Additive Manufacturing 21 (2018) 666–682

Contents lists available at ScienceDirect

Additive Manufacturing

journal homepage: www.elsevier.com/locate/addma

Topology optimization of self-supporting support structures for additive manufacturing

Francesco Mezzadri, Vladimir Bouriakov, Xiaoping Qian*

Department of Mechanical Engineering, University of Wisconsin-Madison, Madison, WI 53706 – 1572, United States

ARTICLE INFO

Keywords:
Additive manufacturing
Topology optimization
Support structure

ABSTRACT

In this paper, we formulate the generation of support structures for additive manufacturing as a topology optimization problem. Compared with usual geometric considerations based support structure design, this formulation affords mechanistic meaning to the computed support structures. Moreover, our study reveals that the topology optimization formulation generally leads to self-supporting designs without extraneous self-supporting constraints. We show the generality of the procedure by computing supports for a variety of parts in both two and three dimensions, including a complex model of the mascot of the University of Wisconsin-Madison. The resulting support structures have been 3D printed, demonstrating that the computed designs can successfully be used as supports.

1. Introduction

In recent years, additive manufacturing has become increasingly important in several fields. A distinctive feature that made additive manufacturing popular is its ability to produce complex parts easily and with no part-specific fixturing or tooling. For this reason, additive manufacturing is attractive, for instance, for producing parts obtained from topology optimization, which are frequently of complex shape and with internal voids.

On the other hand, some drawbacks of additive manufacturing exist as well. Indeed, the very idea at the basis of additive manufacturing consists in building a part layer by layer. This makes so that long downward-facing surfaces need to be supported by some other structure which will then have to be removed after the printing. These “support structures” are thus made of sacrificial material which is basically wasted (recycling is limited to a few times before needing re-polymerization) and their removal adds a post-processing step.

This problem can be handled in different ways, depending on whether we can modify the part to print or not. In the first case, we can try to modify parts by removing all overhangs which are not self-supporting. This is, for instance, the trend that has been recently followed by the topology optimization community. Thus, several manufacturability constraints and filters have been introduced in topology optimization frameworks to compute optimal designs which are free of non self-supporting surfaces. Since the first work in this direction [1], strategies to achieve manufacturability in a topology optimization context have greatly evolved, allowing, for instance, to compute the optimal design for arbitrary choices of the critical overhang angle. For instance, [2] introduced an overhang constraint acting on the directional gradient of density along the build direction, allowing an arbitrary choice of both build direction and critical overhang angle. In [3], instead, the constraint is based on the volume of the support structures, while other approaches rely on filter-based overhang restrictions [4,5]. Overhang control has then also been analyzed in the framework of shape optimization by [6]. Finally, it is worth citing also the first real implementations of a overhang control procedure [7,8] as well as notable recent contributions like [9] and [10].

However, sometimes we may need to print a part with some given specifics. In this case, we cannot modify the part to print. In this second case, we are bound to use support structures. The drawbacks associated with the use of supports are then handled by using as little support material as possible and by making their subsequent removal easier. In this regard, first it is important to choose a suitable orientation of part building, so to reduce as much as possible the area of the surfaces needing a support. This can be done, for instance, by the algorithm presented in [11]. Then, several strategies can be used to reduce the amount of material employed in building the supports. In this regard, it is worth citing [12], where supports are made of cellular structures, and [13], where supports are generated geometrically and are constituted by tree-like structures which touch the surfaces to support in points chosen by sampling. Finally, the interface between the support and the part to print can be modified so to ease the removal of the support itself. In this regard, tips can be added at the ends of the supports, as done, for instance, also in [13].

* Corresponding author.
E-mail address: qian@engr.wisc.edu (X. Qian).

https://doi.org/10.1016/j.addma.2018.04.016
Received 9 March 2018; Received in revised form 10 April 2018; Accepted 11 April 2018
Available online 19 April 2018

However, the procedures currently used to generate supports are usually geometry based [14], do not have mechanistic meaning and are likely not optimal in any sense. There are only a few exceptions in the recent literature. For instance, in [15] supports are involved in a topology optimization framework together with the part to build, but the aim is not to optimize supports, but to find trade-off solutions accounting of both performance of the part and costs of support structure. Similar considerations can be made on [16], which includes also orientation. In [17], the authors instead consider a multi-objective optimization for support structures, introducing a repulsion index which is used to ease the removal of the computed supports. However, the approach is analyzed only in 2D examples, the results appear to have not self-supporting parts and the actual printability of the computed structures is not verified.

We, on the other hand, present a simpler procedure which does not rely on multi-objective optimization. Indeed, since easy removal acts only at the interfaces between the support and the part to print, we can simply modify these interfaces adding tips at the ends of the supports, as mentioned above. We nonetheless formulate the generation of self-supporting support structures as a topology optimization problem, giving mechanistic meaning to the computed supports and exploiting all the advantages associated with the use of topology optimization. We analyze the computed supports in both 2D and 3D examples and we actually 3D-print several computed structures, thus demonstrating their printability. Moreover, we notice that the optimized supports usually present the remarkable characteristic of being directly self-supporting, without needing any extraneous overhang control. In rare cases where this is not true, we can instead use the overhang constraints previously outlined to ensure that the optimized supports are self-supporting, thus bridging, in a way, the two approaches for handling overhangs that we outlined above.

Thus, we compute supports that

- have direct mechanistic meaning. In particular, they are characterized by maximum stiffness for a given volume fraction;
- theoretically use less material than support structures generated by the geometric approaches for achieving the same stiffness. In actual computations, we may fall in a local minimum, but the optimization procedure generally ensures better results than geometric-based methods;
- are usually directly self-supporting, without extraneous self-supporting constraints. In case the supports present critical overhangs, manufacturability can nonetheless be ensured by overhang control strategies recently introduced in topology optimization;
- have features compatible with the 3D printer's resolution by the use of length-scale control.

We remark that, throughout the paper, topology optimization is employed to optimize the support structure for a given design of the end part. In the following, by “design” we then refer to the scaffold structures generated by the topology optimization problem.

In Section 2 we outline formulate the topology optimization problem. In particular, first we outline the reasons how we choose loading, boundary conditions and cost functional. All these choices are performed by trying to reproduce the exact loading and boundary conditions which occur while printing a part with some overhangs, while the cost functional is chosen by considering what the main purpose of a support structure is. Then, we present more rigorously the mathematical formulation of the topology optimization problem, consisting in a volume-constrained compliance minimization problem (or, analogously, in a compliance-constrained volume minimization problem). An interesting analogy with the topology optimization of bridges and of roof supports can also be performed. The formulation of the problem is indeed really similar (as it can be expected also from the similarity of the roles performed by a roof support and by a support structure) with, however, an important difference in the location of zero-displacement boundary conditions which greatly affects the self-support nature of the optimized designs.

We then introduce several test problems in both 2D and 3D domains to demonstrate that this topology optimization problem can be successfully used to compute supports in a variety of situations. We do this in Section 3, which also contains some details on the implementation of the optimization problems.

Section 4 is then devoted to a thorough analysis of a test problem in a 2D rectangular domain. Experimental results are here presented together with mechanical considerations, including, for instance, an analogy with transmissible loads. A remarkable feature of the computed designs are their tree-like structural shapes which partially resemble bridges, roof supports and the supports for additive manufacturing in [13]. Interestingly, tree-like designs similar to the ones arising from our optimization can be also commonly found in architecture in several structures not coming from a topology optimization procedure, such as Gothic ribbed vaults in the Medieval period or, more recently, branched pillars like those of the Sagrada Familia in Barcelona or of many other structures (e.g. see [18]). In this context, it is also worth to mention the work on the structural meaning of tree-like structures by the architect Frei Otto [18,19]. We also analyze self-support (discussing also when it may not hold and the remedies), the influence of the maximum volume fraction and of the aspect ratio of the domain, the effect of vertical zero-displacement boundary conditions and cases with non-uniform loads. Some remarks on length-scale control are given as well, noticing how we can use minimum length-scale control to match the fineness of the feature of the optimized design with the resolution of 3D printers.

In Section 5 we instead consider more complex examples, starting from 2D curved domains. We then compute the optimized supports for popular structures, such as the MBB beam and the cantilever beam. Finally, we pass to analogous 3D examples, where particular attention is devoted to extending this approach to the optimization of practical 3D support structures where large-scale computing is used. This allows to compute supports characterized by fine features. We also 3D print the computed results to demonstrate that the designs are self-supporting and that they can indeed acts as support structures. We then also compare the amount of material used for building the supports with our strategy and with existing software.

In Section 6 we then optimize the support for a complex model of the mascot of the University of Wisconsin-Madison. We use this as a test-problem to show the ability of our procedure to compute support structures for complex geometries. Here large scale optimization is particularly relevant: some of the presented results have been obtained performing the optimization with more than 1 billion variables.

Lastly, Section 7 concludes this work.

2. The topology optimization problem

To formulate the generation of support structures as a topology optimization problem we must define:

- the design domain where the topology optimization is conducted;
- loading and boundary conditions;
- a cost functional.

In this section, we show how domain and loading/boundary conditions can be deduced by the analysis of the structure that we want to print. The cost functional, instead, must be a measure that tells us in which sense the computed structure is “optimal”. Thus, it must have a physical meaning which is consistent with the purpose of our optimization. After identifying all these components, we finally present a more rigorous formulation of the topology optimization problem of our interest.

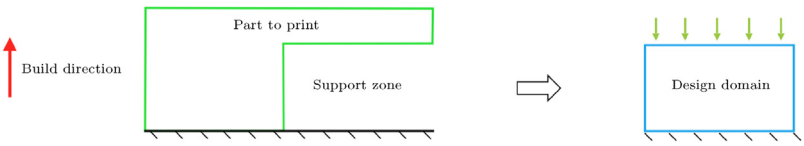


Fig. 1. Given a part to be printed (in green), a support structure will be built in the "support zone" between the downward-facing surfaces and the printing plate. This is also the domain where the optimization will take place. The printing plate gives zero-displacement boundary conditions along all the bottom of the domain, while the surface to support gives a distributed load along the top.

2.1. Outline of loading and boundary conditions

We now define loading and boundary conditions of the topology optimization problem of our interest referring to a simplified setting, which can then be easily generalized. Consider Fig. 1, which represents a simple overhanging part that we may want to print. Suppose that the orientation is provided and that the part must be printed along a given build direction.

The supports will evidently be built in the zone underlying the part to support. Thus, this "support zone" gives us directly the design domain.

Limiting ourselves to a purely mechanical optimization, the only load which the support must withstand is given by the weight of the part which must be supported. This is represented by a distributed load on the top of the design domain.

Lastly, supposing, for simplicity, that the support structures can be built only from the base where the part is being built, the boundary conditions are given by a zero-displacement condition on the entire bottom part of the domain.

The resulting design domain, together with loading and boundary conditions, is represented on the right of Fig. 1.

As mentioned earlier, this simplified setting can be easily generalized. For instance, support structures may be built not only from the printing plate, but also from non-overhanging regions which have already been printed. This could help to further reduce the amount of material employed in the supports and it is also what happens, for instance, every time the supports are located inside a hole of the part to print. These cases can be easily reproduced simply by introducing a zero-displacement boundary condition along all those boundaries where the supports can be generated.

For instance, in the example in Fig. 1, supports may be generated from the entire left side of the domain as well, attached to the vertical boundary of the part in green. All this area is indeed printed before the zone which needs the supports. If we want to allow this situation, we can then add a zero-displacement boundary condition along the entire left boundary.

2.2. Formulation of the problem

To choose suitable cost and constraints, we must consider that limiting the amount of used material is of the utmost importance in the design of support structures. Indeed, the supports are removed after printing is complete and the material we employ in their construction is, thus, ultimately wasted. Volume fraction must then be involved in the formulation of the problem, as the cost functional or as a constraint.

Furthermore, we also need a mechanistic meaning consistent with the purpose of sustaining the weight of overhanging regions. In particular, it is certainly desirable that the support structure is sufficiently stiff to support the overhang without excessive deformations.

Taking into account these considerations, we can formulate two optimization problems, depending on whether a maximum volume fraction or a maximum compliance is given:

- if a maximum volume fraction is given, it is suitable to choose a volume-constrained compliance minimization problem;
- conversely, if we are given a minimum required stiffness, we can directly minimize the amount of wasted material by a compliance-constrained volume minimization problem.

More formally, let Ω be the domain identified by the support zone (e.g. the area bounded in blue in Fig. 1) and let Γ be its boundary. Furthermore, let Γ_D be the part of the boundary where the supports can be built and let Γ_N be the rest of the boundary. Given a prescribed maximum volume fraction $\overline{V}$ or a prescribed maximum compliance $\overline{C}$, we can consider the following optimization problems:

Volume constrained compliance minimization

$$\begin{aligned} &\text{minimize } J_C= && \int_\Omega \nabla \boldsymbol{u}\cdot E(\gamma)\boldsymbol{s}(\boldsymbol{u})\mathrm{d}\Omega \\ &\text{s. t.\&} -\nabla\cdot E(\gamma)\boldsymbol{s}(\boldsymbol{u}) = \boldsymbol{f} \quad \text{on } \Omega && \\ & && \tfrac{1}{V}\int_\Omega \gamma \mathrm{d}\Omega \le \overline{V} \end{aligned} \tag{1}$$

Compliance constrained volume minimization

$$\begin{aligned} &\text{minimize } J_V= && \tfrac{1}{V}\int_\Omega \gamma \mathrm{d}\Omega \\ &\text{s. t.} && -\nabla\cdot E(\gamma)\boldsymbol{s}(\boldsymbol{u}) = \boldsymbol{f} \quad \text{on } \Omega \\ & && \int_\Omega \nabla \boldsymbol{u}\cdot E(\gamma)\boldsymbol{s}(\boldsymbol{u})\mathrm{d}\Omega \le \overline{C} \end{aligned} \tag{2}$$

where γ is the density, V denotes the volume, $\boldsymbol{u}$ is the displacement vector and $E(\gamma)$ is the Young's modulus expressed by the SIMP method [20,21]

$$E(\gamma) = E_{\min} + (E_{\max} - E_{\min})\gamma^p,$$

where in this paper, we set $p = 3$, $E_{\max} = 1$ and $E_{\min} = 10^{-9}$.

Finally,

$$\boldsymbol{\sigma}(\boldsymbol{u}) = 2\mu\nabla\boldsymbol{u} + \lambda I \mathrm{Tr}(\nabla\boldsymbol{u}) = E(\gamma)\boldsymbol{s}(\boldsymbol{u}) \tag{3}$$

where I is the identity matrix, $\nabla\boldsymbol{u}$ represents the symmetric gradient and μ, λ are the first and the second Lamé's parameters, respectively. These parameters are dependent on the Poisson's ratio ν and on the Young modulus $E(\gamma)$, which is collected for clearness.

In both optimization problems, the state equation is provided with the boundary conditions

$$\begin{aligned} &\boldsymbol{u} = \boldsymbol{0} && \text{on } \Gamma_D \\ &E(\gamma)\boldsymbol{s}(\boldsymbol{u})\cdot\boldsymbol{n} = \boldsymbol{u}_N && \text{on } \Gamma_N, \end{aligned} \tag{4}$$

representing the loads and the zero-displacement conditions introduced at the beginning of this section.

In the following, the cost is chosen to be the compliance and we thus consider the problem (1). Indeed, volume-constrained compliance minimization problems are vastly studied and comparisons with other results in the literature are straightforward. Nonetheless, this does not affect the generality of our analysis. Indeed, under mild hypotheses, we can pass from the solution of a compliance minimization problem to that of a volume minimization problem simply by a scaling [22,p. 88]. Thus, the presented results are equally applicable to a compliance-constrained volume minimization problem as well.

Therefore, topology optimization of support structures can be formulated by a well-known topology optimization problem, which is nonetheless rich in mechanical meaning.

It is also worth noticing the similarity of this problem with compliance minimization for bridges and of roof supports. As a consequence, the support structures computed with our approach all present tree-like structures resembling the famous roof supports computed by topology optimization for the Qatar Convention Center [23], the structures arising in topology optimized bridges (see, for instance, the bridge problem analyzed in [24]) and other roof supports (e.g. see also [25,26]). An important difference is however given by the position of zero-displacement boundary conditions, which in bridges and roof

support typically involve only a few points of the boundary. On the other hand, we here have zero-displacement boundary conditions on large parts of the boundary. This means that in our case the pillars of the "bridges" which support the overhangs can be placed everywhere on the printing plate. This freedom in choosing the position of the pillars, in turn, fosters an analogy with transmissible loads, which helps explain why the computed supports are generally self-supporting, as analyzed in the following.

In this context, it is also worth highlighting that we here limited ourselves to the mechanical aspect of the problem: i.e. supporting the weight of the build part. Indeed, it is indispensable that a support structure successfully bears the overhanging layers of material before their solidification. Nonetheless, non-mechanical factors, such as considering thermally conductive pathways to conduct heat in e.g. laser-based processes, and for limiting deformations due to the build-up of residual stress in the printed part, are also important in the design of support structures. These additional factors will be studied in future work.

2.3. Sensitivity analysis

For completeness, we now report also the sensitivity analysis of the compliance minimization problem formulated above. In this regard, let us introduce the spaces of trial and test functions, which, in a two-dimensional case, are defined by

$$\text{Trial space: } \mathscr{V} = \{\boldsymbol{u} \in [H^1(\Omega)]^2 : \boldsymbol{u} = \boldsymbol{u}_0 \text{ on } \Gamma_D\} \tag{5}$$

$$\text{Test space: } \widetilde{\mathscr{V}} = \{\tilde{\boldsymbol{u}} \in [H^1(\Omega)]^2 : \tilde{\boldsymbol{u}} = \boldsymbol{0} \text{ on } \Gamma_D\}. \tag{6}$$

We then multiply the state equation by $\tilde{\boldsymbol{u}} \in \widetilde{\mathscr{V}}$ and integrate applying the given boundary conditions. In this way, we get the weak form of the problem:

$$J = \langle \nabla \boldsymbol{u}, E(\gamma)\boldsymbol{s}(\boldsymbol{u})\rangle_\Omega \tag{7a}$$

$$\langle \nabla \tilde{\boldsymbol{u}}, E(\gamma)\boldsymbol{s}(\boldsymbol{u})\rangle_\Omega = \langle \tilde{\boldsymbol{u}}, \boldsymbol{f}\rangle_\Omega + \langle \boldsymbol{u}_N, \tilde{\boldsymbol{u}}\rangle_{\Gamma_N} \quad \forall\, \tilde{\boldsymbol{u}} \in \widetilde{\mathscr{V}} \tag{7b}$$

$$\langle \gamma, 1\rangle_\Omega / V \le \overline{V}. \tag{7c}$$

Here $\langle\cdot,\cdot\rangle_\Omega$ and $\langle\cdot,\cdot\rangle_{\Gamma_N}$ denote the l^2-inner product on the domain and on the Neumann boundary, respectively.

We can now define the optimality conditions of the problem. The Lagrangian function $\mathscr{L}$ for the problem of minimizing the compliance (7a) under the PDE constraint (7b) can be written as

$$\mathscr{L} = \langle \nabla \boldsymbol{u}, E(\gamma)\boldsymbol{s}(\boldsymbol{u})\rangle_\Omega + \langle \nabla \boldsymbol{v}, E(\gamma)\boldsymbol{s}(\boldsymbol{u})\rangle_\Omega - \langle \boldsymbol{v}, \boldsymbol{f}\rangle_\Omega - \langle \boldsymbol{u}_N, \boldsymbol{v}\rangle_{\Gamma_N}, \tag{8}$$

where $\boldsymbol{v} \in \widetilde{\mathscr{V}}$ is the Lagrange multiplier for the elastic equilibrium equation (7b).

Imposing the first order optimality conditions, we then get the adjoint equation

$$\begin{aligned}\mathscr{L}_{\boldsymbol{u};\tilde{\boldsymbol{u}}} &= \langle \nabla \tilde{\boldsymbol{u}}, E(\gamma)\boldsymbol{s}(\boldsymbol{u})\rangle_\Omega + \langle \nabla \boldsymbol{u}, E(\gamma)\boldsymbol{s}(\tilde{\boldsymbol{u}})\rangle_\Omega + \langle \nabla \boldsymbol{v}, E(\gamma)\boldsymbol{s}(\tilde{\boldsymbol{u}})\rangle_\Omega \\ &= 0 \quad \forall\, \tilde{\boldsymbol{u}} \in \widetilde{\mathscr{V}},\end{aligned} \tag{9}$$

where $\mathscr{L}_{\boldsymbol{u};\tilde{\boldsymbol{u}}}$ denotes the directional derivative of $\mathscr{L}$ with respect to $\boldsymbol{u}$ along $\tilde{\boldsymbol{u}}$. Solving the adjoint equation for $\boldsymbol{v} \in \widetilde{\mathscr{V}}$, we thus find the adjoint variable.

It is then easy to compute the directional derivative of the Lagrangian and of the integral constraints with respect to the design variable γ along the test function $\tilde{\gamma}_t$. Indeed, calling $E'(\gamma) = p(E_{\max} - E_{\min})\gamma^{p-1}$, we find

$$\mathscr{L}_{\gamma;\tilde{\gamma}_t} = \langle \nabla \boldsymbol{u}, E'(\gamma)\boldsymbol{s}(\boldsymbol{u})\tilde{\gamma}_t\rangle_\Omega + \langle \nabla \boldsymbol{v}, E'(\gamma)\boldsymbol{s}(\boldsymbol{u})\rangle_\Omega \quad \text{Cost sensitivity} \tag{10a}$$

$$V_{\gamma;\tilde{\gamma}_t} = \frac{\langle \tilde{\gamma}_t, 1\rangle_\Omega}{V} \quad \text{Volume sensitivity.} \tag{10b}$$

2.4. Formulation with PUP constraint

As already stated, in the problems of our interest usually we do not need any extraneous overhang constraint to compute self-supporting structures. Thus, the optimization problems introduced in the previous subsections are sufficient to compute self-supporting support structures.

If this is not true, we can rely on one of the overhang constraints or filters existing in the literature. In these cases, in this paper we use the PUP constraint. Therefore, we here summarily describe the optimization problem with this additional constraint. For details, the reader is referred to [2].

The PUP formulation requires density filtering, here performed with an Helmholtz PDE filter [27,28], and Heaviside filtering [29]. We denote the filtered density field by $\tilde{\gamma}$. The PUP constraint is then defined as

$$\int_\Omega H\left(\boldsymbol{b}\cdot\frac{\nabla\tilde{\gamma}}{\|\nabla\tilde{\gamma}\|_2} - \cos\overline{\alpha}\right)\boldsymbol{b}\cdot\nabla\tilde{\gamma}\,\mathrm{d}\Omega \le \overline{P}_{\overline{\alpha}} \tag{11}$$

where $\boldsymbol{b}$ is the build direction, $\overline{\alpha}$ is the critical overhang angle, $H\left(\boldsymbol{b}\cdot\frac{\nabla\tilde{\gamma}}{\|\nabla\tilde{\gamma}\|_2} - \cos\overline{\alpha}\right)$ is a shifted Heaviside projection of the undercut perimeter and $\overline{P}_{\overline{\alpha}}$ is the maximum allowed projected perimeter.

Finally, the PUP formulation also includes a grayness constraint

$$\frac{1}{\overline{V}}\int_\Omega 4\tilde{\gamma}(1-\tilde{\gamma})\mathrm{d}\Omega \le \overline{\varepsilon} \tag{12}$$

where $\overline{\varepsilon}$ is the threshold to intermediate densities. This constraint is needed to mitigate intermediate densities associated with the large filter radius which is often required to avoid oscillations.

3. Implementation of the topology optimization problem

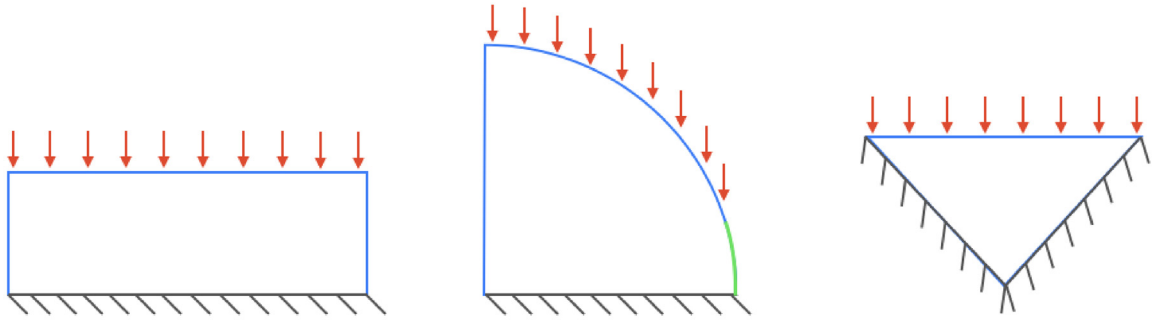
In the previous sections, we showed how topology optimization of support structures can be formulated as a compliance minimization problem. In order to demonstrate that the computed optimized designs can indeed be used as supports, we now need to implement the topology optimization problem. In this context, we here present a variety of overhang situations, which we use to show the generality of our approach. Lastly, we also present some details on the implementation of the problems themselves.

3.1. Description of the test problems

We show the applicability of our approach to computing the support structures for several different overhangs. In this regard, both 2D and 3D examples are considered.

Some relevant 2D problems are represented in Fig. 2. In particular, Fig. 2a represents a simple 2D test problem in a rectangular domain. Conceptually, it represents the situation already introduced in Fig. 1 and we use it in the next section to validate the procedure. Then, we pass to more complex geometries. So, Fig. 2b represents a problem with a curved overhanging boundary, where the arc is assumed to be self-supporting up to an height of 0.3 times its radius (the boundary is indeed almost vertical in that zone). Finally, Fig. 2c represents a problem corresponding to computing supports of a triangular hole such as the one which arises in the MBB beam.

We perform 3D topology optimization of support structures as well. Analogously to the 2D cases, we again consider various situations. In particular, first we analyze optimized supports for a dome and for a 3D MBB-beam. Then, we compute the support structures for a complex 3D model of "Bucky Badger", the mascot of the University of Wisconsin-Madison. This complex part is represented in Fig. 3. This example in meant to demonstrate the generality of our approach and its applicability to printing complex real-world structures. Regarding orientation, the model is meant to be printed on its back, as in Fig. 3b. The surface which needs to be supported is thus characterized by curved



(a) Topology optimization of support structures in a rectangular design domain

(b) Design domain with curved overhang. Boundary in green is self-supporting

(c) Topology optimization of support structures in a triangular hole

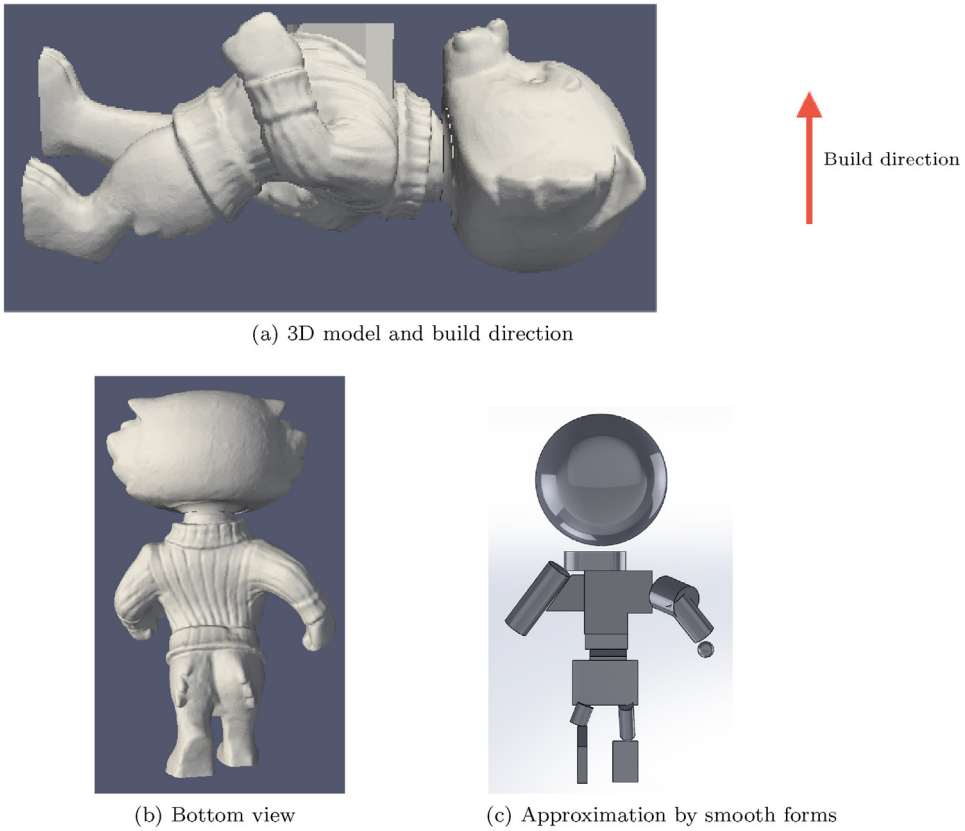
Fig. 2. Loading, boundary conditions and design domains of some 2D topology optimization examples.

and tilted regions one next to the other. In order to simplify the definition of these forms, with no lack of generality we approximated the back of Bucky by elementary functions, such as the surfaces of cylinders, planes and spheres, as in Fig. 3c. Then, we defined the loads on these surfaces and run the optimization.

Loads are uniform and set equal to one. Operatively, we integrate a unitary, downward-facing load over the loaded boundaries represented in Fig. 2. The Young's modulus of the material is set to 1 (and it is set to 10^{-9} in the void) and the Poisson's ratio is set to 0.3 for all problems. In this context, it is nonetheless also worth remarking that the absolute values of load and Youngs modulus are not important for generating the support designs.

3.2. Details of the implementation

The 2-dimensional problems have been solved in the FEniCS framework [30,31] using the Method of Moving Asymptotes (MMA) as optimizer [32] with standard parameters. The discretization has been performed by T3 elements. In 2D problems without extraneous overhang constraints, we also controlled the minimum length-scale by a robust filter (see [33]; for selection of filter size, the reader may instead refer to [34]). This can be interesting since we can use length-scale control to match the fineness of the features of the optimized support with the resolution of any 3D printer. In the following, r_f denotes the filter radius characterized in terms of the number of elements involved in the filtering, while r denotes the actual radius of the filter. The plot of the results is provided directly by FEniCS and reported with no modification.



(a) 3D model and build direction

(b) Bottom view

(c) Approximation by smooth forms

Fig. 3. Model of the mascot of the University of Wisconsin-Madison viewed from the side where supports will be built (b) and approximation by smooth forms (c) to simplify the implementation in TopOpt.

Three-dimensional problems have been solved using the TopOpt code in [35], which is an efficient, parallel C++ code for solving topology optimization problems. These properties are particularly attractive for dealing with the higher complexity of 3D problems. We adapted the code to the domains and to the boundary/loading conditions of the problems of our interest. We ran the optimization procedures on the cluster of computers of the UW-Madison Center For High Throughput Computing (CHTC). In this case, the discretization was performed by hexagonal elements. The results of the optimization were stored in.vtu files. The figures here reported and the.stl files used for printing the results have been obtained by elaborating the.vtu files in Paraview. The resulting topologies have been extracted using the "Iso Volume" filter, filtering to solid all densities larger than 0.2 or 0.5, depending on the problem.

4. Validation of the procedure: self-support and mechanical considerations for a simple 2D example

In this section, we analyze the simple 2D problem represented in Fig. 2a. For a first validation of the procedure, we consider a 3×1 rectangular domain and analyze how the optimized structures evolve as the maximum volume fraction is changed. In particular, as the maximum volume fraction is reduced, tree-like structures appear. After commenting on these structures and introducing some mechanical considerations, we consider what happens as we change the aspect ratio of the domain. In this context, we observe that we again obtain tree-like structures (which are characteristic also of all the other design domains, as we will see in the following) and analyze the problem of self-support. This is particularly important, since, as mentioned earlier, the optimized support structures can be practically used only if, in turn, they do not need other supports. We observe that, generally, self-support is ensured directly by the nature of the problem. We substantiate this by mechanical considerations and by an analogy with transmissible loads. In case non self-supporting parts are present, we can instead exploit existing tools for overhang constrained topology optimization. Indeed, we can simply forbid critical overhangs by using some manufacturability constraint, as we show later in this section. Finally, we also consider more general settings of loading and boundary conditions, as well as the effect of length-scale control.

4.1. Effect of the volume fraction

In topology optimization of supports, the ultimate purpose is to compute the structure that optimally transfers the load from an overhanging region to the printing plate (or to some other part that has already been printed). Let us consider a simplified 2D case where supports can be built only starting from the printing plate and loads are uniform. Then (considering, for simplicity and with no loss of generality, a domain oriented along the build direction) we always have a distributed load which must be transferred from some upper boundary to the bottom boundary of the domain itself, where the printing plate is located. Thus, the ideal load transfer occurs along a straight line connecting the top region where the load is applied to the bottom of the domain. If the load is applied to the entire top boundary, the optimal support would then be a single straight pillar covering the entire domain.

Of course, the result of the optimization cannot be a completely full structure when we impose a volume constraint as well. In this case, some holes must appear. Let us then consider a 2D problem in a 3×1 rectangular domain, like the one in Fig. 2a and see what happens as the maximum volume fraction V_f is changed. The results, obtained discretizing the domain by 76,800 T3 elements, are reported in Fig. 4. Starting with $V_f = 90\%$, we obtain the design in Fig. 4a, which is almost completely full, as expected. As the volume constraint gets tighter, more holes appear. In particular, at $V_f = 60\%$ (Fig. 4c) the design is characterized by three arcs separated by straight pillars. This design does not change dramatically when the maximum volume fraction is further reduced: of course, the arcs get wider, but we always get exactly three arcs for all $V_f \leq 60\%$. Then, in Fig. 4d start appearing some branches, which become evident in Fig. 4e, corresponding to $V_f = 30\%$. We therefore obtain tree-like support structures.

Here, trunks and branches play two different roles. Indeed, branches connect zones at the top of the domain to other branches and, ultimately, to a trunk, to which they transfer the load coming from a relatively small part of the boundary. Trunks are instead pillars with a prevalently structural function, supporting the load coming from several branches and transferring it directly to the printing plate.

Intuitively, we can already expect that the number of trunks cannot get too small also when the maximum volume fraction gets really low. Indeed,

- we always need some trunk to support the load coming from the branches;
- structures with too few trunks would likely require more material. Indeed, branches would need be really long to connect all the parts of the top boundary to the nearest trunk;
- long branches would also transfer the load less efficiently. Indeed, if the nearest straight trunk is far, they would be tilted more horizontally than vertically.

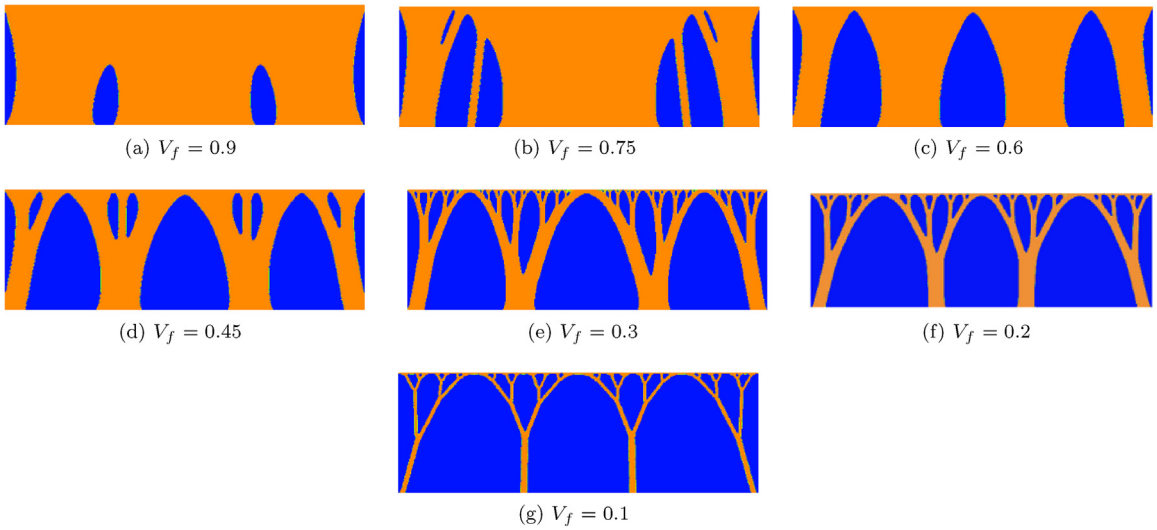
This gives a first, intuitive explanation of why the computed structures are generally self-supporting. Indeed, branches efficiently transfer load only if they are not too tilted with respect to the build direction. In particular, if they are tilted by an angle larger than 45° with respect to the build direction, they are more horizontal than vertical, while load must be transferred vertically, from top to bottom of the domain. Therefore, since trunks can grow everywhere on the bottom boundary, it is reasonable to think that the optimization will place them so that branches are prevalently vertical. This is a major difference with respect to usual topology optimization of bridges and roof supports and highlights the importance of the chosen boundary conditions.

Regarding self-support, we notice that horizontal regions can appear at the top of the arches, but they are nonetheless short and can be considered self-supporting, as demonstrated by 3D-printed examples presented in the following sections. This appears even more so reasonable considering that we do not need high surface quality in supports and that additive manufacturing is typically employed for printing parts which are not excessively large. Nonetheless, if we had a large domain and if we needed to reduce these horizontal parts, we could always do so by increasing the volume fraction.

4.2. Effect of aspect ratio

The previous considerations apply also when the aspect ratio of the domain is changed. Indeed, repeating the reasoning of the previous subsection, we can intuitively expect that the optimal structures will always have "enough" trunks (whose number depends on the aspect ratio of the domain) and relatively short, mostly vertical branches. This is confirmed by the results in Fig. 5, where the maximum volume fraction is fixed at 30% and the aspect ratio of the design domain is changed. In the following, we denote the horizontal space variable by x. The discretization has been performed using about the same number of T3 elements as in the 3×1 domain (76, 800). Small variations in the number of elements were however unavoidable as the aspect ratio was changed.

Again, in all cases we obtain tree-like structures and they appear to be self-supporting. As the aspect ratio is changed, the number of arcs and pillars changes, but the general design remains the same. This substantiates the claim that the optimized structures are generally self-supporting and the number of arcs which appear in the various cases confirms the considerations made in the previous subsection.



(a) $V_f = 0.9$ (b) $V_f = 0.75$ (c) $V_f = 0.6$

(d) $V_f = 0.45$ (e) $V_f = 0.3$ (f) $V_f = 0.2$

(g) $V_f = 0.1$

Fig. 4. Optimized topology as the maximum volume fraction V_f is changed.

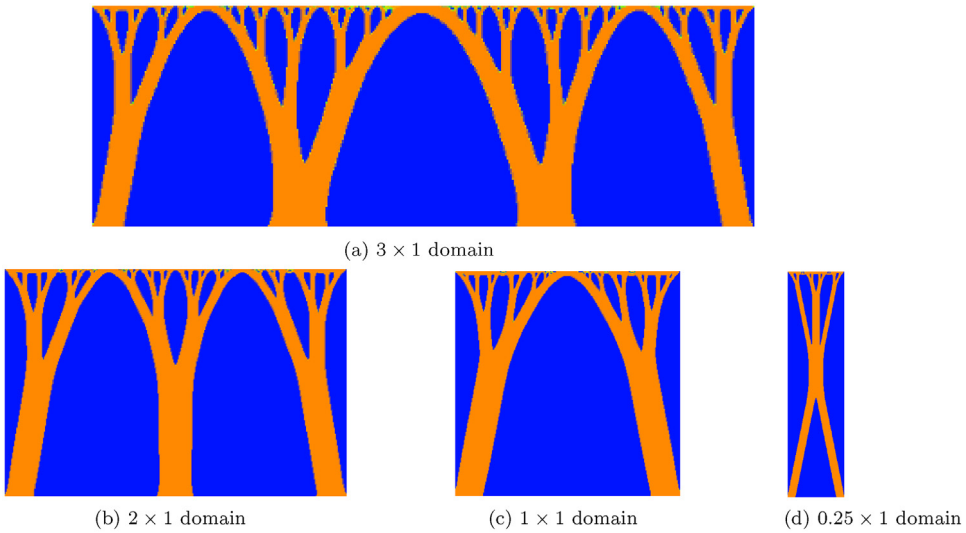
Moreover, we can also perform an analogy with transmissible loads. In problems with transmissible loads, the aim is to optimize with respect to the position of the load as well. Indeed, in these problems, the precise location of the load is variable: only the magnitude of the load and the line along which it acts are given. Thus, it is as if were computing an optimal structure while letting the optimization find the best position of the load along its line of action. This case has been extensively studied, in rectangular domains, in [36]. In particular, the authors found that the optimal solution for minimum compliance of a central point-load is a "triangle" whose sides form a 45° angle with the direction along which the load is applied. In case of distributed load, more similar to the cases of our interest, the optimization leads to an analogous parabolic arch structure.

In the problems of our concern, the load is fixed, but we have zero-displacement boundary conditions all over the bottom boundary of the domain. This means that the supports can be built so that the load is in an optimal position. Indeed, the number of trunks and the distance between them determines how trunks and branches are tilted with respect to the load. Thus, by analogy with the problem of transmissible loads and with results in [36], we expect that the optimal structure is given by arcs with a 45° inclination. This is indeed what happens. So, the optimal support in a 1×1 domain is the single arc of Fig. 5c, which is, moreover, similar to the result for transmissible loads in [36]. If the domain is 2×1, we instead have 2 arcs as in Fig. 5b, and so on for a 3×1 domain (Fig. 5a) and for wider aspect ratios. If the domain is instead shorter along x, it is not possible to place the trunks optimally with respect to the load. So, we have a single arc with an inclination smaller than 45 degrees, as in Fig. 5d.

The results in Fig. 5 also give some important information on the scalability of the problem. Indeed, for long, 2D rectangular domains, we may think to simply reproduce the optimal design for the 1×1 domain in Fig. 5c as many times as needed instead of actually running the optimization process.

4.3. A non self-supporting transition case

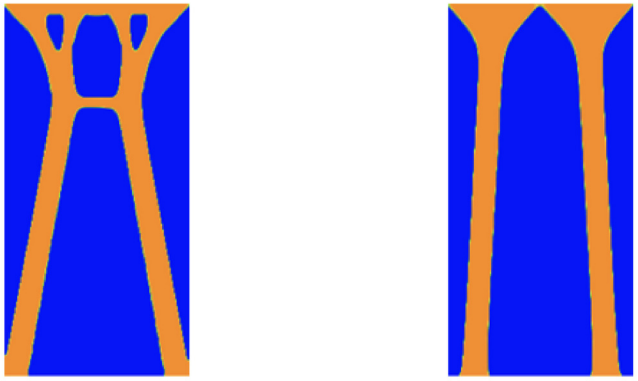
In rare cases, however, we might still have some non self-supporting overhangs. For 2D rectangular domains, this happens in "transition cases" between long aspect ratio (like the ones in Figs. 5a–c) and short aspect ratio (like Figure 5d). In these cases, x is not too small and some overhanging part can appear between the two pillars of the only arc which is present.



(a) 3×1 domain

(b) 2×1 domain (c) 1×1 domain (d) 0.25×1 domain

Fig. 5. Optimized support structures in rectangular domain of different aspect ratios. All optimized designs exhibit tree-like, self-supporting structures. The optimized supports in long domains can be regarded as a succession of arcs like the one in Fig. 1c.



(a) Supports without overhang constraint (b) Supports with overhang constraint

Fig. 6. Comparison of the computed supports in a 0.5 × 1 rectangular domain with and without an overhang constraint [2].

These overhangs can however be easily removed by an overhang constraint [2]. For instance, in Fig. 6 we report the optimal supports computed in a 0.5 × 1 rectangular domain with and without an overhang constraint. The maximum volume fraction is again $V_f = 30\%$ and the discretization is performed by 51, 200 T3 elements. In the overhang constrained case, we used the PUP overhang constraint [2] with overhang angle $\overline{\alpha} = 75°$, allowed projected overhang perimeter $\overline{P}_{\overline{\alpha}} = 0.55$, intermediate density constraint threshold $\bar{\varepsilon} = 0.15$, maximum β of Heaviside in continuation methods $\beta_{\max} = 16$ and filter radius $r_f = 10$. For consistency, all parameters not involving the PUP constraint have been kept equal in the unconstrained case.

Interestingly, the cost is smaller for the structure computed using the overhang angle constraint. Indeed, the unconstrained design presents a cost of 8.12, while the cost of the overhang constrained design is 7.40. On one hand, why this happens is not immediately clear, since adding a constraint reduces the space of admissible solutions and the cost should thus be higher. On the other, the design in Fig. 6b more closely resembles the arcs noted for the other aspect ratios and is perfectly consistent with the remarks made throughout this section. Thus, forbidding the non self-supporting design of Fig. 6a can actually lead to a smaller compliance.

4.4. Zero-displacement boundary conditions on the sides of the design domain

We generally obtain self-supporting structures also when supports are built on parts other than the base plate. As mentioned earlier, this corresponds to adding zero-displacement boundary conditions on some vertical boundary as well. Self-support can be justified as done above for supports built only on the printing plate. Indeed, branches tilted more than 45° would arguably worsen the load transfer.

Figs. 7 and 8 show that this in indeed what happens. Fig. 7 represents the optimized designs for different boundary conditions in a 1 × 1 domain and Fig. 8 represents the optimized designs in a 3 × 1 domain when supports can be built on both vertical boundaries. In both cases, $V_f = 20\%$ and the discretization is performed by T3 elements.

In Fig. 7 we observe how the optimized design changes as zero-displacement boundary conditions are placed only at the bottom of the domain or on one or on both sides of the domain as well. All these situations are represented in Fig. 7a–c respectively.

All solutions appear self-supporting: zones with angles flatter than 45° are indeed always short. Moreover, it is interesting to notice how the final value of the compliance gets smaller as the zero-displacement boundary conditions involve more boundaries. This was to be expected by the very fact that the design changes: the final design of Fig. 7a is indeed admissible also when we add vertical boundary conditions. Therefore, if it were optimal also when zero-displacement boundary conditions are placed on the sides of the domain as well, we would get the same result also in Fig. 7b and c.

We also observe that vertical boundary conditions act as additional trunks placed along the sides of the domain. This is ultimately the reason why the cost is reduced: indeed, vertical boundary conditions behave as additional trunks which do not use any material and are characterized by zero displacement. So, branches can connect directly to the sides of the domain. In the extreme case of Fig. 7c, we no longer have any trunk: since the domain is short, it is indeed enough to have a single arc made of branches directly connected with the sides of the domain.

In Fig. 8 the final design changes in a similar way. The fact that the domain is longer, however, makes so that trunks appear in the middle of the domain and the usual tree-like structure is more apparent. Only the branches near the sides are directly linked to the vertical boundaries, tilted of about 45°. This is consistent with the previous analysis and compatible with self-support. Also in this case, the cost is reduced with respect to the case where supports can be built only on the printing plate.

4.5. Length-scale and other considerations

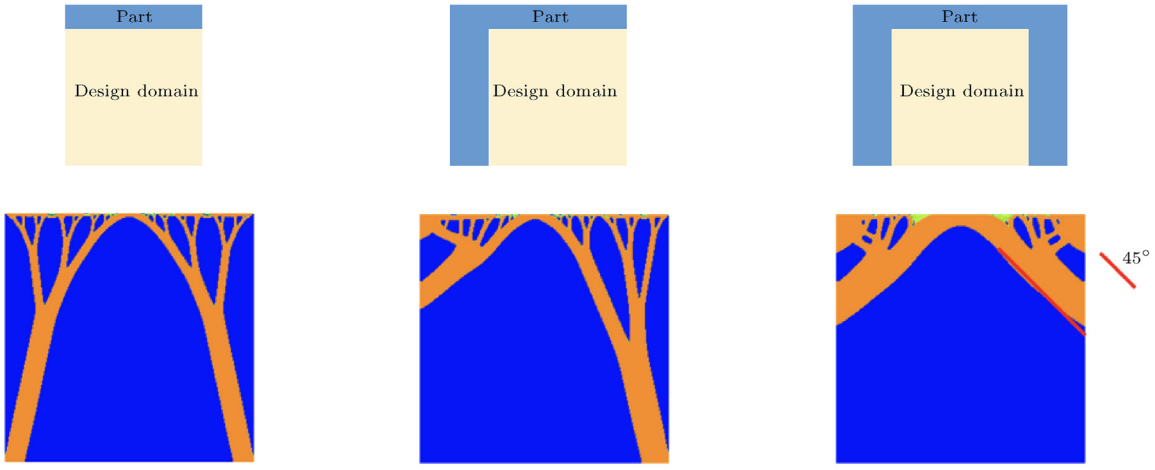
A further difference between using topology optimization to produce support structures and other strategies in the literature is that here we do not perform a sampling of the surface to be supported. Indeed, we do not have a set of points where supports will be built, but branches can touch the overhanging region everywhere. In particular, branches get more numerous and thinner as we approach the top boundary of the domain, where the load is applied. Indeed, ideally, we need one branch for every point where the load is applied. This originates several branches, each needing to transfer a small load (hence their thinness).

In order to avoid having features too fine to be printed, we can use length-scale control. In this way, minimum length-scale can be chosen so to match the resolution of the printer. For instance, for a rectangular domain we obtain the results in Fig. 9.

When we do not use any length-scale control, as in Fig. 9a, we have several thin branches. If these features are too small to be printed successfully, we need to impose a minimum length-scale. Fig. 9b introduces length-scale control, but the filter radius is small and fine features persist. Starting from $r_f = 3$ (Fig. 9c), we have designs more compatible with the resolution of a 3D printer. Considering yet larger values of r_f, features get even larger. For instance, in Fig. 9e we set $r_f = 7$ and we only have a few, large branches which should be easily printable with most 3D printers commonly used.

5. Results and analysis in more complex domains

In this section, we present and analyze the results of experiments in non-rectangular domains. First, we analyze the results obtained in 2D



(a) Supports only from printing plate. Cost: 24.68

(b) Supports from printing plate and left boundary. Cost: 14.15

(c) Supports from printing plate and vertical boundaries. Cost: 6.26

Fig. 7. Optimized supports in a 1×1 rectangular domain with 0-displacement boundary conditions on different parts of the boundary. In all cases, the optimized designs are self-supporting.

domains. After applying a linear extrusion to the optimized support structures thus obtained, we then actually print them, in order to show that the computed structures are indeed self-supporting. We then pass to 3D optimization, in particular considering supports for the MBB beam and for domes of various heights. Again, we print all the optimized structures, so to demonstrate self-support. We also compare our results with the support structures computed by existing software and consider ways to ease the removal of the supports after printing.

5.1. 2D supports in various domains

Let us consider the curved domain in Fig. 2b. Choosing, for instance, $\overline{V} = 0.15$ and $r = 0.009$, we get the design in Fig. 10. The load is supposed uniform and the discretization has been performed by 112,510 T3 elements.

Fig. 10 shows that curved boundaries do not dramatically affect the result of the optimization. Indeed, we obtain arcs and tree-like structures as in the cases in rectangular domains. Moreover, the design is self-supporting, since there are no long, horizontal regions which may collapse during the printing process.

Let us then compute supports for non self-supporting holes occurring in some popular structures, such as the MBB beam (see for instance Fig. 2c) and the cantilever beam. The results are reported in Fig. 11. Here, the design in light gray represents the part that we want to print. The structure in red is the computed optimized support. The parts in blue represent zones which are included in the domain but where no material is placed by the optimization.

The supports for MBB beam have been computed in a discretization made of 210, 234 T3 elements, choosing a maximum volume fraction $\overline{V} = 0.2$ (relative to only the two triangles needing supports) and $r = 0.008$. Regarding the cantilever beam, the domain has been discretized by 167, 591 T3 elements with $\overline{V} = 0.2$ and $r = 0.01$.

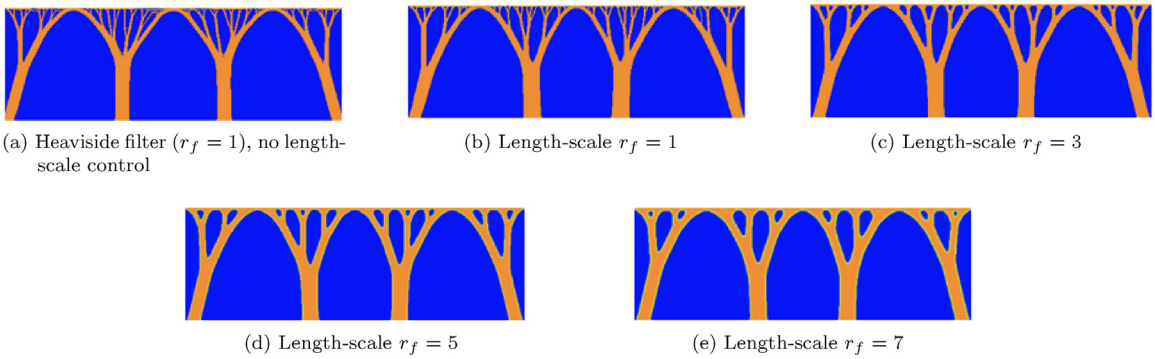
The optimized supports are tree-like and self-supporting, as we better demonstrate in the next sub-section. Moreover, as in previous cases, the structure is made by a series of arcs. Finally, we also notice that the pillars between two arcs are nearer and nearer as the height of the part to support is smaller. In this regard, see, for instance, the sides of the top triangle of the cantilever beam. This happens because the "cost" (in terms of used material) for adding a new vertical pillar (which optimally transfers the load from the top to the bottom of the domain) gets smaller as the height of the pillar itself is reduced. Thus, supports in short zones are made of several short pillars, while in high zones we have less, taller pillars to which multiple branches are connected. Also in these last areas, however, we do not have so few pillars as to have branches tilted more than 45° with respect to the build direction. Consistently with the analysis performed in Section 4, this can be again explained by the fact that such a structure would be arguably characterized by worse loading transfer.

The presence of arcs in all these examples is also one of the features that differentiate our results from tree-like supports present in the literature. Indeed, supports in [13] are generated starting from a sampling of the surface of the overhanging part and each branch touches exactly one sample point. At the tip of arcs, on the other hand, the two branches making up the arc touch the same points, which, moreover, do not come from a sampling but from the result of an optimization, provided with mechanistic meaning.

5.2. 2D extruded optimized designs

In order to demonstrate that the support structures computed by a

Fig. 8. Optimized self-supporting structure in a 3×1 rectangular domain with 0-displacement boundary conditions on the bottom and on the sides of the domain.



(a) Heaviside filter ($r_f = 1$), no length-scale control

(b) Length-scale $r_f = 1$

(c) Length-scale $r_f = 3$

(d) Length-scale $r_f = 5$

(e) Length-scale $r_f = 7$

Fig. 9. Comparison between no length-scale control and different choices of minimum length-scale. Length-scale control can be used to ensure that the optimized support does not present features so small to be incompatible with the resolution of the 3D printer.

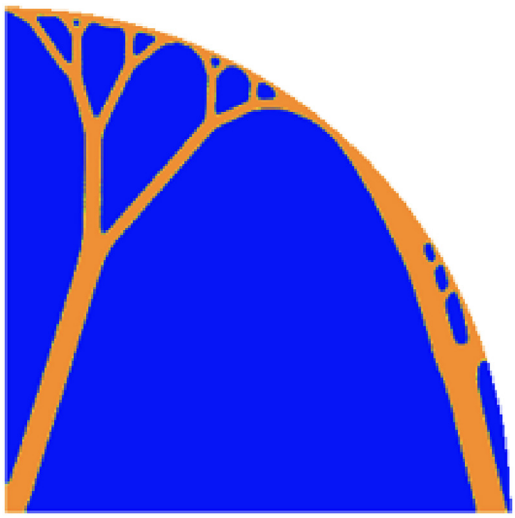


Fig. 10. Optimized support for curved domain.

topology optimization procedure are indeed self-supporting, we here show that it is possible to actually print them without any additional support.

First, let us consider some supports computed in the previous subsection, apply a linear extrusion to the computed results and try to print them. For instance, the designs in Fig. 12 have been obtained by applying a linear extrusion to the supports represented in Fig. 9d and those for the MBB beam in Fig. 11a.

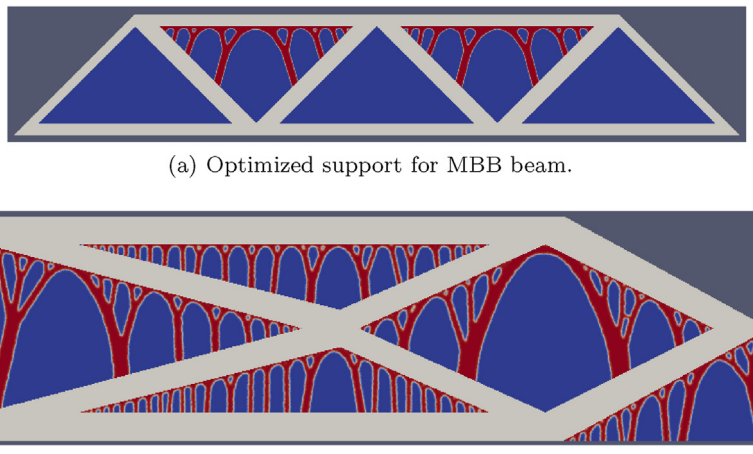
We can then print the extruded designs. For instance, let us first print a simple example in a rectangular domain. Applying a linear extrusion to the design in Fig. 9d and 3D-printing the resulting structure using Autodesk Ember, we obtain the structure in Fig. 13a.

The structure has been printed along a build direction going from the bottom of the structure represented in Fig. 12a to the top. This reproduces the direction along which the structure would be printed if it had to support an overhanging part. The fact that we were able to print this structure without further supports demonstrates that self-support holds. Indeed, the overhanging parts of the structure are small enough to be self-supporting.

We can then pass to a more complex example and consider, for instance, the design in Fig. 12b. In this case, we print a detail of both the computed support structure and the MBB beam. In this way, we evaluate not only self support, but also the ability of the optimized structure to indeed behave as a support. For clearness, the part and the support are printed in two different colors. The results, printed in an Ultimaker 3, are reported in Fig. 13b.

The printed part shows that the computed structures successfully behave as supports. Indeed, they allow us to print a design that would otherwise be impossible to print along the chosen orientation. Simultaneously, they do not require further supports to be printed.

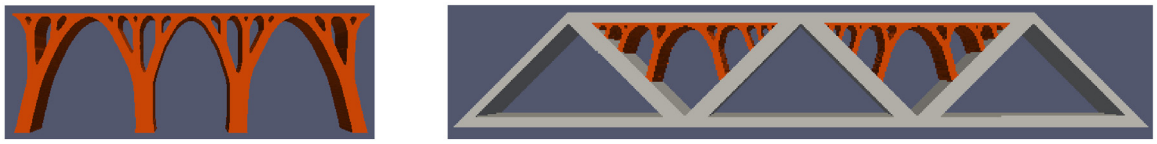
We highlight that the printed supports did not go through any post-processing (other than linear extrusion) before printing. They are directly the result of the topology optimization performed with loading and boundary conditions described in Section 2 with numerical data as in 3.1. Therefore, this analysis confirms that in general the optimization leads to self-supporting designs. We thus have the remarkable feature that the results of the topology optimization can be printed directly.



(a) Optimized support for MBB beam.

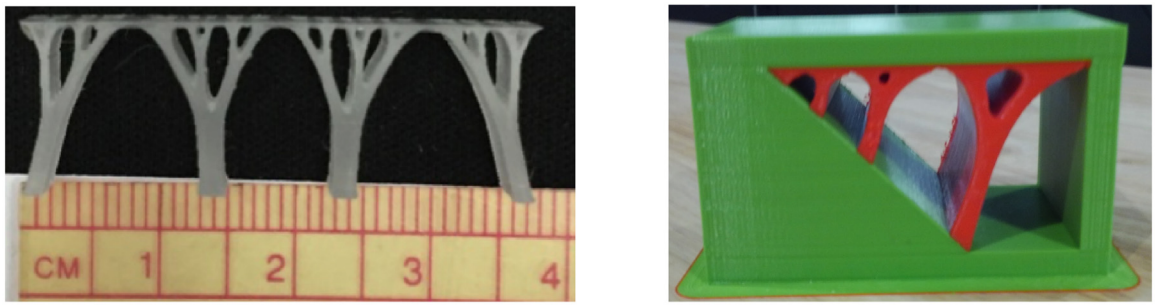
(b) Optimized support for cantilever beam

Fig. 11. Optimized supports for holes and overhangs of popular structures.



(a) Extruded design of Figure 9d (b) Extruded MBB beam (gray) and supports (red)

Fig. 12. Linear extrusions of the optimized supports in Figs. 9d and 11a .



(a) 3D printed extruded solution for uniform distributed load in a rectangular 3×1 domain. Printed in an Autodesk Ember.

(b) 3D printed detail of a support for part of a hole of the MBB beam. Printed in an Ultimaker 3.

Fig. 13. Linear extrusions of the optimized supports in Figs. 9d and 11a .

5.3. Large scale 3D optimization

It is now interesting to see whether this holds also when a 3D optimization is run. 3D problems have been run on a cluster of computers, as detailed in Section 3.1. The results obtained here and in the following Section have been computed by recurring to large-scale computing so to demonstrate that we can effectively compute high-resolution optimized supports. In practice, however, lower resolutions may be acceptable and can lead to a significantly smaller computational effort. Analogously, computational effort may be reduced also by stopping the optimization procedure before its complete convergence. Depending on the problem, on the desired resolution and on the available computational power it is, then, possible to find different trade-offs between the support optimization effort and the material usage reduction.

Considering, for instance, the MBB beam, the results of the optimization are reported in Fig. 14 and the subsequent 3D-printed structure are reported in Fig. 15. The discretization has been performed using 33 million hexagonal elements, corresponding to more than 100 million variables. Volume fraction is $V_f = 10\%$.

The first thing we notice is that we obtain tree-like structures also when we consider a 3-dimensional domain. In a way, the results are, thus, similar to those obtained by performing a linear extrusion of 2D results. Both, indeed, present structures characterized by arcs and branches. However, in extruded 2D examples, we can have arc and pillars only along one direction (which is, along the length), while the linear extrusion simply replicates the 2D design along the width. In 3D results, on the other hand, we have several pillars, all separated from one another, on both length and width. The fineness of the features is evident not only in the computed results of Fig. 14, but also in the 3D printed sample in Figure 15a, which has been printed in a Form 2 V3 (FLTOTL03) 3D printer with SLA process (using tough resin with layer thickness at 0.1 mm. Slicing was performed in PreForm software).

We also see that the computed supports are, again, mostly vertical. Critical overhangs are sufficiently short and can be printed with no additional supports. This is demonstrated by Fig. 15b, which also shows that the computed structure can indeed act as a support. Indeed, both the green and the gray parts have been printed together on an Ultimaker 3 3D printed (using PLA plastic, AA 0.4 print core, with 0.1 mm layer height and 20% infill. Slicing conducted in Cura software). The fact that we were able to print the green part (which is evidently not self-supporting) without structural failures and without supports other than the results of our optimization testifies the applicability of the results of our approach for computing real supports to real 3-dimensional structures. In the following, we also show how the supports computed by a topology optimization process compare to those computed by existing software.

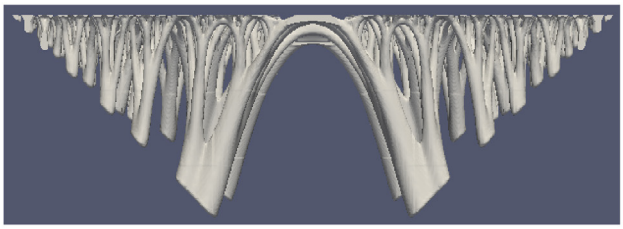
We remark that all the represented structures have been obtained with no post-processing. The results in Fig. 14 are directly the result of the optimization (extracted by a.vtu file using *Paraview*, as described in Section 3).

We then consider the supports for a curved overhang. For example, we can assume we need to print a dome. We consider domes of different heights. Fig. 16 represents the results of the optimization for a “tall” dome (whose elevation from the baseplate is equal to the diameter of the dome itself), for a “short” one (whose elevation is now equal to 20% of the diameter) and for a dome which touches the printing plate.

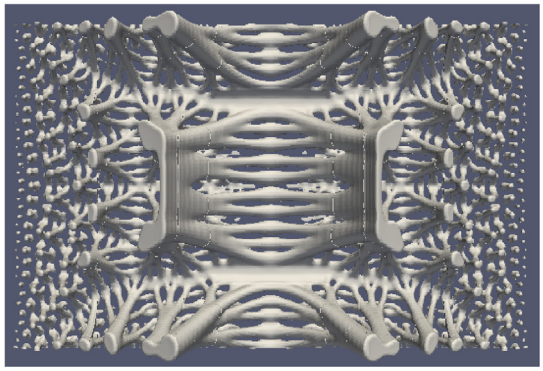
The supports for the tall dome have been obtained by discretizing the domain by 132 million hexagonal elements (corresponding to more than 400 million variables), volume fraction is $V_f = 5\%$ and the filter radius is $r = 0.02$. The other domes have instead been reported for comparison and they have been obtained with looser discretizations (amounting to just a few million hexagonal elements).

It is interesting to notice that more pillars appear as the dome gets shorter. This is reasonable and consistent to what previously noticed. Indeed, the best way to transfer the load from the part to support (at the top of the domain) to the base plate is by a straight pillar. A single, large pillar is however incompatible with any reasonable volume constraint. Therefore, we get also branches, which connect the points of the overhanging part to a pillar or to another branch. Clearly, when the structure is shorter, adding a new pillar in less “expensive” in terms of volume fraction, while it improves the load transfer to the printing plate. Therefore, shorter structures tend to have more pillars and, in general, more vertical parts. Nonetheless, as explained earlier, also tall structures require pillars ensuring the load transfer, making so that also in these cases the computed supports are mostly vertical and, thus, self-supporting.

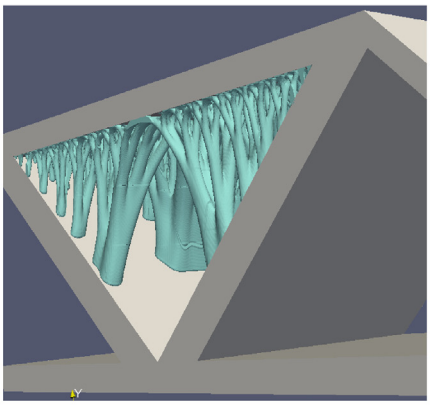
Let us now 3D print the tall dome, whose optimized supports are reported in Fig. 16a. The structure in Fig. 17 demonstrates that we are indeed able to print the curve surface (in red) using the computed supports (black), which are, once again, tree-like and self-supporting. The reported design has been again printed by a Ultimaker 3 3D printer using PLA plastic (AA 0.4 print core) with 0.1 mm layer height setting



(a) Computed support (front)



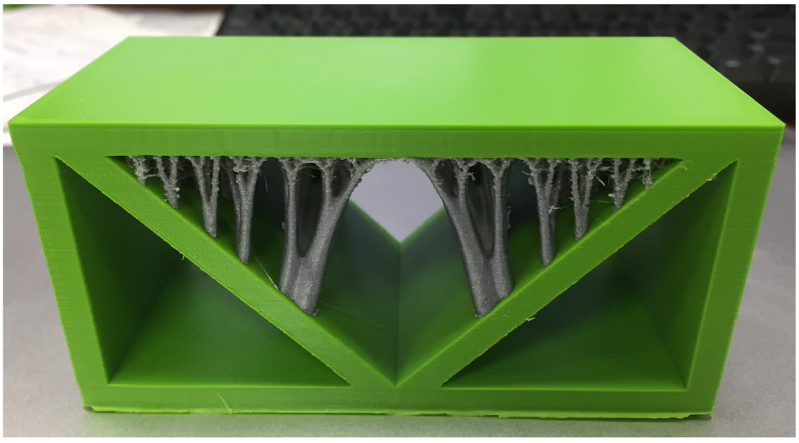
(b) Computed support (bottom view)



(c) Computed support (turquoise colored) inside the overhang to support of the MBB beam (in gray)

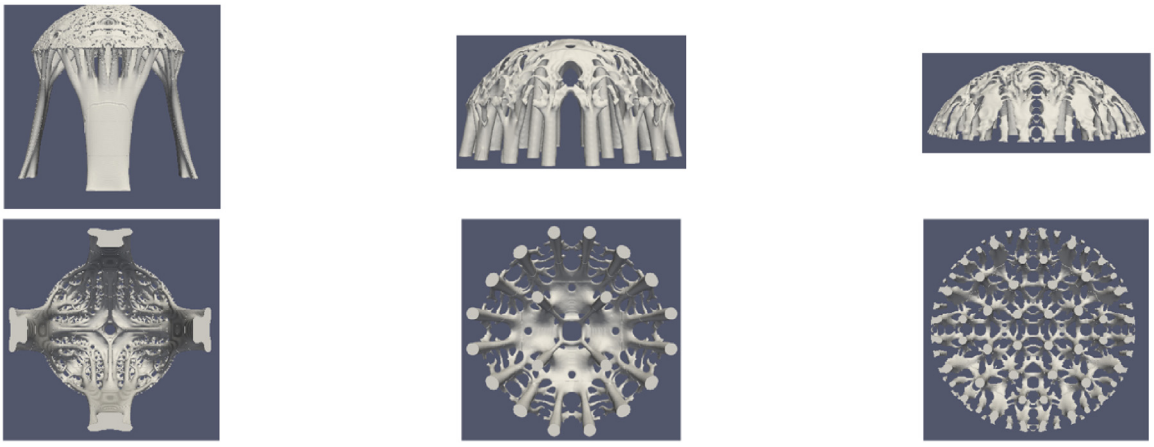
Fig. 14. Optimized support structure for a 3D MBB beam.

(a) 3D-printed support



(b) MBB beam (in green) printed using the computed structures as supports (in gray)

Fig. 15. 3D-printed supports of the optimized 3D MBB beam.



(a) Tall dome (b) Short dome (c) Dome on printing plate

Fig. 16. Optimized support for domes of different heights.

Fig. 17. 3D printed optimized supports for a tall dome (printed in an Ultimaker 3).

and 20% infill. The support materials and dome were printed in separate color filament to aid with visualization of the final print.

5.4. Comparison with supports generated by existing software

After having established that the computed structures can successfully act as supports in a variety of situations, it is now interesting to see how they compare to supports created by existing software. This analysis was conducted through the Cura slicer, which simulates the print itself. Cura can then be also used to estimate print time and filament length and to approximate the mass of filament used by each nozzle. In this regard, Cura simulated a print in a Ultimaker 3 3D printer with dual extruders both printing in ABS plastic using AA 0.4 standard printing cores and with recommended settings at a fine print profile specifying a layer thickness of 0.1 mm.

First, let us compare graphically the renders of the support structures computed by our approach with those recommended by Cura software. For instance, Fig. 18 refers to the supports for the MBB beam. Analogous results can be obtained by considering other situations, such as the domes we previously analyzed.

More precisely, in Table 1 we compare numerically the amount of material used by the two approaches. In the table, "TO" denotes our strategy of computing the support structures by topology optimization.

This comparison shows that the structures computed by a topology optimization procedure greatly reduce the amount of material used for the support. This is particularly remarkable since in the optimization we did not choose the strictest volume constraint possible: indeed, we simply chose a priori a reasonable value for the maximum volume fraction. It is therefore possible to further reduce the amount of used material.

On the other hand, printing supports generated by topology optimization may require more time. This is caused by the more complex shapes involved and by the fact that the printing of several columns and branches makes so that the flux of material is started and stopped repeatedly, and the filament is hence less continuous. Times are nonetheless comparable and support structures generated by topology optimization can be printed faster in some cases, as in that of a short dome.

5.5. Easy removal of supports

Finally, it is interesting to give a few remarks about the possibility of coupling our approach with techniques that make the removal of the supports easier after printing is complete. Indeed, the removal of the supports is crucial in order to reduce the cost of post-processing and to enhance surface quality. Moreover, it makes the comparison in the previous subsection more consistent, since supports generated by Cura software include easy removal.

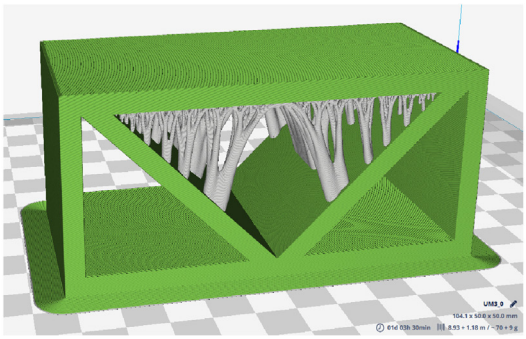
Easy removal is not included in the optimization procedure. Indeed, the optimization produces a structure which, ideally, touches every point where the load is applied. Therefore, the computed support would be hard to remove, since it would be connected to the part to print in large zones.

This can however be solved easily by considering some existing techniques for achieving an easy removal of supports. Indeed, we can compute the support structure with a topology optimization procedure exactly as done before. Then, we simply need to modify the structure at the interfaces between support and part to print. In this regard, we can simply replace the continuous connections between support and part to print by "comb-like" structures, where the part to print is connected to the support by several "teeth" which are easy to detach.

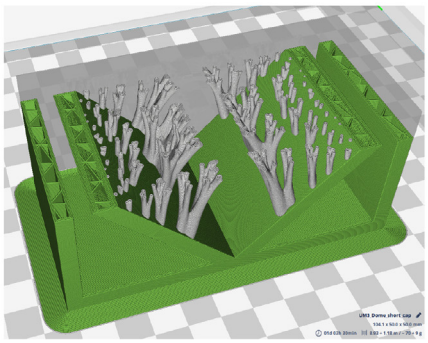
Fig. 19 represents a render of the support computed by our approach for the MBB beam "corrected" at the interfaces so that it is easily removable after the printing is complete. Since these changes involve only the interfaces between part and support, the amount of used material varies only slightly. Thus, the comparison in Table 1 holds with no relevant modifications.

6. Application to a real structure

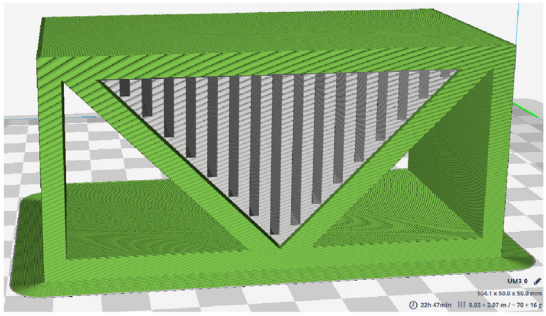
After analyzing several domains characterized by loading and



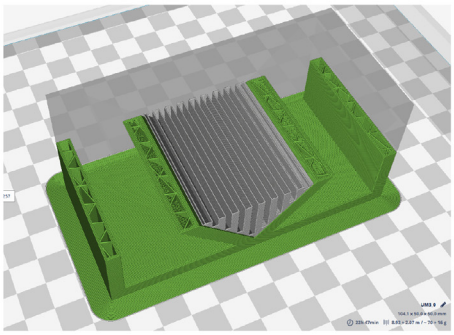
(a) MBB beam with supports generated by topology optimization



(b) Section of supports generated by topology optimization



(c) MBB beam with supports generated by Cura software



(d) Section of supports generated by Cura software

Fig. 18. Comparison of the supports for the MBB beam: supports computed by topology optimization vs. supports generated by Cura software.

Table 1
Numerical comparison of support structures computed by topology optimization and by Cura software.

	Print time	Filament length	Mass supports	Mass reduction (%) with TO
MBB TO	1 d 3 h 50 min	1.18 m	9 g	43.8
MBB Cura	0 d 22 h 47 min	2.07 m	16 g	
Tall dome TO	0 d 16 h 59 min	3.97 m	23 g	37.8
Tall dome Cura	0 d 14 h 25 min	5.71 m	37 g	
Short dome TO	0 d 17 h 47 min	2.75 m	21 g	36.4
Short dome Cura	0 d 18 h 29 min	4.27 m	33 g	

boundary conditions defined on different kinds of boundaries, we aim at computing the optimized support for a more complicated, “real” structure. In this regard, we chose to print a representation of the mascot of the University of Wisconsin-Madison, which is called Bucky Badger. The model.stl file is represented in Fig. 3b and approximated be the forms in Fig. 3c.

We considered two different approaches: conducting the optimization in the entire domain at once or dividing it in some sub-domains. This last idea is particularly interesting for two reasons. First, running the optimization in more separate processes allows us to use finer discretizations more easily. Indeed, if we can split the domain and run the program separately in the various zones, we have great advantages in the scalability of the problem. The second reason is that in a real structure it can definitely happen that we have self-supporting zones alternated with other zones which instead need a support. Considering various subdomains instead of running the optimization all at once over the entire domain can thus be easier and it can also lead to supports which are easier to 3D print. On the other hand, the division of the domain can be not banal and volume fraction, filter radius, etc. of the optimization in the various subdomains must be chosen consistently.

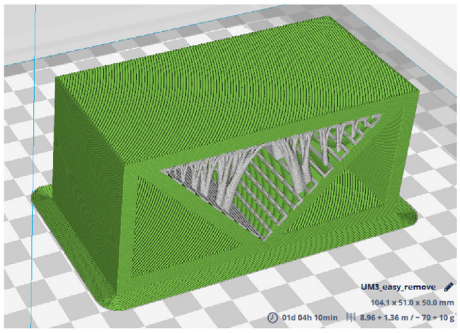
Fig. 20 shows the results obtained conducting the optimization in the entire domain using 82, 944, 000 hexagonal elements (250, 737, 123 DOFs) and choosing a volume fraction $V_f = 0.05$ and filter radius $r = 0.04$. The computation has been performed using 300 cores (distributed in 15 nodes, each containing 20 cores and provided with 128 GB of RAM). Total computational time was 11 h and 43 min.

We again notice that optimized supports are characterized by tree-like structures. Indeed, in Fig. 20a and c we can easily observe that the design is again made of branches connected to main pillars. Moreover, although the overhang situation appears more critical than in previous examples, the optimized structure appears self-supporting. We will better demonstrate self-support in the following with an actual 3D print of an analogous result.

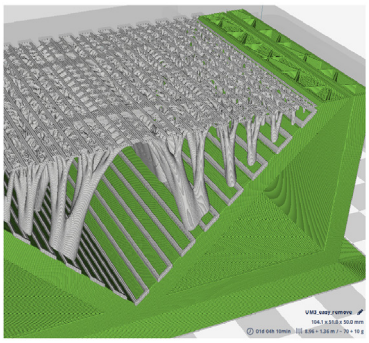
We also notice that we can divide the optimized structure in regions which are one independent from the others. This is particularly evident in Fig. 20b. We see, for instance, that the structure supporting the head of Bucky is not connected to any other support by structurally important elements. We can easily identify four similar “independent” regions: head, body and arms, left leg and, finally, right leg.

Exploiting that the supports of these regions are connected to other parts only by thin branches with no evident structural purpose, we can run the optimization separately in these four areas. Running the optimization in subdomains, we thus achieve the advantages described earlier. Moreover, this subdivision domain is consistent with the results obtained running the optimization all at once and arguably leads to similar results.

The optimized structure obtained from this subdomain-based

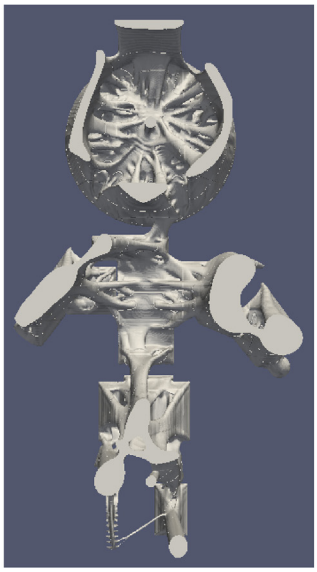


(a) Supports modified for easy removal

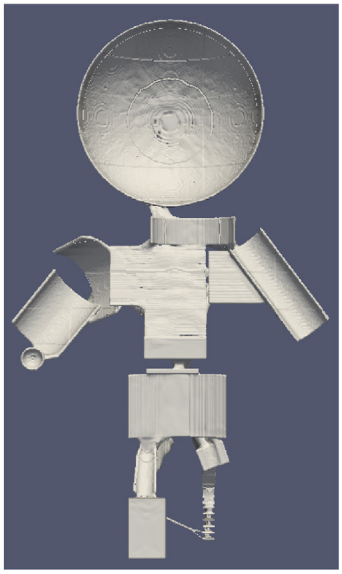


(b) Detail of the interface between support and part to print

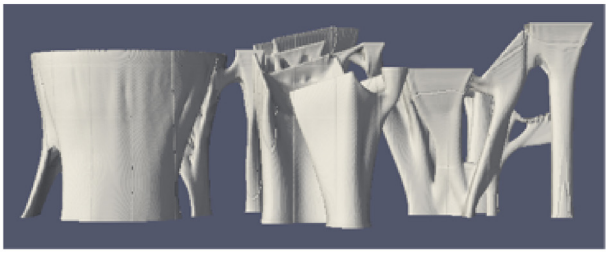
Fig. 19. Render of the optimized support for the MBB beam modified for easy support removal.



(a) Bottom view



(b) Top view



(c) Side view and build direction

Fig. 20. Optimized design with the entire volume as design domain.

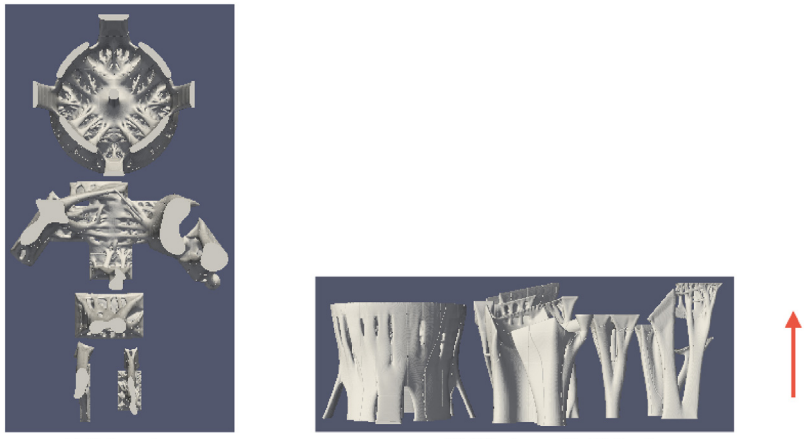
optimization is reported in Fig. 21. Table 2 reports the data regarding the discretization, the choice of volume fraction and filter radius and the computation for each subdomain.

The first thing we notice is that the design in Fig. 20 and that in Fig. 21 are really similar. Not only they are both characterized by a tree-like structure, but also the disposition of pillars and branches is analogous. This confirms that we could indeed run the optimization separately in the chosen subdomains.

A difference is instead that the resolution of the designs in Fig. 21 is evidently higher. The total number of elements used in the discretization is in fact larger, as we can see in Table 2. Indeed, the total number of DOFs of the discretization exceeds 1.2 billion, almost five times as many as the DOFs of the discretization used for computing the structure in Fig. 20.

Lastly, we observe the flexibility of subdomains. Indeed, we can locally modify volume fraction (for instance, to make some zones less bulky or, on the contrary, to use more material only where needed) and filter radius. For instance, Table 2 shows that in the region containing body and arms we used a larger filter radius, so do avoid thin branches which can easily form in this zone.

Lastly, the optimized support has been used to actually 3D print the model of Bucky. Fig. 22a represents a render of the optimized support applied to the model. Fig. 22a, c and d then show the 3D printed part (in red) supported by the optimized structure (in black) from various

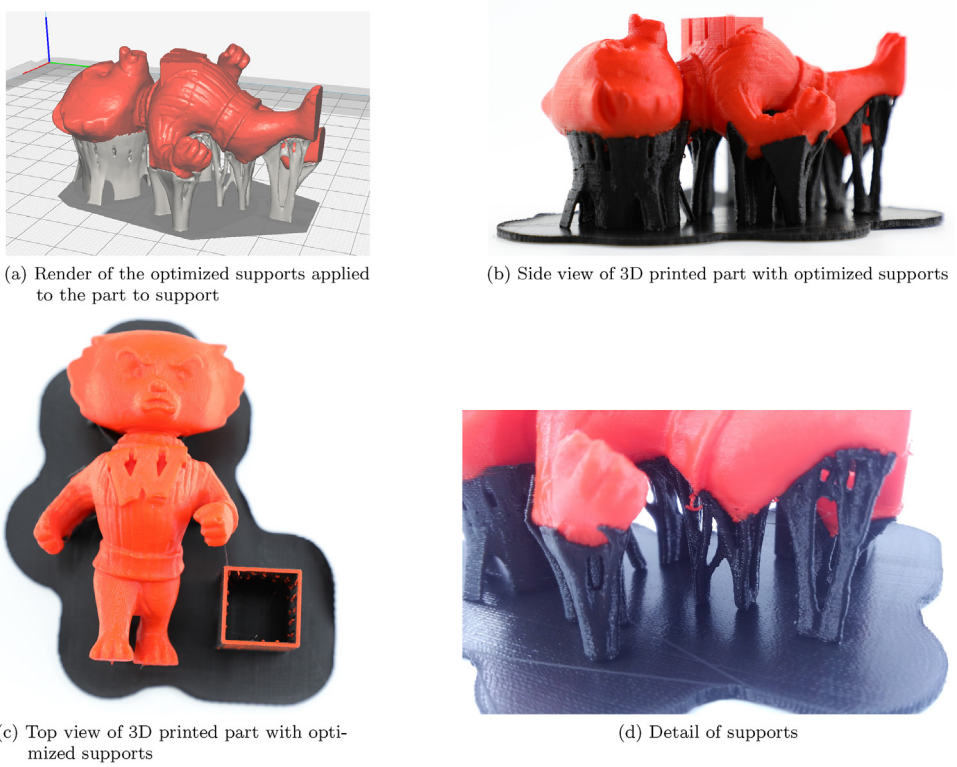


(a) Bottom view (b) Side view and build direction

Fig. 21. Optimized support with 4 different subdomains: one for the head, one for the body and the arms and two for the two legs.

Table 2
Numerical data of discretizations, optimization parameters and computational times in the subdomains.

Sub-domain	Elements	DOFs	Vol. frac.	Filter rad.	Number of cores	Computational time
Head	132,217,728	405,017,091	0.05	0.02	300	12 h 40 m
Body and arms	130,842,624	394,948,755	0.04	0.05	300	16 h 52 m
Left leg	65,421,312	198,033,795	0.05	0.02	160	14 h 28 m
Right leg	67,108,864	202,903,299	0.05	0.02	160	12 h 58 m
Total	395,590,528	1,200,902,940	–	–	–	–



(a) Render of the optimized supports applied to the part to support

(b) Side view of 3D printed part with optimized supports

(c) Top view of 3D printed part with optimized supports

(d) Detail of supports

Fig. 22. 3D printed optimized support structures for Bucky and associated part to support.

viewpoints. Printing was performed in PLA plastic on Ultimaker 3 printer using an AA 0.4 core. The fact that we were able to print the computed design without additional support structures demonstrates that self-support of the optimized structure indeed hold. Furthermore, Fig. 22 also demonstrates that the computed structure can act as a support. Indeed, it has been able to successfully support the model of Bucky that we printed along with the support itself.

7. Conclusions

We have presented a framework for generating self-supporting support structures for additive manufacturing by a topology

optimization procedure. We have formulated the problem as a compliance minimization problem that exhibits analogies with compliance minimization of bridges and of roof supports, with, however, a difference in zero-displacement boundary conditions. This difference fosters a comparison with transmissible loads which, together with other mechanical considerations, helps explaining why the optimized structures are usually self-supporting. Manufacturability constraints and length-scale control can moreover be used as needed.

Numerical experiments demonstrate that the procedure is general and applicable to support parts of various shapes, including those describing a complex model of the mascot of the University of Wisconsin-Madison. Moreover, a comparison with supports generated by existing software shows that the computed structures indeed employ less material than other approaches currently used. Finally, we actually printed some designs, showing that the computed structures are indeed printable and can self-support.

Acknowledgements

This work is supported in part by a UW-Madison Draper Technology Innovation Fund.

The 3D optimization took place with the compute resources and assistance of the UW-Madison Center For High Throughput Computing (CHTC). The 3D printing took place in the UW Makerspace.

References

[1] D. Brackett, I. Ashcroft, R. Hague, Topology optimization for additive manufacturing, Proceedings of the Solid Freeform Fabrication Symposium, Austin, TX, 2011, pp. 348–362.
[2] X. Qian, Undercut and overhang angle control in topology optimization: a density gradient based integral approach, Int. J. Numer. Methods Eng. 111 (3) (2017) 247–272.
[3] A.M. Mirzendehdel, K. Suresh, Support structure constrained topology optimization for additive manufacturing, Comput.-Aided Des. 81 (2016) 1–13.
[4] M. Langelaar, An additive manufacturing filter for topology optimization of print-ready designs, Struct. Multidiscip. Optim. (2016) 1–13.
[5] M. Langelaar, Topology optimization of 3d self-supporting structures for additive manufacturing, Addit. Manuf. 12 (2016) 60–70.
[6] G. Allaire, C. Dapogny, A. Faure, G. Michailidis, Shape optimization of a layer by layer mechanical constraint for additive manufacturing, C. R. Math. 355 (6) (2017) 699–717.
[7] A.T. Gaynor, N.A. Meisel, C.B. Williams, J.K. Guest, Topology optimization for additive manufacturing: considering maximum overhang constraint. AIAA Aviation, American Institute of Aeronautics and Astronautics, 2014.
[8] A.T. Gaynor, J.K. Guest, Topology optimization considering overhang constraints: eliminating sacrificial support material in additive manufacturing through design, Struct. Multidiscip. Optim. 54 (5) (2016) 1157–1172.
[9] Xu Guo, Jianhua Zhou, Weisheng Zhang, Zongliang Du, Chang Liu, Ying Liu, Self-supporting structure design in additive manufacturing through explicit topology optimization, Comput. Methods Appl. Mech. Eng. 323 (2017) 27–63.
[10] Yoram Mass, Oded Amir, Topology optimization for additive manufacturing: accounting for overhang limitations using a virtual skeleton, Addit. Manuf. 18 (2017) 58–73.
[11] P. Alexander, S. Allen, D. Dutta, Part orientation and build cost determination in layered manufacturing, Comput.-Aided Des. 30 (5) (1998) 343–356.
[12] G. Strano, L. Hao, R.M. Everson, K.E. Evans, A new approach to the design and optimisation of support structures in additive manufacturing, Int. J. Adv. Manuf. Technol. 66 (2013) 1247–1254.
[13] J. Vanek, J.A.G. Galicia, B. Benes, Clever support: efficient support structure generation for digital fabrication, Comput. Graphics Forum 33 (5) (2014) 117–125.
[14] Yong Chen, Kang Li, Xiaoping Qian, Direct geometry processing for telefabrication, J. Comput. Inf. Sci. Eng. 13 (4) (2013).
[15] Matthijs Langelaar, Topology optimization for additive manufacturing with controllable support structure costs. in: G. Stefanou, V. Plevris, M. Papadrakakis, V. Papadopoulos (Eds.), ECCOMAS Congress 2016 VII European Congress on Computational Methods in Applied Sciences and Engineering, 2016.
[16] Matthijs Langelaar, Combined optimization of part topology, support structure layout and build orientation for additive manufacturing, Structural and Multidisciplinary Optimization, (2018).
[17] Yu-Hsin Kuo, Chih-Chun Cheng, Yang-Shan Lin, Cheng-Hung San, Support structure design in additive manufacturing based on topology optimization, Struct. Multidiscip. Optim. 57 (1) (2018) 183–195.
[18] Iasef Md Rian, Mario Sassone, Tree-inspired dendriforms and fractal-like branching structures in architecture: a brief historical overview, Front. Archit. Res. 3 (2014) 298–323.
[19] W. Nerdinger, Frei Otto Complete Works: Lightweight Construction Natural Design, Birkhäuser, Basel; Boston; Berlin, 2005.
[20] M.P. Bendsoe, Optimal shape design as a material distribution problem, Struct. Optim. 1 (1989) 193–202.
[21] Martin Philip Bendsoe, Ole Sigmund, Topology Optimization: Theory, Methods and Applications, 2nd ed., Springer, 2004.
[22] Peter W. Christensen, Anders Klarbring, An Introduction to Structural Optimization, volume 153 of Solid Mechanics and its Applications, Springer, 2009.
[23] M. Sasaki, Morphogenesis of Flux Structure, Architectural Association Publications London, 2007.
[24] T. Zegard, G.H. Paulino, Bridging Topology Optimization and Additive Manufacturing, Structural and Multidisciplinary Optimization, 2016.
[25] Asger Nyman Christiansen, J. Andreas Bærentzen, Morten Nobel-Jørgensen, Niels Aage, Ole Sigmund, Combined shape and topology optimization of 3d structures, Comput. Graphics 46 (2015) 25–35.
[26] J. Kingman, K.D. Tsavdaridis, V.V. Toropov, Applications of topology optimization in structural engineering, Civil Engineering for Sustainability and Resilience International Conference (CESARE), 24–27 April 2014, Amman, Jordan, 2014.
[27] Atsushi Kawamoto, Tadayoshi Matsumori, Shintaro Yamasaki, Tsuyoshi Nomura, Tsuguo Kondoh, Shinji Nishiwaki, Heaviside projection based topology optimization by a PDE-filtered scalar function, Struct. Multidiscip. Optim. 44 (1) (2011) 19–24.
[28] B.S. Lazarov, O. Sigmund, Filters in topology optimization based on Helmholtz-type differential equations, Int. J. Numer. Methods Eng. 86 (6) (2011) 765–781.
[29] Shengli Xu, Yuanwu Cai, Gengdong Cheng, Volume preserving nonlinear density filter based on Heaviside functions, Struct. Multidiscip. Optim. 41 (2010) 495–505.
[30] M.S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M.E. Rognes, G.N. Wells, The FEniCS project version 1.5, Arch. Numer. Softw. 3 (100) (2015) 9–23.
[31] A. Logg, K.A. Mardal, G.N. Wells, et al., Automated Solution of Differential Equations by the Finite Element Method, volume 84 of Lecture Notes in Computational Science and Engineering, Springer, 2012.
[32] Krister Svanberg, The method of moving asymptotes: a new method for structural optimization, Int. J. Numer. Methods Eng. 24 (1987) 359–373.
[33] Fengwen Wang, Boyan Stefanov Lazarov, Ole Sigmund, On projection methods, convergence and robust formulations in topology optimization, Struct. Multidiscip. Optim. 43 (6) (2011) 767–784.
[34] Xiaoping Qian, Ole Sigmund, Topological design of electromechanical actuators with robustness toward over- and under-etching, Comput. Methods Appl. Mech. Eng. 253 (2013) 237–251.
[35] Niels Aage, Erik Andreassen, Boyan Stefanov Lazarov, Topology optimization using petsc: an easy-to-use, fully parallel, open source topology optimization framework, Struct. Multidiscip. Optim. 51 (2015) 565–572.
[36] M.B. Fuchs, E. Moses, Optimal structural topologies with transmissible loads, Struct. Multidiscip. Optim. 19 (4) (2000) 263–273.