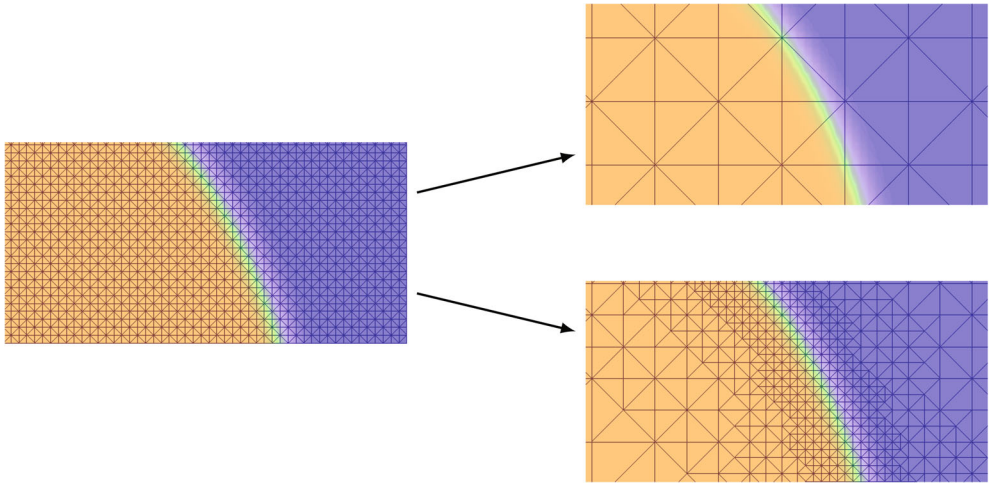


FIGURE 8 The projection of the density onto a coarse analysis mesh leads to loss of information if $r_a < r_d$ at solid void boundaries (top-right), while it preserves the information if $r_a = r_d$ at solid void boundaries (bottom-right)

TABLE 1 Error estimate (15) of the solution of the primal problem in adaptive and uniform meshes

	$n_{ref} = 1$	$n_{ref} = 2$	$n_{ref} = 3$	$n_{ref} = 4$	$n_{ref} = 4$	Uniform
$\|\|err\|\|_1$	1.0320e-4	1.0354e-4	1.0354e-4	1.0356e-4	1.0356e-4	1.0388e-4

at the interface, we have the same nodes where the density changes. Thus, the density gradient-based mesh with $r_a = r_d$ at solid-void interfaces preserves all the information contained in the design mesh when we project/interpolate onto the analysis mesh. This also helps solve the issue of QR-patterns: indeed, if the resolution is sufficiently high along solid-void interfaces (and, consequently, in thin features), we can avoid misrepresentations and disconnections in the features of the design. We further discuss this in Section 5.2, where we also comment on the maximum coarsening that we can perform on the initial analysis mesh.

Regarding the mapping of sensitivities from the analysis to the design mesh, the advantage regards the fact that we have a direct mapping between sensitivities in all elements where the density gradient is nonzero. Indeed, having $r_a = r_d$ in these regions, we can map every topologically active node of the analysis mesh to exactly one node in the design mesh.

3.4 | A posteriori error analysis

Finally, we employ an a posteriori error analysis to estimate the error of the analysis performed in the adaptive mesh and to compare it with the error in the uniform mesh. In this regard, we estimate the error as

$$e_T = (h_T \|res\|_{L_2(T)})^2, \tag{15}$$

where h is a representative size for an element T and res is the residual. In practice, first we compute u by solving the primal problem, while h_T is the diameter of the cell in the considered mesh (the diameter of a cell T is to be intended as $\sup_{x,y\in T} |x - y|$). Alternative estimates that employ the total residual or that consider also other contributions coming from errors on edges and faces behaved similarly.

Given a design, let us compute this error estimate when the analysis is solved in density gradient-based adaptive meshes with different number of refinements n_{ref}. We compare the results with the error estimate obtained by solving the analysis in a uniform mesh of the same resolution as the design mesh. This case is denoted as "uniform" in the following. In this regard, we did not use directly the design mesh, but we refined a starting uniform mesh everywhere to a resolution r_d. This way, we ensure that we are considering only the effect of the density gradient-based adaptiveness in our comparisons. In all cases, the primal problem is solved by the same direct solver.

The error estimates for the solution of the primal problem for the design of Figure 1A are given in Table 1, which reports the numerical values of the sum of the elemental errors in the entire design, $\|err\|_1$. These results show that the error does not change significantly in the different cases, confirming that the density gradient-based meshes allow to compute the solution of the primal problem in a sufficiently accurate way, irrespective of n_{ref}. Figure 9 then shows where the error is larger for $n_{ref} = 4$ and for the uniform case. The errors are almost identical, although some differences can be noted close to application points of loads and boundary conditions. This is reasonable, since these are points where an accurate analysis is evidently important, but they can also be characterized by a coarse analysis mesh when they are far from the solid-void interfaces. In order to reduce these differences and as already noticed in Remark 4, it is possible to refine the adaptive mesh also in these points, if needed. However, this was not required in our experiments, where the errors in uniform and adaptive meshes did not differ significantly and never led to the computation of different topologies when the adaptive mesh was used.

Finally, Figure 9 also shows that the largest errors are concentrated at solid-void interfaces both in uniform and in adaptive meshes. This confirms the necessity of high resolution at solid-void interfaces and that it is reasonable to use coarser discretizations in the regions that are not topologically active.

Proceeding with the finite-element analysis, in a topology optimization problem we will then have to compute the solution of the adjoint equation and the cost sensitivity. Finally, we will have to project the result onto the design mesh. To verify that also these steps are performed correctly, Figure 10 represents the projected sensitivity computed in the uniform mesh for various choices of n_{ref}. The color-bar has been scaled to improve the readability, but no postprocessing was performed on the data.

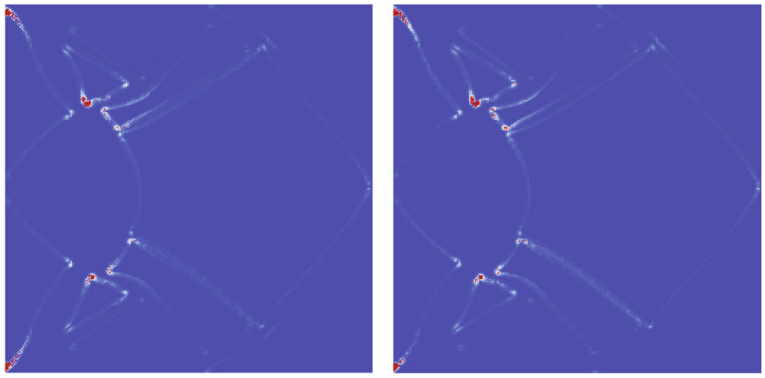


FIGURE 9 Error in uniform mesh (left) versus error with $n_{ref} = 4$ (right), rescaled (in the same range) for visibility

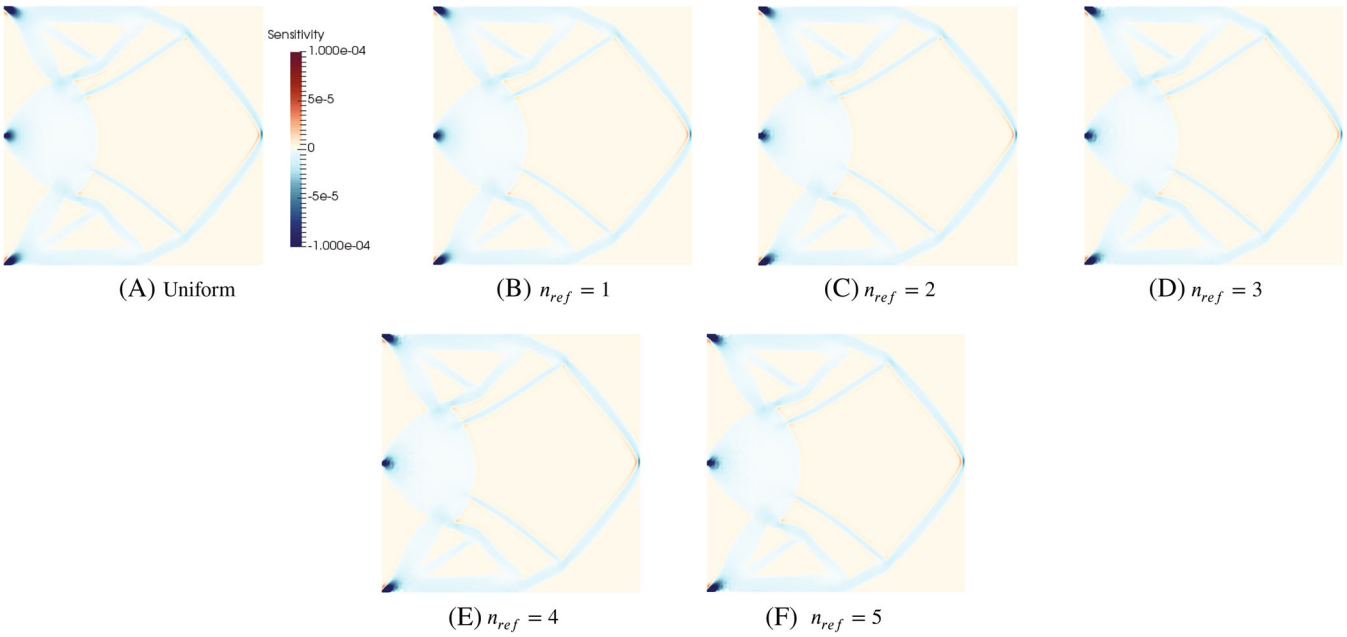


FIGURE 10 Plot of the cost sensitivity (projected onto the same design mesh) in density gradient-based and uniform meshes

In all cases, the computed sensitivities are very similar. Indeed, the sign of the sensitivity is the same everywhere in the domain and the values are always indistinguishable in the figure. Analogously to the error distribution, also for sensitivities we can nonetheless notice some differences between uniform and analysis meshes in a few solid regions close to the application points of the loads and boundary conditions. In particular, the sensitivity there appears as if it were at a lower resolution in the adaptive mesh, especially when multiple refinements are performed. However, these differences involve only limited zones where the design is not topologically active, and the signs and the overall values of the sensitivity are the same as in the uniform mesh. Therefore, we can expect that these differences do not lead to changes of the optimized topology. We later show that this is indeed what happens in the numerical experiments.

4 | ANALYSIS OF THE COMPUTATIONAL SAVING

In this section, we analyze and estimate the expected computational saving when the density gradient-based mesh is used. In this regard, we are particularly interested in estimating how many elements we can save and how many refinements it is reasonable to make to maximize the efficiency. Thus, first we find theoretical bounds of the minimum number of elements of our analysis mesh. Then, we use these bounds to perform some considerations and formulate a more precise

heuristic estimate. Finally, we provide some numerical results that link the saving in terms of elements to the saving in terms of computational times.

4.1 | Maximum theoretical saving of elements in limit cases for a 2D mesh

The final analysis resolution has two limit thresholds:

- $r_a = r_{a0}$ everywhere if the entire design has constant density (e.g., if it is all solid or all void). Indeed, in this case, $\nabla\gamma_{ai} = \mathbf{0}$ everywhere at each refinement;
- $r_a = r_d$ everywhere if the entire design is made of regions where γ is never constant. Indeed, in this case, $\nabla\gamma_{ai} \neq \mathbf{0}$ everywhere at each refinement.

In a typical topology optimization design, we are in between these situations. Nonetheless, these limit cases give us some information on the maximum computational saving that we may expect, and lead to some considerations on how many refinements it is reasonable to perform.

Let us then analyze them, considering, for the moment, a 2D mesh of triangular elements in a square domain. Assume, for simplicity, that every refinement doubles the resolution, like in Figure 11. If the entire mesh were refined, the number of elements, which we denote by n, would quadruple, as represented in the figure. This is due to the fact that, along every axial dimension, the number of discretization intervals is doubled, so that n increases by $2 \cdot 2 = 4$.

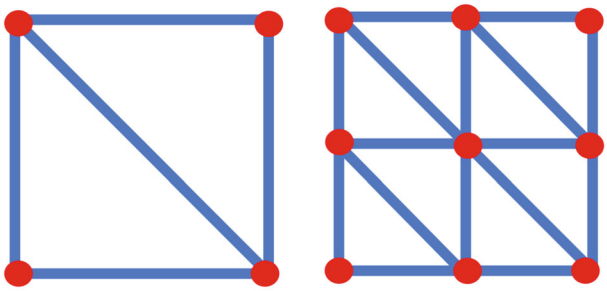
More in general, a density gradient-based adaptive analysis mesh is initialized by a uniform mesh as in (8). If the entire mesh were refined, repeating the same considerations as above, the number of elements would increase by a factor k^2. Furthermore, since each element has three nodes, each of which is shared by six elements in inner parts of the mesh and by fewer elements along the boundaries of the domain, the number of nodes is $n_n > 1/2n$. The inequality approaches the equality for large meshes, where the effect of the boundary nodes becomes negligible. Finally, considering a 2D mesh and nodal-based optimization, we have two DOFs in each node. The total number of DOFs is, then, $n_{DOF} = 2n_n$.

Therefore, if we have a mesh that is refined everywhere n_{ref} times starting from an initial resolution r_{a0}, the number of elements, nodes, and DOFs can be estimated as follows:

$$\begin{aligned} n &= k_g r_{a0}^2 k^{2n_{ref}}, \\ n_n &\approx \frac{1}{2} n, \\ n_{DOF} &\approx n, \end{aligned} \tag{16}$$

where k_g is a parameter that relates the resolution r_{a0} to the geometry of the domain (for instance, if the domain is a square of side 1, $k_g = 2$, as the domain is split into two elements if $r_{a0} = 1$, like in Figure 11A).

By the choice of r_{a0} as in (8), the numbers of elements, nodes, and DOFs in Equation 16 in an analysis mesh that is refined everywhere coincide with the uniform design mesh. Thus, the values in (16) describe n in the limit case $r_a = r_d$.



(A) Elemets before refinement. (B) Resolution after 1 refinement.

FIGURE 11 Example of a refinement that doubles the resolution

The opposite limit case $r_a = r_{a0}$ can instead be interpreted as the theoretical lower threshold to the number of elements of the analysis mesh, for a given choice of r_{a0} and n_{ref}. In this case, the number of elements, nodes, and DOFs can be computed again by (16) with $n_{ref} = 0$.

Comparing these limit cases, we can determine the maximum computational gain that we may theoretically have for any choice of r_{a0} and n_{ref}. In particular, denoting the minimum number of elements of the analysis mesh by $\hat{n}_a$ and the number of elements of the design mesh by n_d, the maximum theoretical saving in terms of elements (and, hence, DOFs) that can be obtained by the proposed density gradient-based technique is

$$\hat{n}_a = \frac{1}{k^{2n_{ref}}} n_d. \tag{17}$$

Equation 17 suggests that, in most cases, 3–4 refinements are enough to achieve most of the computational savings because the main gains are at the first refinements. As an example, consider a square design mesh with $n_d = 65{,}536$. Table 2 reports the thresholds to the maximum theoretical saving as computed by (17) with $k = 2$.

The maximum saving ratio given by the generic ith refinement can also be computed exactly by

$$\hat{s}_i = (k^2 - 1) \sum_{i=1}^{n_{ref}} \frac{1}{k^{2i}}. \tag{18}$$

Evidently, $\hat{s}_i$ after the first refinements.

4.2 | Estimating the DOF saving in real 2D designs

In practice, however, we will always have some parts of the analysis mesh that are refined. Thus, the analysis of the previous subsection applies directly only to solid and void regions that are sufficiently far from zones with intermediate densities. The saving will then be smaller than the threshold of the previous subsection.

Nonetheless, the general behavior is quite similar. For instance, Figure 12 represents the adaptive analysis meshes obtained for a realistic TO design, considering different choices of n_{ref}. The corresponding results regarding the number of elements of the design, n_d, and of the analysis, n_a, are reported in Table 3, together with the relative saving in terms of elements

$$\delta n := \frac{n_d - n_a}{n_d} \%. \tag{19}$$

We notice that we now have a bound on the minimum number of elements, which is directly linked to the number of elements with nonzero density gradient. The saving is, nonetheless, distributed analogously to the ideal case, with larger savings concentrated at the first refinements. This is reasonable and confirms that performing 3–4 refinements allows to achieve most of the possible savings.

In order to make more precise estimates, we can exploit the formulation of the density gradient-based mesh. In particular, by the definition of our refinement criterion, we will have

TABLE 2 Threshold to saving in terms of number of elements with the density gradient-based adaptive mesh

n_{ref}	n_d	$\hat{n}_a$	% maximum saving
1	65,536	16,384	75%
2	65,536	4096	93.7%
3	65,536	1024	98.4%
4	65,536	256	99.6%

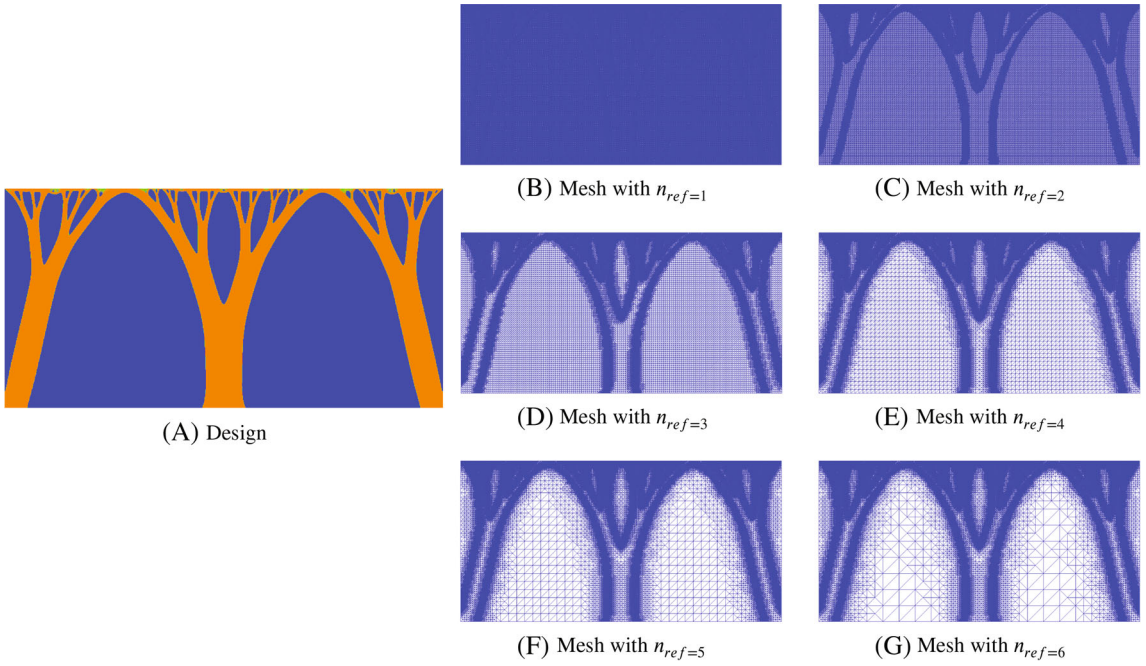


FIGURE 12 An example of the progressive local refinement of the analysis mesh

TABLE 3 Saving in terms of number of elements and nodes with the density gradient-based adaptive mesh on a real design

n_{ref}	n_d	n_a	δn
1	1,638,400	666,027	59.3%
2	1,638,400	450,118	72.5%
3	1,638,400	404,534	75.3%
4	1,638,400	395,148	75.9%
5	1,638,400	393,684	76.0%
6	1,638,400	393,085	76.0%

$$n_a \propto \nabla \gamma_a \tag{20}$$

and the number of elements of the final analysis mesh will increase with the overall perimeter and the amount of intermediate densities. In addition, we will also have a contribution given by the shape of the design and by the refinement technique. Indeed, since the passage from fine to coarse mesh is not abrupt, gray regions that are close to one another will be included in a single high-resolution region, which involves, in general, less elements than the separate high-resolution regions that we would have if the gray zones were far from one another. This latter contribution, however, cannot be evaluated a priori, since it strongly depends on the design and on the extension of the buffer zone of the used refinement technique. Thus, we do not consider it explicitly in our theoretical estimates, but we will include it in later heuristic evaluations. For the moment, we instead focus on the number of elements coming directly from (20).

4.2.1 | Theoretical estimates

Consider a density gradient-based adaptive mesh, built with n_{ref} refinements. By $n_{a0}, n_{a1}, \dots, n_a$ we denote the number of elements of the analysis mesh after $0, 1, \dots, n_{ref}$ refinements, respectively. Analogously, $\gamma_{a0}, \gamma_{a1}, \dots, \gamma_a$ denote the density γ projected onto the analysis mesh after $0, 1, \dots, n_{ref}$ refinements. Furthermore, considering the refinement criterion (12), we define d_i as a mesh function which, in each element, is defined as

$$d_i = \begin{cases} 1 & \text{where } \left|\frac{\partial \gamma_{ai}}{\partial x_j}\right| \geq \epsilon_{ref} \text{ for at least one } j = 1, 2 \\ 0 & \text{elsewhere,} \end{cases} \tag{21}$$

for $i = 0, \dots, n_{ref} - 1$.

Starting from the initial analysis mesh of n_{a0} elements, at each refinement the analysis mesh will be refined at least in every element where $d_i = 1$. Therefore, the least number of elements of the final analysis mesh will be given by

$$\hat{n}_a = n_{a0} + (k^2 - 1)d_0^T e_0 + (k^2 - 1)d_1^T e_1 + \cdots + (k^2 - 1)d_{n_{ref}-1}^T e_{n_{ref}-1}, \tag{22}$$

where $e_0, e_1, \dots, e_{n_{ref}-1}$ are vectors of ones of dimension $n_{a0}, n_{a1}, \dots, n_{a(n_{ref}-1)}$, respectively. In (22), the terms -1 are used to avoid counting twice the elements that are refined.

Since the design does not change during the refinements, we may roughly assume, for the moment, that if an element is refined at some refinement i, then it has already been refined at the previous refinements and it will again be refined at the following. By this assumption, we can link the functions d_i together. For instance, if we further define

$$d_a = \begin{cases} 1 & \text{where } \left|\frac{\partial \gamma_a}{\partial x_j}\right| \geq \epsilon_{ref} \text{ for at least one } j = 1, 2 \\ 0 & \text{elsewhere,} \end{cases} \tag{23}$$

the above consideration leads to

$$d_{n_{ref}-1}^T e_{n_{ref}-1} = d_a^T e_a / k^2 \tag{24}$$

with e_a vector of ones of dimension n_a. Notice that d_a can be computed directly from the design. Indeed, (8) implies that the resolution of the analysis mesh and of the adaptive mesh is the same in the elements where the refinement criterion (12) is satisfied. Thus, d_a is available directly from $\nabla\gamma$. Then, we may assume that the elements that are refined at the last refinement had been refined also at the previous refinement. Therefore, we can further estimate

$$d_{n_{ref}-2}^T e_{n_{ref}-2} = d_{n_{ref}-1}^T e_{n_{ref}-1} / k^2 = d_a^T e_a / (k^2)^2, \tag{25}$$

and so on. Proceeding iteratively and replacing into (22), we obtain

$$\begin{aligned} \hat{n}_a &= n_{a0} + \frac{k^2 - 1}{(k^2)^{n_{ref}}} d_a^T e_a + \frac{k^2 - 1}{(k^2)^{n_{ref}-1}} d_a^T e_a + \cdots + \frac{k^2 - 1}{k^2} d_a^T e_a \\ &= n_{a0} + (k^2 - 1) d_a^T e_a \sum_{i=1}^{n_{ref}} \frac{1}{(k^2)^i}. \end{aligned} \tag{26}$$

This lower bound (26) is well-posed. Indeed, consider, for instance, a square domain of length 1, which is discretized by a design mesh of $r_d = 256$ and by an analysis mesh of $r_{a0} = 16$, with $n_{ref} = 4$ and $k = 2$. By (8), if we have nonzero density gradients in all elements of the design, the estimate is well-posed if $\hat{n}_a = n_d$. This is indeed what happens. Indeed, $n_d = 256^2 = 65{,}536$ and the estimate (26) analogously provides $16^2 + 3 \cdot 256^2 \cdot \left(\frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{1}{256}\right) = 65{,}536$. If, instead, the density gradient is zero everywhere, the bound gives $\hat{n}_a = n_{a0}$, as we would expect.

Furthermore, (26) is available to compute before any refinement is made. Indeed, n_{a0} and n_{ref} are user related choices (linked by (8)), k is known, and d_a can be computed through $\nabla\gamma$. Therefore, we can use (26) as a lower bound of the minimum possible number of elements in the density gradient-based adaptive mesh. This lower bound again confirms that the main savings are at the first refinements. Indeed, the term $\frac{1}{(k^2)^i}$ becomes quickly small as i is increased. The last terms of the sum in (26) will, then, provide a smaller contribution to the element saving.

4.2.2 | Heuristic estimates

However, if a precise estimate is needed, it is necessary to perform additional considerations. In particular, (26) makes the following, significant simplifications.

1. While the number of elements with nonzero density gradient is the same in the design mesh and in the final analysis mesh, we make a simplification when we assume that the same relationship can be propagated iteratively to the previous refinements as in (24)–(26). Indeed, at each refinement, the density is projected/interpolated onto a different mesh. In particular, coarser meshes will tend to have a higher ratio of nodes with nonzero density gradient to total nodes with respect to finer meshes. This happens because coarser meshes must represent complex density distributions in fewer nodes, and the projected density tends, then, to change more frequently from one node to the other. This behavior was observed, for instance, in Figure 4. Thus, (26) tends to underestimate the number of elements in the adaptive mesh.
2. As remarked in the previous subsection, our considerations neglect the fact that elements that are neighbor to elements with nonzero gradient are also refined.

We address these two contributions by heuristic considerations. In particular, as regards the first issue, let us analyze Table 3. As previously noticed, the main savings are concentrated at the first refinements. Thus, we can accept n_a after a few refinements as a practical estimate of the minimum number of elements of the adaptive mesh $\hat{n}_a$. For instance, let us accept $\hat{n}_a$ as n_a at the last row of Table 3 ($n_{ref} = 6$). Let us then compute, for each choice of n_{ref}, the ratio

$$s = \frac{n_d - n_a}{n_d - \hat{n}_a} \tag{27}$$

of the maximum number of elements that we could save. The results are reported in Table 4.

These ratios follow closely the maximum savings of the limit case in Table 2 and Equation 18. Therefore, we can assume that, in a real design, for given n_{ref} and k, we are saving a quantity of elements corresponding to the analytical limit savings (third column of Table 2 for $k = 2$) of the maximum amount of elements that we can save. In this way, we implicitly include all evaluations regarding the refinements and we can compute the saving for any n_{ref} whenever the saving for a single n_{ref} is known.

In this regard, we have already remarked that the number of elements with nonzero density gradient is the same in the design mesh and in the final analysis mesh. Thus, for $n_{ref} = 1$, we can estimate the elements that satisfy the refinement criterion (12) almost exactly, and estimate the number of elements of the corresponding analysis mesh by

$$\hat{n}_{a1} = n_{a0} + \frac{k^2 - 1}{k^2} d_a^T e_a. \tag{28}$$

This estimate can then be propagated to other choices of n_{ref} by requiring that each new refinement provides the same saving (with respect to the maximum possible saving for the considered problem) as expressed by the theoretical ratio (18). Therefore, for a given choice of n_{ref}, we set

TABLE 4 Maximum saving in terms of number of elements with the density gradient-based adaptive mesh on a real design

n_{ref}	n_a	$s\%$
1	666,027	78.1%
2	450,118	95.4%
3	404,534	99.1%
4	395,148	99.8%
5	393,684	99.9%
6	393,085	100.0%

$$\hat{n}_a = n_d - (n_d - \hat{n}_{a1}) \cdot \frac{\hat{s}_{n_{ref}}}{\hat{s}_1}. \tag{29}$$

Finally, regarding the refinement of elements with null density gradient that are close to refined elements, we can multiply the estimate $\hat{n}_{a1}$ by a heuristic parameter $k_h > 1$ that accounts for this effect. We then revise the estimate into

$$\tilde{n}_a = n_d - (n_d - k_h \hat{n}_{a1}) \cdot \frac{\hat{s}_{n_{ref}}}{\hat{s}_1}. \tag{30}$$

The precise value of k_h depends on the design and on the refinement. Nonetheless, in practice, we have found that k_h between 1 and 2 is often a good choice in our experimental setting. All the other terms can be computed by (28) and (18) once the refinement parameters have been chosen and $\nabla\gamma$ has been computed. Thus, (30) can be used a priori, before any refinement is performed.

As an example, Table 5 reports a comparison between the actual number of analysis elements for the 2D design in Figure 12, the lower threshold (26) and the a priori estimate (30) with $k_h = 1.3$.

4.3 | Generalization to 3D

The considerations made with regard to 2D problems apply directly also to 3D cases. The analysis must only be adapted to the 3D setting by revising the above expressions to account for the presence of a third spatial dimension. For instance, consider a unit cube that is packed with six tetrahedra. If the resolution doubles after one refinement, the number of elements n increases by $2^3 = 8$ (as the doubling now occurs along three dimensions).

Thus, if the entire mesh is initialized by a uniform mesh of resolution r_{a0}, the initial mesh contains $n_{a0} = k_g r_{a0}^3$ elements (where, as before, k_g accounts for the geometry of the domain, for example, $k_g = 6$ for the unit cube packed with six tetrahedra). If the entire mesh were refined, at each refinement the number of elements increases by a factor k^3, where k denotes, as before, the increase of the resolution produced by a refinement. Once n is known, it is easy to obtain the number of nodes: indeed, each tetrahedron has four nodes and each cube (which has eight nodes) is packed with six tetrahedra. Therefore, every node of the cube is shared, on average, by $24/8 = 3$ tetrahedra. As an inner node is shared by eight cubes, it is also shared, on average, by $8 \cdot 3 = 24$ tetrahedra, so that $n_n > 4/24n = 1/6n$ (where the inequality is given by the presence of boundary nodes). Finally, each node has three DOFs, and hence $n_{DOFs} = 3n_n$.

Therefore, the expressions in (16) must be replaced by

$$\begin{aligned} n &= k_g r_{a0}^3 k^{3n_{ref}}, \\ n_n &\approx \frac{1}{6} n, \\ n_{DOF} &\approx \frac{1}{2} n. \end{aligned} \tag{31}$$

Similarly, the maximum theoretical saving in terms of elements (and, hence, DOFs) that can be obtained is

TABLE 5 Comparison between the number of analysis elements, the lower threshold (26) and the estimate (30) for the 2D design in Figure 12

n_{ref}	$\hat{n}_a$ as in (26)	$\tilde{n}_a$ as in (30)	Actual n_a
1	541,039	703,350	666,027
2	266,698	469,588	450,118
3	198,113	411,147	404,534
4	180,967	396,537	395,148
5	176,680	392,885	393,684
6	175,609	391,971	393,085

$$\hat{n}_a = \frac{1}{k^{3n_{ref}}} n_d, \tag{32}$$

which replaces (17). Therefore, for a square domain and considering, for instance, $n_d = 65{,}536$ and $k = 2$, the maximum gain after each refinement is given by Table 6, which can be seen as the 3D equivalent of Table 2. We notice that the main savings are even more concentrated at the first refinements.

We can then proceed as in 2D, noticing that, in actual designs, some parts will always be refined, proportionally to the number of elements where the density gradient is nonzero. The considerations made in 2D can then be largely applied with no modification. The main difference is given by the net contribution of each refinement to the saving, by effect of the presence of a third space dimension. In particular, we can replace (18) by

$$\hat{s}_i = (k^3 - 1) \sum_{i=1}^{n_{ref}} \frac{1}{k^{3i}}, \tag{33}$$

while, for $n_{ref} = 1$, we can estimate the elements of the density gradient-based analysis mesh by

$$\hat{n}_{a1} = n_{a0} + \frac{k^3 - 1}{k^3} d_a^T e_a, \tag{34}$$

where d_a is defined analogously to (23) with $j = 1, 2, 3$. Equation (34) replaces (28) for 3D problems. At this point, the heuristic estimates of $\hat{n}_a$ and of the number of analysis elements for an arbitrary number of refinements can still be expressed as in 2D by (29) and (30), respectively, with $\hat{s}_i$ as in (33) and $\hat{n}_{a1}$ as in (34).

It is also possible to adapt the previous analysis to different types of 2D and 3D elements. In this regard, (16) and (31) can be easily revised by analyzing the discretization by the considered finite element. This allows to obtain the maximum theoretical saving in the limit cases for any choice of n_{ref}. The other considerations can then be applied quite directly irrespective of the considered finite element, adapting only the choices of k, k_g, k_h to the chosen discretization.

4.4 | Estimate of computational complexity of various operations

Based on the number of DOFs and on the operations that are performed, we can estimate the computational complexity of running the analysis on a uniform and on a density gradient-based mesh, in order to make a comparison.

In particular, in usual topology optimization settings, the analysis mesh is uniform, with $r_a = r_d$. Thus, we do not have any remeshing to perform. The operations that are performed consist uniquely in computing the primal and adjoint solutions, the cost, and the sensitivity in a mesh of n_d elements.

Instead, when we use an adaptive mesh, we must consider both the computation of the density gradient-based analysis mesh and the cost to run the analysis in the density gradient-based mesh of n_a elements.

4.4.1 | Computation of the analysis mesh

With regard to the adaptive analysis mesh, its construction requires, at every refinement, to project or interpolate the density gradient γ from the design onto the analysis mesh computed by the previous refinement, and then to compute its gradient. Since in each of these projections/interpolations the number of unknowns equals the number of DOFs of

TABLE 6 Threshold to saving in terms of number of elements and nodes with the density gradient-based adaptive mesh

n_{ref}	n_d	$\hat{n}_a$	% maximum saving
1	65,536	8192	87.5%
2	65,536	1024	98.4%
3	65,536	128	99.8%

the analysis mesh at the current refinement, these operations are not expensive compared with the problems to be solved in analysis. Indeed, the maximum number of unknowns will be given by $n_{a,n_{ref}-1}$, which occurs for the problem that is needed to compute the last refinement and obtain the final density gradient-based mesh. This number is not large with respect to n_d, because (as seen in the previous subsections) the number of unknowns tends to decrease sharply even after one refinement.

After we have computed $\nabla\gamma_{ai}$, we need to mark all elements where the gradient exceeds the tolerance. In this regard, we need a loop over all elements of the analysis mesh computed by the previous refinement. Through the refinements, we will then loop over a total of

$$n_{a_{tot}} = n_{a0} + n_{a1} + \cdots + n_{a,n_{ref}-1}$$

indices. At each index of the loop, we will have an *if* statement, which will mark the element, depending on whether $\nabla\gamma$ exceeds the tolerance in the element itself. Beyond this, in actual implementations, there may be other auxiliary computations, such as maps between node and vertex numbering, passage of local indices in case of parallel computing, etc.

Due to these many contributions, it is quite impractical to provide a precise estimate of the computational cost of remeshing. Therefore, we will later analyze numerically the time for building the adaptive mesh in some problems. For the moment, we just notice that, arguably, these computations are not particularly expensive with respect to the complexity of the problem. Indeed, not only each mesh considered in the gradient-based refinements has significantly less variables than the design mesh, but we also do not have to recompute the adaptive mesh at each iteration if we use the measure of change of density distribution (13). Lastly, we remark that the loop and the operations that it contains can be easily parallelized, further reducing the computational times.

4.4.2 | Solution of analysis

Solving the analysis consists of the same operations irrespective of whether we are using an adaptive or a uniform analysis mesh. What changes is the mesh where the operations are performed. In particular, when a density gradient-based analysis mesh is used, all operations are performed in a mesh of n_a nodes, where n_a can be significantly smaller than n_d, especially for sharp designs.

The computational complexity depends on several factors that depend on the choices performed in the implementation (including the solvers that are used to solve the primal and adjoint problems). Nonetheless, the complexity of these operations increases (possibly, more than linearly) with the number of variables. The savings achieved by using an adaptive analysis mesh are concentrated here.

Based on the analysis on the number of elements, the ratio of saved elements with adaptive mesh increases with the size of the problem. Thus, it is reasonable to expect that the efficiency of the adaptive mesh scales well with the problem, and that a good efficiency is obtained especially for large problems. This is substantiated hereafter by considering a 2D test problem. The scalability in terms of the number of unknowns is later studied for several test problems also in Section 5.

4.4.3 | Efficiency comparison with uniform design mesh

We here substantiate the previous considerations by analyzing the computational times required to solve some test problems. The numerical data and the computational setting is described in Section 5.1.

Considering the compliant mechanism with $r_d = 256$, the total times (in seconds) required by the analysis and by the computation of the gradient based mesh with various choices of n_{ref} is reported in Table 7. In the table, t_a denotes the total time required to perform the analysis, while t_m denotes the total time to compute the density gradient-based analysis mesh. This latter time is comprehensive of all the intermediate projections of the density gradient (one per adaptive refinement). The final designs obtained with and without adaptive analysis mesh are the same as in Figure 14. Similar results are obtained with all the other considered choices of n_{ref}.

We notice that the results obtained with density gradient-based analysis mesh require a significantly smaller analysis time, while the optimized design remains similar and does not become less defined. The saving increases with the number of refinements, but the improvement becomes less significant as n_{ref} increases. This is consistent with the previous analysis on the savings in terms of the number of analysis elements. The computed design has, instead, the same resolution and the same fine features of the one computed in the uniform (fine) analysis mesh.

Furthermore, in all cases, the time required to compute the adaptive mesh is quite small with respect to the analysis time. In this context, it is worth noticing that the remeshing time does not increase significantly when more refinements are performed. This follows the fact that every additional refinement adds computations on coarser meshes, and is consistent with what was expected.

Finally, Table 8 analyzes the scalability of the procedure. In this regard, we analyze how t_a and t_m change when r_d is changed, for the same number of adaptive refinements n_{ref}.

Table 8 shows that the density gradient-based adaptive mesh scales well with the dimension of the problem. Indeed, while the computation of the adaptive mesh is not advantageous for smaller problems, computational times soon improve (with respect to solving analysis in a uniform mesh) when higher resolutions are required. In this context, it is worth noticing that while the analysis time in uniform mesh increases by a factor of ≈ 5 when the resolution is doubled, t_a in the adaptive mesh increases by a factor of ≈ 4 and t_m increases by a factor around 2 (i.e., linearly with the resolution of the problem). This further demonstrates that the use of the proposed technique is particularly favorable in larger problems.

This is yet more evident in large 3D problems. For instance, in the 3D problem of Figure 24D, where the discretization was conducted with more than 21 million elements, we have

$$\begin{aligned} t_a \text{ with } n_{ref} = 4 \quad & 6629, \\ t_m \text{ with } n_{ref} = 4 \quad & 520. \end{aligned} \tag{35}$$

As regards the comparison with the uniform analysis mesh, we did not run the experiment with uniform analysis mesh, as it would be impractical due to the dimensions of the problem. Nonetheless, the time saving is apparent from the analysis later performed in Section 5.4.2, and we can assume that t_m is almost negligible with respect to the time that would be required to perform the analysis on the uniform mesh. Finally, we remark that also parallel computing was here used, confirming the efficiency of the density gradient-based mesh also in this framework.

TABLE 7 Analysis of computational times for the designs of Figure 14

n_{ref}	t_a	t_m
Not adaptive	508.5	-
1	269.7	56.1
2	222.7	57.2
3	212.1	57.1
4	208.8	57.5
5	205.7	58.2

TABLE 8 Analysis of the scalability of computational times

r_d	t_a not adaptive	t_a with $n_{ref} = 4$	t_m with $n_{ref} = 4$
32	5.2	5.1	4.6
64	21.7	16.5	10.8
128	98.9	53.0	17.7
256	508.5	208.8	57.5

5 | NUMERICAL EXPERIMENTS

5.1 | Numerical data and computational setting

Figure 13 summarizes the considered experiments. The problems involve classical topology optimization problems, such as the compliant mechanism and the MBB beam (see, e.g., References 1,33,34), and roof support problems with distributed load that could arise in applied contexts such as topology optimization of support structures for additive manufacturing or of bridge generation[35,36].

In all the problems, the Young's modulus, the length L, and the force F are assumed to be unitary. The distributed load in the 2D support problem is also unitary, while the distributed load of the 3D problems is 100 and is modeled by applying normalized point-loads to all the nodes of the loaded boundary. In this latter case, the mesh of the loaded boundary is always refined to avoid any possible inconsistency in the load application on the adaptive mesh. The Poisson's ratio is $\nu = 0.3$. In addition, in the compliant mechanism, the inlet is 0.06 L long and the outlet is 0.14 L long, both centered at half the height of the domain. The elasticity constant of the spring is $k_s = 0.01$. The problems are also subjected to a volume constraint, with prescribed volume fraction $\overline{V} = 0.5$ for the compliant mechanism and for the MBB beam, $\overline{V} = 0.25$ for the 2D support, $\overline{V} = 0.1$ for the 3D bridge, and $\overline{V} = 0.04$ for the 3D tree.

We solve the problems in the FEniCS environment[37,38], stopping the procedures after $it_{tot} = 150$ iterations. Densities are assumed to be nodal-based and material properties are interpolated by the SIMP method[1,39] with penalty $p = 3$. The initial density distribution is given by a uniform density field, where all values are set equal to the maximum volume fraction $\overline{V}$ admitted by the problem. The optimization is performed by the method of moving asymptotes (MMA).[31] The reported values of the final cost have been computed on a uniform mesh, in order to afford for consistent comparisons between the optimality of the designs obtained by uniform and adaptive analysis. The analysis systems are solved directly in 2D by sparse LU and iteratively in 3D by GMRES with Hypre AMG preconditioner.

Regarding the parameters of the density gradient-based analysis mesh, in all cases we set $\epsilon_{cd} = 0.05$ and we initialize $\epsilon_{ref} = 0.1$. We then relax ϵ_{ref} at later iterations, after the design starts forming, setting it to 0.5 from iteration 10 and to 1 from iteration 25. Unless otherwise specified, in 2D the adaptive mesh is used as soon as possible, that is, from iteration 2 (at iteration 1, the initial density is uniform and the gradient is zero everywhere). In 3D, the adaptive mesh is used immediately as well, but we wait a couple more iterations to avoid numerical difficulties given by uniform density distributions. We then use the adaptive analysis mesh starting from iteration 5. For simplicity, when the problems are solved

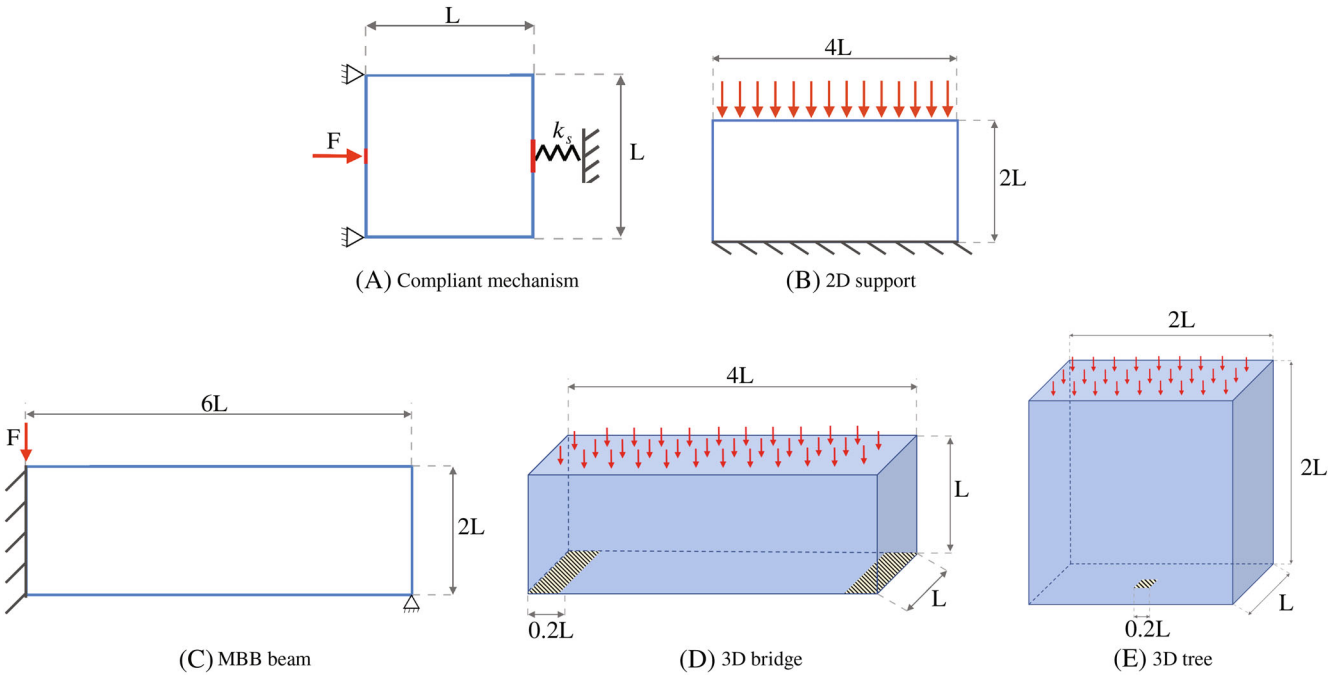


FIGURE 13 Numerical experiments

in series, the functions belonging to the density and analysis meshes are mapped by projections whenever is needed, using the *project* command of FEniCS. When parallel programming is employed, the above command can no longer be used between non uniform meshes (as they are partitioned differently, and each processor may lack of all the information needed to perform the projection), and we use the *LagrangeInterpolator* command, which can interpolate between meshes in parallel.

Regarding the computational setting, 2D experiments with less than 1 million nodes are solved in series on a laptop provided with a dual core 2.7 GHz Intel Core i5 (Broadwell architecture) and with 8 GBs of memory. Larger 2D problems are, instead, solved on an Azure *Standard_D3_v2* computer, with four virtual CPUs and 14 GiB of memory. Finally, 3D experiments are performed on an Azure's *Standard_HB120rs_v2* machine, which is provided with 120 virtual CPUs (processor: AMD EPYC 7V12) and 456 GiB of memory.

To evaluate the efficiency of the proposed mesh, we often provide measures associated to the number of DOFs. This is motivated by the fact that a direct wall-clock time comparison would be less meaningful due to the differences in the computational setting, as pointed out also in Reference 27. Thus, we use the measure E_a, which we define, similarly to [27, Eq. (36)], as

$$E_a := \frac{\sum_{k=1}^{it_{tot}} \left(n_{DOF,design}^{(k)} \right)^{n_s}}{\sum_{k=1}^{it_{tot}} \left(n_{DOF,analysis}^{(k)} \right)^{n_s}}. \tag{36}$$

In the previous expression, n_s relates the DOF number to the computational effort when a sparse linear solver is used[27,40]. As in Reference 27, we set $n_s = 1$ in 2D and $n_s = 4/3$ in 3D. In addition, we also report the efficiency measure

$$E_a^{(end)} := \frac{\left(n_{DOF,design}^{(end)} \right)^{n_s}}{\left(n_{DOF,analysis}^{(end)} \right)^{n_s}}, \tag{37}$$

which refers to the final design, which is stable and with low grayness. Therefore, $E_a^{(end)}$ gives an idea of the efficiency we could expect if more iterations were performed.

5.2 | Compliant mechanism

First, let us consider the 2D compliant mechanism. Densities are filtered by an isotropic PDE filter[41,42] with filter radius $r_f = 3$. Here, r_f must be intended to characterize the actual filter radius in terms of the number of elements in the filtering (see also Reference 42). The final designs obtained with uniform and adaptive analysis mesh with various choices of the number of adaptive refinements n_{ref} are reported in Figure 14. The design mesh is made of 131,072 finite elements. We also report the numerical value of the final cost and the last analysis mesh in all cases.

The designs in Figures 14A–F are almost identical. Also the numerical values of the final cost are practically the same: the difference is always smaller than 0.5%. Therefore, the density gradient-based analysis mesh is able to detect the topology change and to provide a sufficiently accurate solution of the analysis problems. This applies irrespectively of the number of adaptive refinements that are performed. The same happens also when $n_{ref} = 6$ and $n_{ref} = 7$, whose designs are not reported for compactness but whose results are reported in Table 9. It is worth noticing that the issue of QR-patterns does not arise, although in these cases the resolution of the initial analysis mesh is very coarse. This is motivated by the fact that the $r_a = r_d$ at solid-void interfaces constitutes an "implicit" upper bound for the coarsening level of the analysis mesh, especially in the more critical regions of the design. This can be noticed by the final analysis meshes reported in Figure 14. The choice of n_{ref} does not appear, then, particularly critical and it is mainly based on efficiency reasons.

Nonetheless, n_{ref} cannot be arbitrarily large, either: indeed, the resolution of the initial analysis mesh, r_{a0}, must be sufficient to activate the gradient-based refinements. The limit case when this does no longer happen for the analyzed problem is represented in Figure 14G, where eight density gradient-based refinements were performed starting from an initial uniform mesh of resolution $r_{a0} = 1$. In this case, the only nodes of the starting mesh are at the boundaries of the domain, and the design is not detected at all. No density gradient-based refinement is, then, triggered, and the analysis mesh remains extremely coarse everywhere during all iterations of the procedure. As a consequence, the mesh cannot

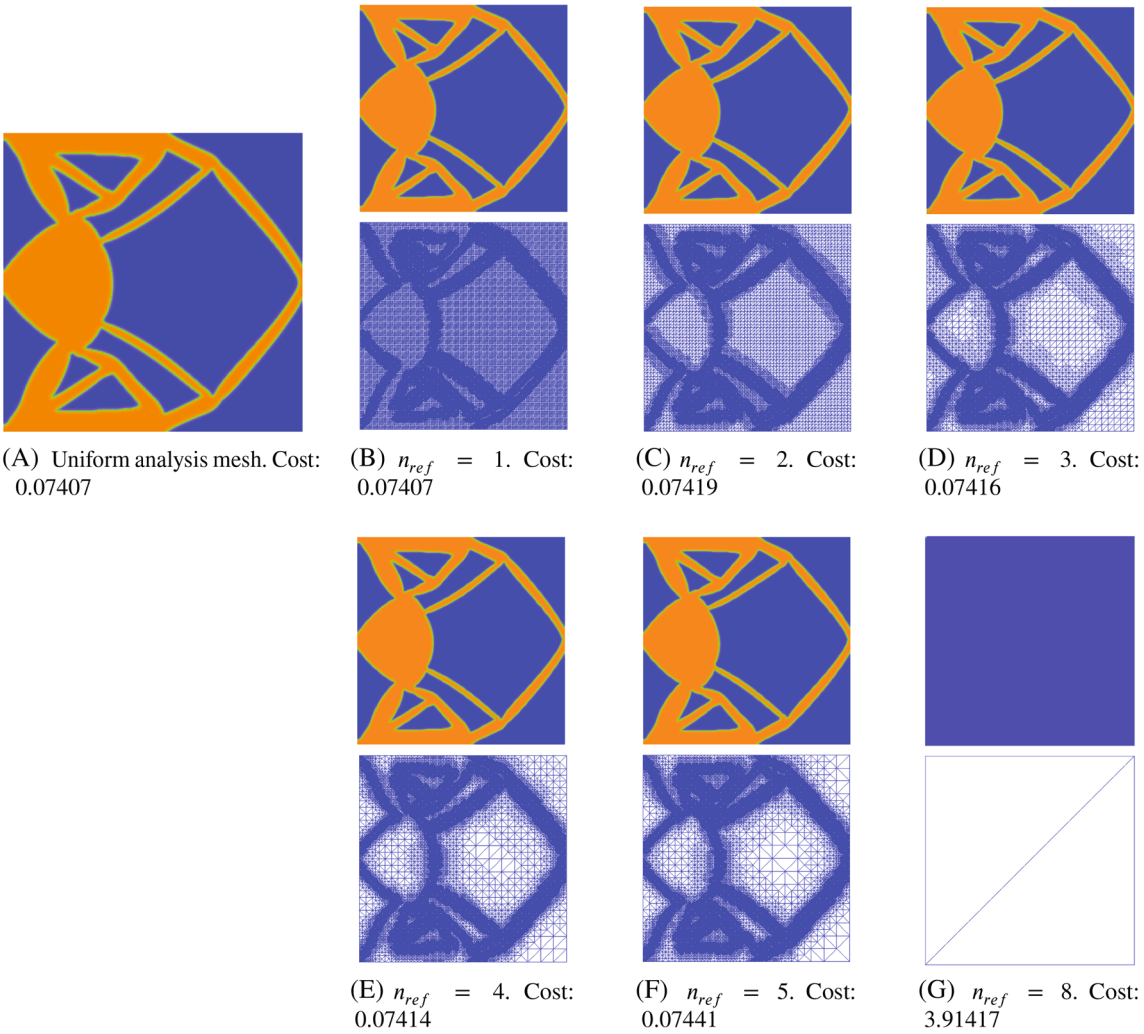


FIGURE 14 Compliant mechanism, final designs with $r_d = 256$. Analysis performed on density gradient-based adaptive meshes built with various choices of the number of refinements n_{ref}

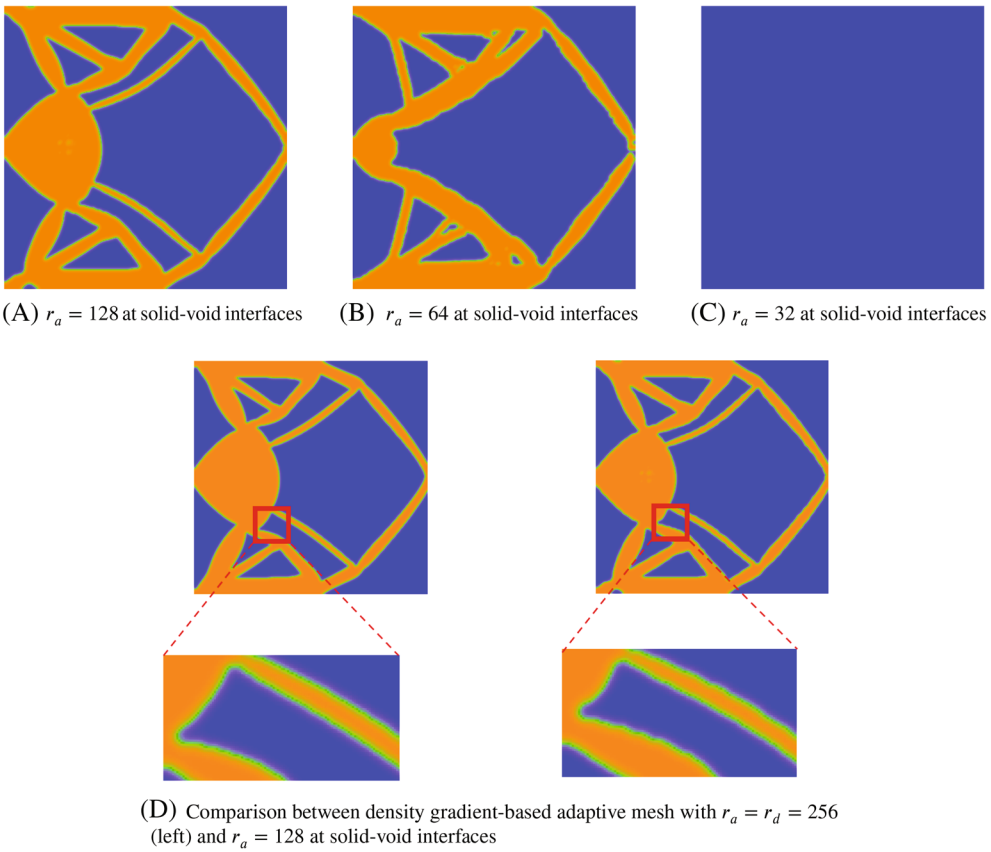
detect any topology change and the optimizer is unable to build the optimized design. Nonetheless, this limit case can be easily avoided: we have shown in the previous section that 3–4 refinements are enough to achieve great part of the improvement in efficiency, and limiting the number of refinements is then a viable way to avoid the above and other numerical issues. In alternative, it is possible to refine also elements that contain boundary nodes close to applications points of loads and boundary conditions.

At the same time, the cost of the analysis was significantly reduced by the adaptive mesh. In this regard, Table 9 reports the details of the efficiency of the density gradient-based analysis mesh. Beyond E_a and $E_a^{(end)}$, we also report the final number of elements in the last computed analysis mesh, n_a, the relative saving δn as in (19), and the ratio c_a/c_u, where c_a and c_u are the final values of the objective for the designs obtained by performing the analysis in the adaptive and in the uniform mesh, respectively.

The values of E_a and of $E_a^{(end)}$ in Table 9 confirm that the density gradient-based analysis mesh improves the efficiency of the solution of the analysis problems not only at the last iterations (when the density gradients are sharp and do not change much between one iteration to the other), but also throughout the entire optimization process, which includes the iterations when the design is being built and large regions with nonzero density gradient are present. Furthermore, they further confirm that we achieve almost the entire efficiency gain when we perform 3–5 refinements. Finally, the optimized design and the corresponding cost remains almost identical to the cost computed in the uniform design mesh, as testified by c_a/c_u. Therefore, the solution of the analysis problems is sufficiently accurate in all the computed density gradient-based meshes.

TABLE 9 Efficiency of the density gradient-based analysis mesh for the compliant mechanism problem with several choices of n_{ref} and $n_d = 131{,}072$

n_{ref}	n_a	δn	E_a	$E_a^{(end)}$	c_a/c_u
1	73,519	43.9%	1.65	1.78	1.00014
2	62,077	52.6%	1.89	2.11	1.00165
3	59,538	54.6%	1.95	2.20	1.00123
4	58,892	55.1%	1.97	2.23	1.00095
5	58,015	55.7%	2.00	2.26	1.00471
6	58,623	55.3%	2.00	2.24	1.00698
7	57,936	55.8%	2.00	2.27	1.00857



(A) $r_a = 128$ at solid-void interfaces (B) $r_a = 64$ at solid-void interfaces (C) $r_a = 32$ at solid-void interfaces

(D) Comparison between density gradient-based adaptive mesh with $r_a = r_d = 256$ (left) and $r_a = 128$ at solid-void interfaces

FIGURE 15 Final designs obtained coarsening the mesh also at solid-void interfaces

On the other hand, if we reduce the resolution of analysis at the solid-void interfaces, the result changes significantly. This is shown in Figure 15, which shows what happens if we coarsen the analysis mesh also at the solid-void interfaces. In all cases, the analysis mesh was computed by four density gradient-based refinements (starting from an appropriate mesh or resolution r_{a0}), which led to the prescribed value of r_a at solid-void interfaces.

In particular, we notice that, if we coarsen the resolution also where the density gradient is nonzero, soon some irregularities appear at the interfaces between solid and void. This already happens for $r_a = 128$ at solid-void interfaces, as shown in Figures 15A,D. In addition, some irregularities also form on the left of the upper and lower edges. Notice that $r_a = 128$ is still much finer than the resolution of the analysis mesh that we commonly used in solid and in void regions with no contraindications. If r_a is further reduced, the design becomes progressively more irregular and changes significantly. Finally, further reductions of r_a at solid-void interfaces completely prevent the formation of the design, as in Figure 15C. In this case, the analysis mesh is not sufficiently fine as to detect any topology change.

As regards scalability, Table 10 reports the efficiency of density gradient-based analysis meshes when the design resolution is changed. The savings tend to increase with the dimension of the problem. The procedure is, then, well scalable.

Finally, in Figure 16 we examine the convergence to the solution and the link between the design and the density gradient-based analysis mesh, considering, for instance, the case with $n_{ref} = 5$.

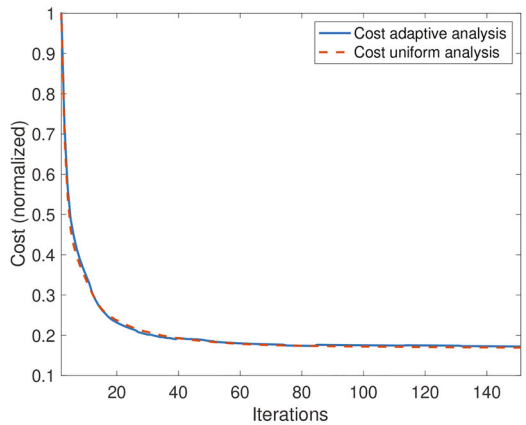
In particular, Figure 16A shows how the (normalized) value of the cost changes during the optimization when the analysis is solved in a uniform or in the density-gradient based mesh. We notice that the convergence is almost identical in the two cases, confirming the capabilities of the proposed adaptive analysis mesh. Furthermore, we observe that the cost rapidly decreases at the first iterations, due to the fact that the design is being built and the topology changes significantly at every iteration. At the same time, when the adaptive analysis mesh is used, the density gradient-based mesh changes at every iteration: indeed, in Figure 16B we observe that M_{cd} in the first iteration is always above the threshold ϵ_{cd}, and it therefore prescribes a remeshing.

At later iterations, the design stabilizes and the cost starts converging, as shown by Figure 16A. Also the behavior of M_{cd} changes, and it exhibits a zig-zag trend with many values below ϵ_{cd}. This happens because M_{cd} is computed based on the density distribution at the current iteration, $\gamma^{(k)}$, and the density distribution at the last iteration when the remesh was performed, $\gamma^{(old)}$. Thus, until $M_{cd} < \epsilon_{cd}$, $\gamma^{(old)}$ remains the same, while $\gamma^{(k)}$ is changing. The value of M_{cd} tends, then, to increase as the topology gets farther from $\gamma^{(old)}$. Then, at some point, we will have $M_{cd} \geq \epsilon_{cd}$, which triggers the generation of a new density gradient-based adaptive mesh and the update of $\gamma^{(old)}$. Thus, we have an abrupt decrease of M_{cd}. This is exactly what we observe in Figure 16B after about 40 iterations.

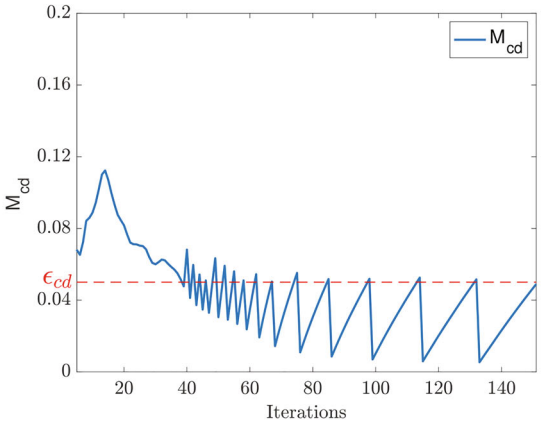
In this regard, it is also worth noticing that the "peaks" of the zig-zagging behavior of M_{cd} (corresponding to the remeshings) become increasingly far from one another. Indeed, at later iterations, the design is increasingly sharp and stable, so that every iteration produces just a moderate change of topology. This further highlights the strict link between the design and the density gradient-based mesh. Moreover, it also shows that the remeshing criterion (14) allows to avoid performing many remeshings (in this experiment, almost two thirds of what we would otherwise perform), thus improving the efficiency. The gain in efficiency would further increase if more iterations were performed.

TABLE 10 Scalability of the procedure, considering design mesh of different dimensions and the same number of refinements $n_{ref} = 4$ for the analysis mesh

n_d	n_a	δn	E_a	$E_a^{(end)}$
8,192	5,978	27.0%	1.33	1.37
32,768	17,541	46.5%	1.75	1.87
131,072	58,892	55.1%	1.97	2.23



(A) Convergence analysis



(B) Measure of the change of density distribution M_{cd} and remeshing

FIGURE 16 History of the evolution of the cost and of M_{cd} for the case $n_{ref} = 5$

5.3 | 2D elasticity problems

Let us now pass to 2D elasticity problems. Here, the PDE filter has radius $r_f = 10$, and also the Heaviside filter[43] is applied. In the Heaviside filter, we set $\eta = 0.5$, while the parameter β (which regulates how sharp the filter is) is doubled every 25 iterations, starting from $\beta = 1$ up to $\beta = 32$. Considering, for instance, the 2D support problem in a design mesh of 409,600 finite elements, Figure 17 shows the effect of n_{ref} on the final design when the density gradient-based analysis mesh is used.

As before, the design does not change significantly when the number of adaptive refinements is changed. Indeed, the final design and the numerical value of the final cost remain almost unaltered, irrespective of n_{ref}. Details on the efficiency and on the numerical values of the final cost for the designs in Figure 17 are reported in Table 11. The same considerations performed with regard to the compliant mechanism can be repeated also in this case.

The 2D support is a problem whose solution is expected to have a tree-like structure, with various thin features resembling branches. When the design mesh is finer, we expect to be able to identify more of these thin features. It is then interesting to evaluate whether our density gradient-based analysis mesh is able to detect the topology change in a sufficiently accurate way as to obtain high-resolution final designs. In this regard, Figure 18 shows the final designs computed

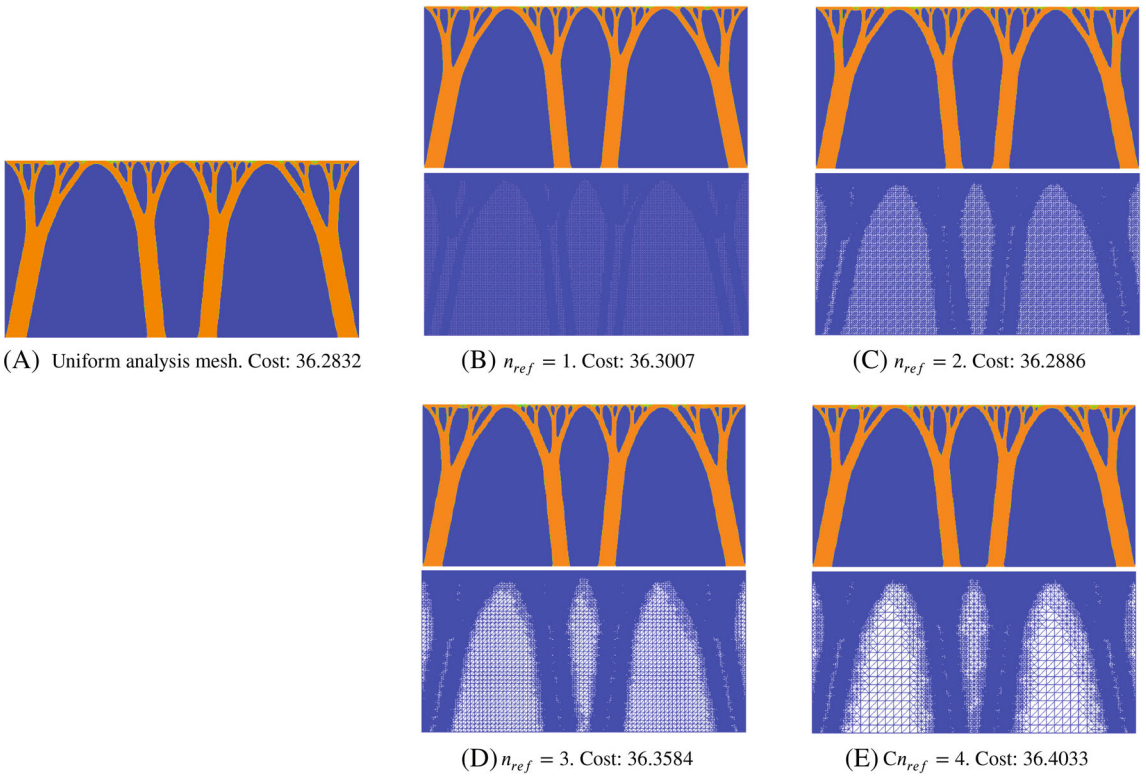
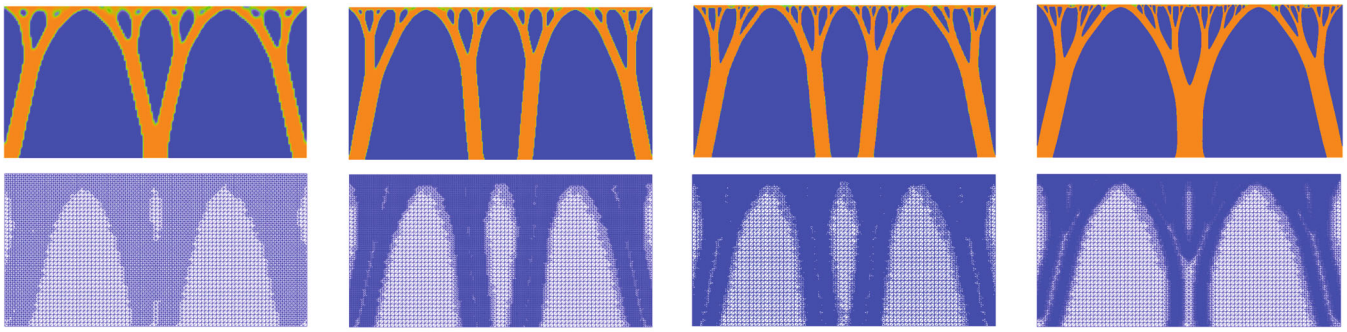


FIGURE 17 2D support, final designs with $r_d = 160$. Analysis performed on density gradient-based adaptive meshes built with various choices of the number of refinements n_{ref}

TABLE 11 Efficiency of the density gradient-based analysis mesh for the 2D support problem with several choices of n_{ref} and for $r_d = 160$ (corresponding to $n_d = 409{,}600$ design elements)

n_{ref}	n_a	δn	E_a	$E_a^{(end)}$	c_a/c_u
1	208,615	49.0%	2.02	1.96	1.00048
2	171,515	58.1%	2.54	2.38	1.00015
3	164,893	59.7%	2.68	2.48	1.00207
4	163,400	60.1%	2.69	2.50	1.00331



(A) $r_d = 40, r_{a0} = 20, n_{ref} = 1$ (B) $r_d = 80, r_{a0} = 20, n_{ref} = 2$ (C) $r_d = 160, r_{a0} = 20, n_{ref} = 3$ (D) $r_d = 320, r_{a0} = 20, n_{ref} = 4$

FIGURE 18 Final designs computed by density gradient-based analysis mesh with different design resolutions

TABLE 12 Comparison between the number of elements in uniform and density gradient-based analysis mesh ($r_{a0} = 20$) for the 2D support

n_d	n_a	δn	E_a	$E_a^{(end)}$	c_a/c_u
25,600	17,168	32.9%	1.59	1.49	1.03280
102,400	56,084	45.2%	2.00	1.82	0.99320
409,600	164,893	59.7%	2.68	2.48	1.00207
1,638,400	398,567	75.7%	4.24	4.10	1.00043

with various design resolutions. In all cases, the minimum resolution of the analysis is $r_{a0} = 20$ and the filter radius (in terms of the number of elements) is $r_f = 10$. Fixing r_{a0} allows to compare more consistently the efficiencies of the adaptive analysis mesh at different design resolutions, while fixing r_f allows us to more easily explore whether our adaptive analysis mesh is able to support fine design details in higher resolution meshes. Differences among the results at design resolutions should, then, not be attributed to mesh convergence issues. This is further substantiated by the fact that all the designs are consistent with what we would get in a uniform analysis mesh, as later remarked with regard to the data in Table 12.

As expected, the final design is more detailed and presents more fine features as the design mesh is refined. This demonstrates that, although r_{a0} is the same in all cases, the density gradient-based adaptive analysis mesh is able to detect the topology change also in fine features, when the design is represented in a fine mesh.

Notice that the previous cases include large scale problems: $r_d = 320$ has more than 1 million design elements and it is then particularly important to use as few analysis elements as possible in order to achieve high efficiency. In this regard, Table 12 reports the saving for the cases represented in Figure 18. We remark that all these designs are in good agreement with what we would obtain with a uniform analysis mesh, as testified by the ratio c_a/c_u that is reported in the table.

The saving in terms of elements scales well with the dimensions of the problem and, in the largest case, it exceeds 75%. This result, together with the fact that we have shown in Figure 18 that the used analysis mesh is able to detect the topology change as a uniform analysis mesh, further testifies the efficiency and applicability of the proposed procedure. Figure 19 is finally used to verify mesh convergence by considering what happens when we use as radius a physical distance (in this case, we choose 0.0361271676301, which corresponds to $r_f = 10$ in the problem with $r_d = 80$) for the first three problems of Figure 18. The designs now do not differ when the resolution is increased, further substantiating the fact that our adaptive analysis does not lead to mesh convergence issues.

Finally, for completeness, we report the result of another well-known 2D elasticity problem, that is, the MBB beam. Also in this case, densities are filtered by both the PDE filter and the Heaviside filter, with the same parameters and continuation methods of the previous experiment. Figure 20 shows two optimized designs computed in a design mesh of 153,600 and of 2,457,600 finite elements, respectively. The analysis is performed on a density gradient-based mesh with $r_{a0} = 20$. The saving in terms of the number of elements is reported in Table 13.

The designs of Figure 20 are in agreement with results in the literature and with what we would obtain performing the analysis in a uniform mesh. Indeed, the analysis in the density gradient-based mesh is again sufficiently accurate to

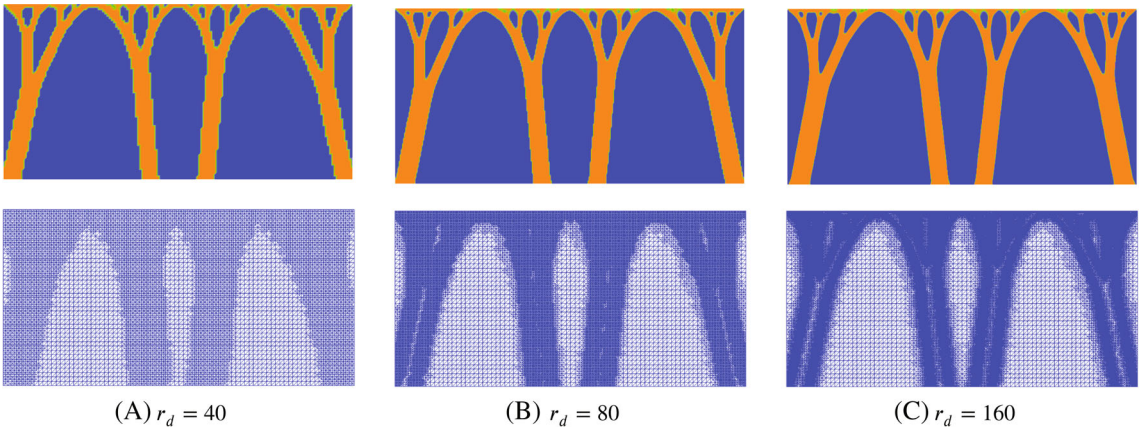


FIGURE 19 Final designs with filter radius as the same physical distance in different design resolutions

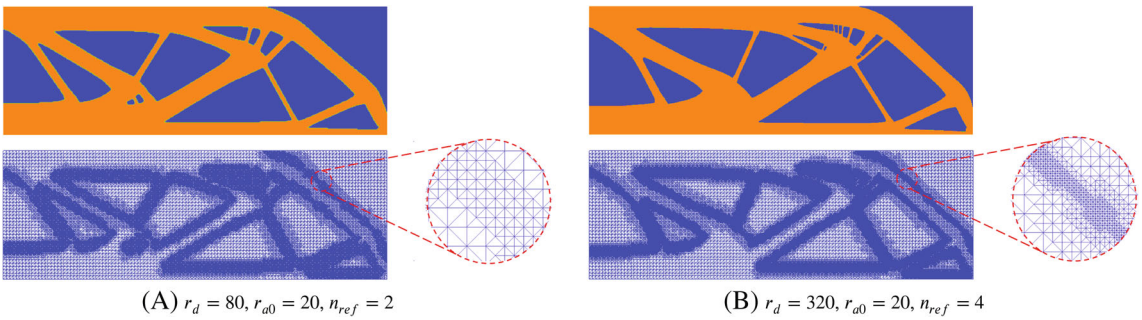


FIGURE 20 2D MBB beam, final designs with different design resolutions and analysis computed in density gradient-based meshes with $r_{a0} = 20$

TABLE 13 Comparison between the number of elements in uniform and density gradient-based analysis mesh for the 2D MBB beam

n_d	n_a	δn	E_a	$E_a^{(end)}$
153,600	70,521	54.1%	2.10	2.18
2,457,600	419,213	82.9%	4.37	5.87

detect the topology change, and to build fine features and small holes. At the same time, the saving in terms of elements is significant and exceeds 80% in the largest problem. This is competitive with results in the recent literature: for instance, although the results are not directly comparable for differences in the general settings and in the refinement algorithms, the efficiency estimates in Reference 27 for the MBB beam with linear elements are between 2.05 and 5.40.

Finally, it is interesting to evaluate how the efficiency varies depending on the feature size. Indeed, designs with many fine features are characterized by a longer perimeter and, hence, by more elements with nonzero density gradient. We then expect that the efficiency of the density gradient-based analysis will increase when a large PDE filter radius suppresses the smaller features of the design, reducing the overall perimeter. We verify that this is indeed what happens considering, for instance, the MBB beam problem. Indeed, the MBB beam is a well-known reference problem in the literature, and it is easy to assess the validity of the obtained designs.

In particular, we consider a similar setting as the one by which Figure 20A was obtained, but we perform more density gradient-based refinements to emphasize the effect of the adaptive mesh and to analyze MBB problems where r_{a0} is smaller than in Figure 13. In particular, we set $r_d = 80$, $r_{a0} = 5$, and $n_{ref} = 4$. Modifying only the value of the filter radius, we obtain the designs in Figure 21. The corresponding analysis of the efficiency of the density gradient-based analysis mesh is reported in Table 14.

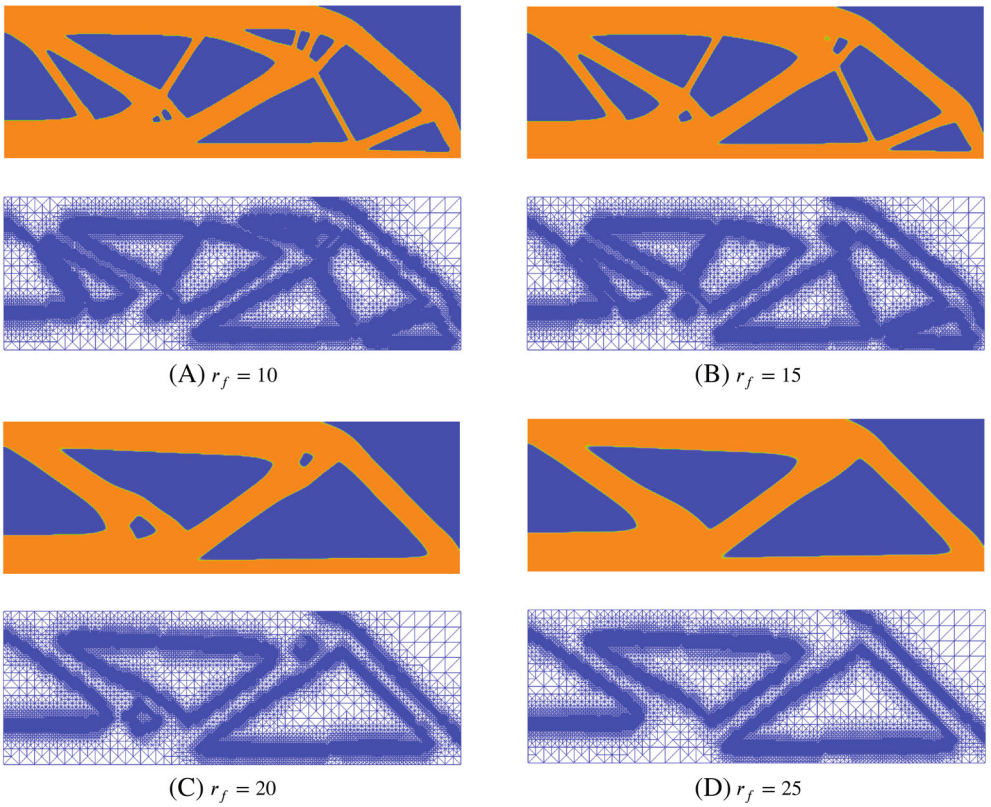


FIGURE 21 Optimized designs and final density gradient-based mesh. Design resolution is $r_d = 80$, while the analysis mesh is built by four gradient-based refinements from $r_{a0} = 5$

TABLE 14 Efficiency of the density gradient-based analysis mesh when the PDE filter radius (and, hence, the feature size) changes. Results corresponding to the designs of Figure 21. Design resolution is $r_d = 80$ in all cases ($n_d = 153{,}600$)

r_f	n_a	δn	E_a	$E_a^{(end)}$
10	67,544	56.0%	2.16	2.28
15	63,022	59.0%	2.23	2.44
20	50,747	67.0%	2.52	3.04
25	46,991	69.4%	2.54	3.28

The results demonstrate that the efficiency of the analysis mesh changes significantly when a large PDE filter radius suppresses the finer features of the design. Indeed, the mesh is fine along the contour of the design, and reducing the length of perimeter implies that less elements have nonzero density gradient (as long as the design is sharp). Furthermore, we notice that E_a increases more slowly than $E_a^{(end)}$ when r_f is increased. This confirms that the gain in efficiency is particularly significant at later iterations, and is consistent with the previous analysis. Indeed, at first iterations, the design will be gray and the density gradient-based analysis mesh will be fine in large regions of the domain, almost irrespective of r_f (and, arguably, a large r_f may also worsen efficiency). On the contrary, after the design has formed, the density gradient mesh is regulated mainly by the perimeter of the design, where all regions with $\nabla\gamma \neq 0$ are now concentrated. The efficiency then is significantly higher when the feature size is larger.

5.4 | 3D problems

In this subsection, we solve the 3D problems introduced in Figure 13. As we will be considering also large-scale 3D problems, we solve iteratively the finite-element systems. This demonstrates that the analysis meshes generated by our

technique do not lead to severe ill-conditioning of the matrices of the system, at least in the considered cases. Therefore, it is possible to couple the adaptive analysis mesh with the inexact solution of the analysis problems for increased efficiency. More precisely, in the following we solve the analysis problems by GMRES solver with Hypre AMG preconditioner. In other occurrences where the solution of a system is required (for instance, for the PDE filter) we use GMRES with Additive Schwarz preconditioner. All these solvers have been applied through PETSc[44,45], using their default settings.

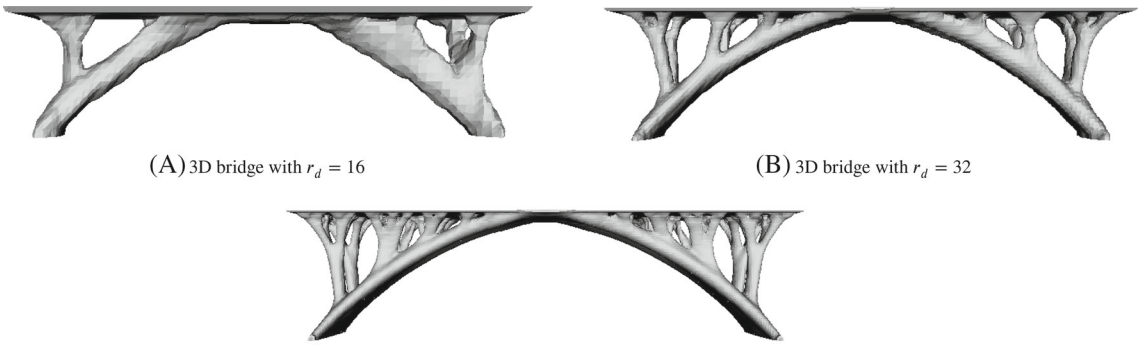
5.4.1 | 3D bridge

Let us solve the 3D bridge problem. We apply both the PDE filter (with $r_f = 3$) and the Heaviside filter (with the same continuation methods as in 2D and with $\beta_{\max} = 16$). The 3D bridge as shown in Figure 13D is conceptually similar to the 2D support. The main difference is that the zero displacement boundary condition is applied only at the sides of the bottom part of the domain. Therefore, as in the 2D support, we expect that, increasing the resolution of the design, we will have more fine features and branch-like structures, especially toward the top of the domain. It is then interesting to verify whether the analysis can be solved in our density gradient-based mesh in a sufficiently accurate way as to detect these thin structures. Figure 22 reports the results for $r_d = 16$, $r_d = 32$, and $r_d = 64$ corresponding to design meshes of 98304, 786432, and 6291456 elements, respectively. In all cases, we set $n_{ref} = 4$. In order to improve the readability of the results, we avoid the presence of discretization voxels by reporting the final designs before the application of the Heaviside filter. No postprocessing is made, and the figures come directly by just using the command *Iso Volume* of the Paraview software, with threshold 0.5. Notice that the following largest scale problems involve millions of variables in a 3D setting, and have been solved on a computer cluster. It is, therefore, impractical to solve the problems also in a uniform analysis mesh, where analysis takes much longer than in the adaptive setting. Thus, we here present only the results obtained using adaptive analysis meshes.

When the resolution of the design mesh is increased, the final result becomes gradually smoother and finer branches are detected. Therefore, the density gradient-based adaptive mesh behaves correctly also in three dimensions, with iterative solvers and for large problems, where several processors are employed in parallel for solving the systems.

As regards efficiency and scalability, Table 15 reports the efficiency indices for the considered problems.

Table 15 allows to appreciate the dimension of the problems, the largest of which involves more than 6 million finite elements. Therefore, the use of an adaptive analysis mesh appears particularly desirable in order to reduce the dimensions of the analysis systems by thousands and, possibly, millions of variables. This is indeed what happens: the efficiency



(A) 3D bridge with $r_d = 16$ (B) 3D bridge with $r_d = 32$

(C) 3D bridge with $r_d = 64$

FIGURE 22 3D bridge for various design resolutions

TABLE 15 Efficiency of the density gradient-based analysis mesh for the 3D support problem

r_d	n_d	E_a	$E_a^{(end)}$	c_a
16	98,304	1.67	1.64	1.2E-2
32	786,432	2.39	2.64	8.7E-3
64	6,291,456	2.97	3.27	7.4E-3

parameters E_a and $E_a^{(end)}$ both reveal a significant efficiency improvement, which translates into the saving of millions of variables in the largest problem. This is particularly significant as the analyzed problem presents several regions with nonzero density gradient (indeed, we have a long perimeter, given by the many pillars and fine features that are formed), which forces to refine the analysis mesh in several regions. In all cases, the design is still rich of details and of fine features (increasingly with the dimension of the design mesh), as shown in the previous figures, testifying that it is possible to solve the analysis in a sufficiently accurate way. Finally, Table 15 also shows that the efficiency tends to increase with the dimension of the problem. This reveals a good scalability of the procedure and is consistent with what we observed in two dimensions.

Finally, Figure 23 shows that the final analysis mesh is finer along the contours of the design, while it is coarser in the other regions. This is particularly apparent in the large hole below the bridge, but it can easily be noticed that the mesh becomes rapidly coarser also in smaller zones, as soon as we are sufficiently far from a solid-void interface. Thus, the density gradient-based analysis mesh works as expected in the 3D setting.

5.4.2 | 3D tree

Although the computational saving achieved with our adaptive mesh for the 3D bridge was significant, it was limited by the long perimeter of the bridge and of the supporting pillars. In the 3D tree example (see Figure 13E), we solve a mechanically similar problem, but with a smaller volume fraction and with boundary conditions that are expected to generate a single pillar (i.e., the trunk of the tree) and branches that should become progressively finer as the design resolution is increased. The aim is to see whether fine features are detected by the adaptive analysis in yet larger problems and to evaluate the computational saving in a 3D structure with smaller volume fraction (which can lead to a shorter perimeter, but which can also complicate the generation of the design under the strict volume constraint). Furthermore, this situation is more similar to other 3D examples in the literature on adaptive analysis meshes, like in Reference 27. Regarding the dimension of the problems, beyond considering $r_d = 16$, $r_d = 32$, and $r_d = 64$ (which, in the domain of the problem, correspond to design meshes as large as the ones used in the 3D bridge experiments), we also solve a problem with $r_d = 96$, which corresponds to a design mesh of 21,233,664 elements.

Setting $\overline{V} = 0.04$ and performing three adaptive refinements to generate the density gradient-based analysis mesh, we obtain the designs in Figure 24. The corresponding efficiency measures are reported in Table 16.

Figure 24 shows that the adaptive analysis mesh is able to detect the topology change in all experiments, including the largest one. In this latter case, reported in Figure 24D, the resolution allows to form very fine columns and thin "sheets" of material between branches. Moreover, the central pillar splits in two at the base, and tends to form two different columns, further showing the ability of the procedure to detect new design features as the design mesh is refined.

At the same time, the efficiency of the adaptive mesh increases with the dimension of the problem, as shown in Table 16. In particular, in the largest case, E_a and $E_a^{(end)}$ are almost double than what we got in the largest 2D experiment

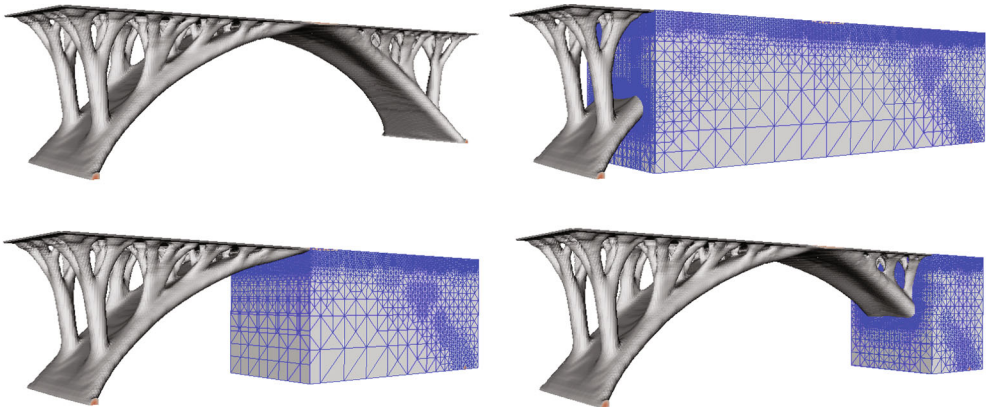


FIGURE 23 Optimized design and sections of the density gradient-based analysis mesh

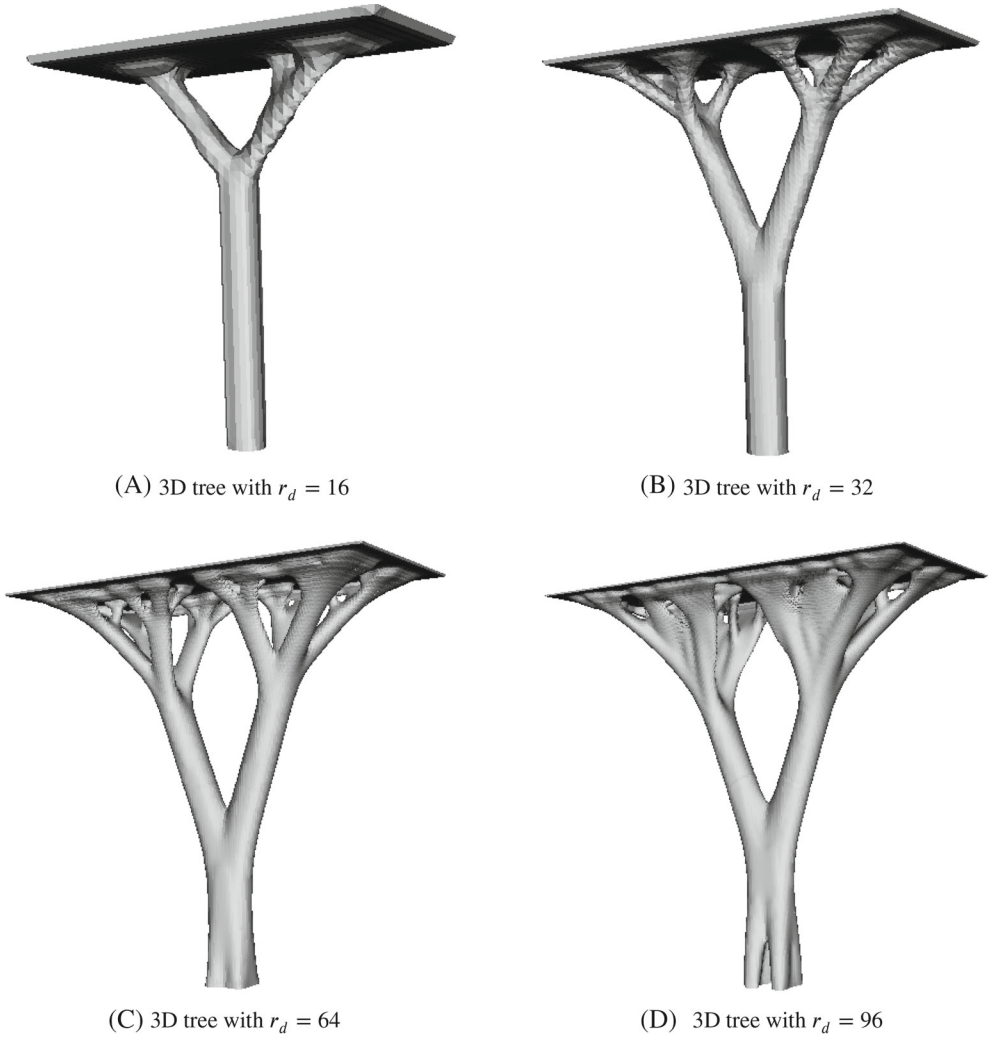


FIGURE 24 3D tree for various design resolutions

TABLE 16 Efficiency of the density gradient-based analysis mesh for the 3D tree problem

r_d	n_d	E_a	$E_a^{(end)}$	c_a
16	98,304	2.59	2.67	2.1E-3
32	786,432	4.00	4.51	1.6E-3
64	6,291,456	6.25	7.72	1.3E-3
96	21,233,664	8.50	10.55	1.4E-3

that we analyzed earlier. This demonstrates that the density gradient-based mesh is particularly effective in large 3D problems. The value of $E_a^{(end)}$ is here particularly interesting: indeed, it characterizes the efficiency of the mesh at the last iteration, where the design is finally sharp and stable. Thus, it can be viewed as the efficiency measure of the optimized structure, which depends on the final perimeter. The obtained value of $E_a^{(end)}$ is much larger than E_a because E_a contains the efficiency at the first five iterations, which are conducted in the uniform design mesh to obtain an initial distribution of the density gradient. These iterations are characterized by unitary efficiency. When the efficiency of the adaptive mesh is much larger than one, this penalizes considerably the final value of E_a. For instance, it is sufficient to start registering E_a from when the adaptive mesh is used to obtain $E_a = 10.70$. Finally, it is worth remarking that the peak efficiency of the density gradient-based analysis mesh is 27.37, which is obtained at iteration 11, when the design is forming but thinner branches have yet to be built.

6 | CONCLUSIONS

We have presented a density gradient-based adaptive refinement of the analysis mesh for efficient topology optimization. In our procedure, the finite-element analysis is performed in an analysis mesh whose resolution is the same as the design mesh only at elements where the gradient of the density function is not zero. This means that the resolution of the analysis mesh is fine only at solid-void interfaces and, possibly, in regions where intermediate densities are present. All other regions (i.e., solid and void regions) are represented in a coarser grid. We generate such mesh by a sequence of successive gradient-based refinements, whose number is arbitrary and depends on the desired coarseness of the mesh in regions where the density gradient is zero.

We have shown that our analysis mesh is fine in all regions where the topology is changing, and that the density gradient-based refinement is favorable also for ensuing consistency in interpolating/projecting quantities across design and analysis meshes of different resolutions. Furthermore, we have shown that the proposed adaptive meshing scheme is simple and that we can avoid remeshing whenever the topology does not change significantly from the last refinement. This follows directly from the fact that the density gradient is an entity that is rich of physical information, and its use as a refinement indicator allows to link the topology evolution to the analysis mesh density. Thus, we have regulated the choice on whether or not to refine by a measure of the change of density distribution. Finally, although the density gradient-based analysis mesh is not explicitly based on error evaluations, we have shown numerically in a test problem that the error of the solution of the analysis (evaluated by an a posteriori measure) remains comparable with the error obtained solving the analysis in the same mesh as the design.

We have also analyzed the efficiency of the proposed density gradient-based analysis mesh. In particular, we have characterized the efficiency especially with regard to the number of elements (and, consequently, of nodes, and DOFs) of the mesh, comparing the uniform and the density gradient-based settings. In this context we have performed theoretical and numerical considerations, providing some thresholds and a heuristic estimate of the number of elements that we may expect to have in the density gradient-based mesh for an arbitrary number of refinements n_{ref}. This way, we have shown that 3–4 refinements are sufficient to approach the maximum theoretical saving that our mesh can produce. Finally, we have also commented on the computational complexity of the procedure and we have provided numerical data that showed that performing the analysis in our density gradient-based mesh leads to a significant reduction in analysis times, while the time required for remeshing is, in comparison, modest.

Finally, we have solved several numerical experiments, both in the 2D and 3D settings. In this context, we have solved large-scale 3D problems with more than 20 million elements, where the efficiency of the solution of the FE analysis improved by 8.5 times when the density gradient-based adaptive mesh was used. We have shown that we compute almost identical designs when the analysis is solved on a uniform or on the density gradient-based mesh. This applies also to designs represented in fine design meshes, which are rich of fine features.

The formulation of the density gradient-based mesh is general and it can be adapted to different problems. This affords for various future works. For instance, it is possible to require that the analysis mesh is finer than the design along solid-void interfaces, which can be helpful in problems characterized by boundary layers. On the other hand, future works can also include the study of whether (and how much) it is possible to coarsen also the solid-void interfaces in smooth problems, in order to achieve a yet higher efficiency.

ACKNOWLEDGMENTS

The first author acknowledges the support of the grant A.004@ECDL@GALLIGANI and the assistance of Maria Cristina Murari to set up the computational resources for the 3D experiments via the Center for Scientific Computing of the University of Modena and Reggio Emilia. The second author wants to acknowledge the support from US NSF grant 1941206 and ONR grant N00014-18-1-2685. Open Access Funding provided by Universita degli Studi di Modena e Reggio Emilia within the CRUI-CARE Agreement. [Correction added on 27 May 2022, after first online publication: CRUI funding statement has been added.]

DATA AVAILABILITY STATEMENT

Data available on request from the authors.

ENDNOTES

[*] In practice, when not otherwise specified, we are going set $it_{ref} = 2$ in our experiments, and use the adaptive analysis from the beginning. Nonetheless, it is convenient to incorporate it_{ref} in the formulation to make it more flexible and because computing a density gradient-based

mesh at the first iteration is generally not reasonable, since the initial design does not come from a topology optimization step and a uniform density distribution is often considered.

ORCID

Francesco Mezzadri https://orcid.org/0000-0002-9779-7647
Xiaoping Qian https://orcid.org/0000-0002-9159-9877

REFERENCES

1. Bendsoe MP, Sigmund O. *Topology Optimization: Theory, Methods and Applications*. 2nd ed. Springer; 2004.
2. Borrvall T, Petersson J. Large-scale topology optimization in 3D using parallel computing. *Comput Methods Appl Mech Eng*. 2001;190:6201-6229.
3. Evgrafov A, Rupp CJ, Maute K, Dunn ML. Large-scale parallel topology optimization using a dual-primal substructuring solver. *Struct Multidiscip Optim*. 2008;36:329-245.
4. Wang S, de Sturler E, Paulino GH. Large-scale topology optimization using preconditioned Krylov subspace methods with recycling. *Int J Numer Methods Eng*. 2007;69:2441-2468.
5. Amir O, Stolpe M, Sigmund O. Efficient use of iterative solvers in nested topology optimization. *Struct Multidiscip Optim*. 2010;42:55-72.
6. Amir O, Bendsøe MP, Sigmund O. Approximate reanalysis in topology optimization. *Int J Numer Methods Eng*. 2009;78(12):1474-1491.
7. Salazar de Troya MA, Tortorelli DA. Adaptive mesh refinement in stress-constrained topology optimization. *Struct Multidiscip Optim*. 2018;58:2369-2386.
8. Salazar de Troya MA, Tortorelli DA. Three-dimensional adaptive mesh refinement in stress-constrained topology optimization. *Struct Multidiscip Optim*. 2020;62:2467-2479.
9. Micheletti S, Perotto S, Soli L. Topology optimization driven by anisotropic mesh adaptation: towards a free-form design. *Comput Struct*. 2019;214:60-72.
10. Ferro N, Micheletti S, Perotto S. Compliance–stress constrained mass minimization for topology optimization on anisotropic meshes. *SN Appl Sci*. 2020;2:1196.
11. Stainko R. An adaptive multilevel approach to the minimal compliance problem in topology optimization. *Commun Numer Methods Eng*. 2006;22(2):109-118.
12. de Sturler E, Paulino GH, Wang S. Topology optimization with adaptive mesh refinement. Proceedings of the 6th International Conference on Computation of Shell and Spatial Structures IASS-IACM 2008: 'Spanning Nano to Mega'; 2008; Ithaca, NY.
13. Maute K, Ramm E. Adaptive topology optimization. *Struct Optim*. 1995;10(2):100-112.
14. Bendsøe MP. Variable-topology optimization: statusand challenges. Proceedings of the European Conference on Computational Mechanics; 1999.
15. de Ruiter MJ, van Keulen F. Topology optimization using atopology description function. *Struct Multidiscip Optim*. 2004;26:406-416.
16. Guest JK, Smith Genut LC. Reducing dimensionality in topology optimization using adaptive design variable fields. *Int J Numer Methods Eng*. 2010;81:1019-1045.
17. Nguyen TH, Paulino GH, Song J, Le CH. A computational paradigm for multi-resolution topology optimization (MTOP). *Struct Multidiscip Optim*. 2010;41:525-539.
18. Parvizian J, Duster A, Rank E. Topology optimization using the finite cell method. *Optim Eng*. 2012;13(1):57-78.
19. Nguyen TH, Paulino GH, Song J, Le CH. Improving multi-resolution ropology optimization via multiple discretizations. *Int J Numer Methods Eng*. 2012;92:507-530.
20. Wang Y, Kang Z, He Q. Adaptive topology optimization with independent error control for separated displacement and density fields. *Comput Struct*. 2014;135:50-61.
21. Park J, Sutradhar A. A multi-resolution method for 3D multi-material topology optimization. *Comput Methods Appl Mech Eng*. 2015;285:571-586.
22. Groen JP, Langelaar M, Sigmund O, Ruess M. Higher-order multi-resolution topology optimization using the finite cell method. *Int J Numer Methods Eng*. 2017;110(10):903-920.
23. Gupta DK, van Keulen F, Langelaar M. Design and analysis adaptivity in multi-resolution topology optimization. *Int J Numer Methods Eng*. 2020;121:450-476.
24. Takezawa A, Kitamura M. Geometrical design of thermoelectric generators based on topology optimization. *Int J Numer Methods Eng*. 2013;90(11):1363-1392.
25. Vatanabe SL, Silva ECN. Design of phononic materials using multi-resolution topology optimization. Proceedings of the ASME International Mechanical Engineering Congress and Exposition; 2013.
26. Kelly DW, Gago JR, Zienkiewicz OC, Babuška I. A posteriori error analysis and adaptive processes in the finite element method. part I - error analysis. *Int J Numer Methods Eng*. 1983;19:1593-1619.
27. Noël L, Schmidt M, Messe C, Evans JA, Maute K. Adaptive level set topology optimization using hierarchical B-splines. *Struct Multidiscip Optim*. 2020;62:1669-1699.
28. Wang C, Qian X. A density gradient approach to topology optimization under design-dependent boundary loading. *J Comput Phys*. 2020;411:109398.

29. Qian X. Undercut and overhang angle control in topology optimization: a density gradient based integral approach. *Int J Numer Methods Eng*. 2017;111(3):247-272.
30. Mezzadri F, Qian X. A second-order measure of boundary oscillations for overhang control in topology optimization. *J Comput Phys*. 2020;410:109365 1-32.
31. Svanberg K. The method of moving asymptotes: a new method for structural optimization. *Int J Numer Methods Eng*. 1987;24:359-373.
32. Gupta DK, Langelaar M, van Keulen F. QR-patterns: artefacts in multi-resolution topology optimization. *Struct Multidiscip Optim*. 2018;58:1335-1350.
33. Sigmund O. On the design of compliant mechanisms using topology optimization. *Mech Struct Mech*. 1997;25(4):495-526.
34. Sigmund O. A 99 line topology optimization code written in MATLAB. *Struct Multidiscip Optim*. 2001;21:120-127.
35. Mezzadri F, Bouriakov V, Qian X. Topology optimization of self-supporting support structures for additive manufacturing. *Addit Manuf*. 2018;21:666-682.
36. Dapogny C, Faure A, Michailidis G, Allaire G, Couvelas A, Estevez R. Geometric constraints for shape and topology optimization in architectural design. *Comput Mech*. 2017;59:933-965.
37. Alnæs MS, Blechta J, Hake J, et al. The FEniCS project version 1.5. *Arch Numer Softw*. 2015;3(100):9-23.
38. Logg A, Mardal K, Wells GN, et al. *Automated Solution of Differential Equations by the Finite Element Method*. Lecture Notes in Computational Science and Engineering. Vol 84. Springer; 2012.
39. Bendsoe M. Optimal shape design as a material distribution problem. *Struct Optim*. 1989;1:193-202.
40. Woźniak M, Kuźnik K, Paszyński M, Calo VM, Pardo D. Computational cost estimates for parallel shared memory isogeometric multi-frontal solvers. *Comput Math Appl*. 2014;67(10):1864-1883.
41. Kawamoto A, Matsumori T, Yamasaki S, Nomura T, Kondoh T, Nishiwaki S. Heaviside projection based topology optimization by a PDE-filtered scalar function. *Struct Multidiscip Optim*. 2011;44(1):19-24.
42. Lazarov B, Sigmund O. Filters in topology optimization based on Helmholtz-type differential equations. *Int J Numer Methods Eng*. 2011;86(6):765-781.
43. Xu S, Cai Y, Cheng G. Volume preserving nonlinear density filter based on Heaviside functions. *Struct Multidiscip Optim*. 2010;41:495-505.
44. Balay S, Gropp WD, McInnes LC, Smith BF. Efficient management of parallelism in object oriented numerical software libraries. *Modern Software Tools in Scientific Computing*. Birkhäuser Press; 1997:163-202.
45. Balay S, Abhyankar S, Adams MF, et al. PETSc users manual. Technical report ANL-95/11 - revision 3.8, Argonne National Laboratory; 2017.

How to cite this article: Mezzadri F, Qian X. Density gradient-based adaptive refinement of analysis mesh for efficient multiresolution topology optimization. *Int J Numer Methods Eng*. 2022;123(2):465-504. doi: 10.1002/nme.6863