Generalized Topological Simplification of Scalar Fields on Surfaces

Julien Tierny and Valerio Pascucci, *Member, IEEE*

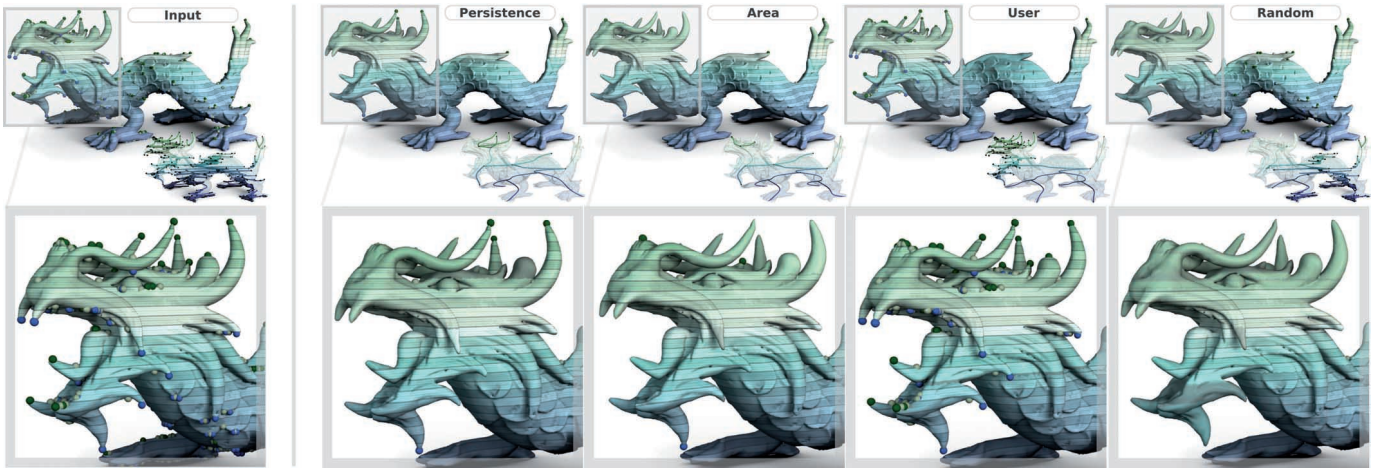


Fig. 1. Given an input scalar field f (left), our combinatorial algorithm generates a simplified function g that provably admits only critical points from a constrained subset of the singularities of f. Our approach is completely oblivious to the employed feature selection strategy, while guaranteeing a small distance $||f - g||_\infty$ for data-fitting purpose. Thus it supports application-dependent simplification scenarios such as the removal of singularities based on local geometrical measures, interactive user selection or even random selection. The topology of the resulting field is summarized with the inset Reeb graphs for illustration purpose.

Abstract—We present a combinatorial algorithm for the general topological simplification of scalar fields on surfaces. Given a scalar field f, our algorithm generates a simplified field g that provably admits only critical points from a constrained subset of the singularities of f, while guaranteeing a small distance $||f - g||_\infty$ for data-fitting purpose. In contrast to previous algorithms, our approach is oblivious to the strategy used for selecting features of interest and allows critical points to be removed arbitrarily. When topological persistence is used to select the features of interest, our algorithm produces a standard ϵ-simplification. Our approach is based on a new iterative algorithm for the constrained reconstruction of sub- and sur-level sets. Extensive experiments show that the number of iterations required for our algorithm to converge is rarely greater than 2 and never greater than 5, yielding $O(n\ log(n))$ practical time performances. The algorithm handles triangulated surfaces with or without boundary and is robust to the presence of multi-saddles in the input. It is simple to implement, fast in practice and more general than previous techniques. Practically, our approach allows a user to arbitrarily simplify the topology of an input function and robustly generate the corresponding simplified function. An appealing application area of our algorithm is in *scalar field design* since it enables, without any threshold parameter, the robust pruning of topological noise as selected by the user. This is needed for example to get rid of inaccuracies introduced by numerical solvers, thereby providing topological guarantees needed for *certified* geometry processing. Experiments show this ability to eliminate numerical noise as well as validate the time efficiency and accuracy of our algorithm. We provide a lightweight C++ implementation as supplemental material that can be used for topological cleaning on surface meshes.

Index Terms—Scalar field visualization, scalar field design, topological simplification.

1 Introduction

As scientific data-sets become more intricate and larger in size, advanced data analysis algorithms are needed for their efficient visualization. For scalar field visualization, topological analysis techniques have shown to be practical solutions in various contexts by enabling the concise and complete capture of the structure of the input data into high-level *topological abstractions* such as contour trees [5, 6], Reeb graphs [16, 20], or Morse-Smale complexes [8, 13]. Moreover, important advances have been made regarding the analysis of topological noise with the formalism of topological persistence [9]. Persistence offers a simple extension to the existing topological abstractions, enabling their multi-resolution representations and consequent progressive data explorations. However, since the notion of feature is application-dependent, in many scenarios using persistence to prioritize topological cancellations can be inappropriate for selecting features of interest (depending on the characteristics of the noise). For this reason, users often employ ad-hoc feature identification strategies that combine several criteria to determine which topological cancellations should be considered signal or noise. For instance, Carr et al. [6] introduced simplification metrics based on local geometrical measures such as contour length, volume, hyper-volume, etc. For medical data, for example, these metrics have been shown to be much more effective than persistence.

- *Julien Tierny is with the CNRS at Telecom ParisTech, France. E-mail: tierny@telecom-paristech.fr.*
- *Valerio Pascucci is with the SCI Institute at the University of Utah, the Pacific Northwest National Laboratory and ViSUS Inc., USA. E-mail: pascucci@sci.utah.edu.*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

While existing simplification schemes produce multi-resolution representations of the topological abstractions, they do not perform an actual simplification of the underlying scalar field. This can often be useful for further analysis. Moreover, in contexts such as scalar field visualization or design, it is desirable to obtain a simplified version of the input field directly without having to compute a computationally expensive topological abstraction. Also, the time complexities of many topological abstraction algorithms are dependent on the number of critical points in the input field. Hence, simplifying the field beforehand can also be beneficial for more sophisticated topological analysis.

In this paper, we present a new combinatorial algorithm for the general simplification of scalar fields on surfaces. Our algorithm is simple, fast in practice, and more general than previous techniques. Given a scalar field f, our algorithm generates a simplified function g that provably admits only critical points from a constrained subset of the singularities of f, while guaranteeing a small distance $||f - g||_\infty$ for data-fitting purpose. In contrast to previous combinatorial approaches, our algorithm is oblivious to the strategy used for selecting features of interest and allows critical points to be removed arbitrarily (Fig. 1). In the special case where topological persistence is used as a feature identification criteria, our algorithm generates a standard ϵ-simplification [10]. The algorithm is simple to implement, handles surfaces with or without boundary, and is robust to the presence of multi-saddles (the input is not restricted to a true Morse function). Extensive experiments show the generality of our algorithm as well as its high performance. In particular, the iterative nature of the approach could require a large number of passes but in practice we have not found example requiring more than five iterations (normally only two are needed). For this reason the experimental results show an $O(n\ log(n))$ practical performance.

To demonstrate the use of our approach, we present applications in terrain simplification and scalar field design. We believe the latter application to be particularly appealing as the algorithm, without any threshold parameter, robustly removes topological noise that arises from the use of numerical solvers in traditional scalar field design. This additional quality control enables scalar field design with topological guarantees for *certified* geometry processing.

1.1 Related work

The direct simplification of scalar fields given topological constraints is a subject that has only recently received attention. Existing techniques can be classified into two (complementary) categories.

Numerical approaches aim at approximating a desired solution by solving partial differential equations, where a subset of the input singularities are used as topological constraints while smoothness constraints are often used to enforce geometry quality. The first work in this direction was presented by Bremer et al. [4], where simplified Morse-Smale complexes are used to guide an iterative and localized simplification of the field based on Laplacian smoothing. However, the actual simplification process needs a (simplified) Morse-Smale complex as an input, which is computationally expensive to obtain. Moreover, it converges slowly and provides no error bounds on the interior of the complex cells. In the context of geometry processing, approaches have been presented for the computation of smooth Morse functions with a minimal number of critical points [12, 15]. Patanè et al. [17] presented a general framework for the topology-driven simplification of scalar fields based on a combination of least-squares approximation and Tikhonov regularization. Weinkauf et al. [22] improved the work by Bremer et al. [4] with bi-Laplacian optimization, resulting in smoother (C^1) output fields.

However, numerical approaches have several common drawbacks. First, they are time consuming. Second, and more important, they are prone to numerical instabilities, either due to *(a)* the poor quality of the triangulation of the input domain, *(b)* the numerical sensitivity of the geometrical operators employed in the system of equations, or *(c)* numerical precision errors. As demonstrated in Sec. 5, these numerical instabilities may often generate additional critical points in the output field (potentially with high persistence), preventing it from strictly conforming to the input topological constraints.

Combinatorial approaches aim at providing a solution with provable correctness that is not prone to numerical instabilities. In a sense, they can be complementary to numerical techniques by fixing the possible numerical issues as a post-process. The first research in this direction, on a related yet different problem, can be attributed to Edelsbrunner et al. with the notion of ϵ-simplification [10]. Given a target error bound ϵ, the goal of their algorithm is to produce an output field everywhere at most ϵ-distant from the input, such that all the remaining pairs of critical points have persistence greater than ϵ. Their algorithm can be seen as an extension of early work on digital terrain processing, where only minimum-saddle persistence pairs were discarded [1, 19]. However, their algorithm is complicated and difficult to implement. Moreover, as persistence pairs are processed in order of their highest extremity, the same vertices are swept several times when cancelable persistence pairs are nested. Recently, Attali et al. [2] and Bauer et al. [3] presented independently a similar approach for ϵ-simplification computation. By using radix sort with fixed word size, these algorithms admit linear time complexity (in practice, $O(n\ log(n))$ with a classical quick sort). Also, by locally reversing the gradient of the field, the authors show that multiple persistence pairs can be cancelled with only one procedure, even if they are nested [3]. However, these approaches suffer from several drawbacks. First, their input is a filtration (or equivalently a Discrete Morse function [11]). Converting the output of these algorithms into piecewise linear (PL) functions (which is the standard scalar field representation for any application) requires an important subdivision of the surface mesh (one new vertex per edge and per face), which can increase the size of the mesh up to an order of magnitude. In many applications, such an important domain subdivision would not be acceptable. Also, these approaches only deal with closed surfaces. The authors address this issue by artificially closing each boundary component, which does not guarantee consistently processed boundaries in the PL output once the surface is re-opened afterwards. In contrast, our algorithm works on PL functions defined on surfaces with or without boundary. Additionally, it addresses a more general problem for which ϵ-simplification is a special case.

1.2 Contributions

This paper makes the following new contributions:

1. **Approach:** An approach for the topological simplification of scalar fields that does not rely on persistent homology. It yields a simpler, more intuitive, and general setting. We enumerate the critical points that are *non-removable* because of the topology of the domain. We consequently derive a strategy that supports the suppression of arbitrary *removable* critical points of the field. This enables the development of a more general simplification framework than previous approaches, for which ϵ-simplification is a special case.
2. **Algorithm:** An iterative, combinatorial simplification algorithm which is very simple to implement (only a few dozens of lines of C++ code). Given a set of user constraints on the extrema of the output function, our algorithm automatically identifies and removes the optimal set of saddles with regard to $||f - g||_\infty$, hence guaranteeing a small distance between the input and the output. In contrast to previous approaches, our technique works directly on PL scalar field representations, is robust to multi-saddles, and handles surfaces with or without boundary. The algorithm uses no computationally expensive topological abstraction, such as a Morse-Smale complex or even a contour tree; hence it is very fast in practice. Our extensive experiments on approximated worst-case scenarios show that this iterative algorithm rarely takes more than two iterations to converge.
3. **Application:** We introduce the foundations for a robust, parameter-free, technique allowing intuitive scalar field design with topological guarantees. The user can select an *arbitrary* set of extrema to preserve with the guaranteed removal of all others. This leads to a more intuitive user interaction that does not require the user to understand sophisticated topological concepts such as persistence. For traditional automated scalar field design,

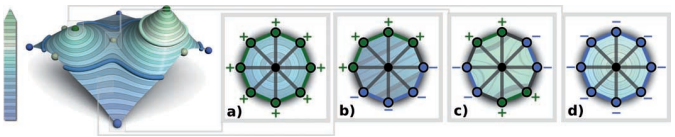


Fig. 2. Scalar field on a terrain (left). A level set is shown in blue; a contour is shown in white. Vertices can be classified according to the connectivity of their lower (blue) and upper links (green). From left to right: a minimum (a), a regular vertex (b), a saddle (c), a maximum (d).

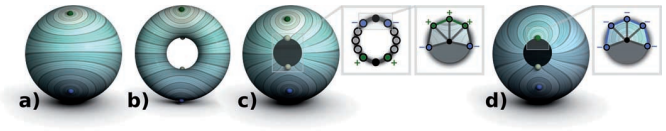


Fig. 3. Non removable critical points: (a) A global minimum and a global maximum have to be maintained for the field not to be constant. (b) $2\ g_{\mathcal{S}}$ saddles cannot be removed. Each boundary component has 2 non-removable global *stratified* extrema, which turn into non-removable saddles (c) or (possibly) *exchangeable* extrema (d).

numerical instabilities can easily lead to undesired topological noise. Our approach can be used to automatically fix this problem by simplifying any extraneous critical points introduced due to numerical instability. We provide a lightweight C++ implementation that can be used as a black-box post-process for fixing such problematic situations.

2 Preliminaries

This section briefly describes our formal setting and presents preliminary results. An introduction to Morse theory can be found in [14].

2.1 Background

The input to our algorithm is a piecewise linear (PL) scalar field $f : \mathcal{S} \to \mathbb{R}$ defined on an orientable PL 2-manifold $\mathcal{S}$. It has value on the vertices of $\mathcal{S}$ and is linearly interpolated on the simplices of higher dimension. Given an isovalue i, the level set $L(i)$ is defined as the inverse image of i onto $\mathcal{S}$ through f: $L(i) = f^{-1}(i)$. Each connected component of $L(i)$ is called a *contour*. The *sub-level set* $L^-(i)$ is defined as the inverse image of the open interval $(-\infty, i)$ onto $\mathcal{S}$ through f, $L^-(i) = \{p \in \mathcal{S} \ / \ f(p) < i\}$. Symmetrically, the *sur-level set* $L^+(i)$ is defined by $L^+(i) = \{p \in \mathcal{S} \ / \ f(p) > i\}$. As i changes continuously in $\mathbb{R}$, the points at which the topology of a contour changes are called *critical points*. In the PL setting, those occur on vertices. Critical points can be classified with simple and inexpensive operations (Fig. 2). The *star* $St(v)$ of a simplex v is the set of simplices σ that contain v as a face. The *link* $Lk(v)$ of a simplex v is the set of simplices in the closure of the star of v that are not also in the star: $Lk(v) = \overline{St(v)} - St(v)$. The *lower link* $Lk^-(v)$ of v is the subset of $Lk(v)$ containing only simplices with all their vertices lower in function value than v: $Lk^-(v) = \{\sigma \in Lk(v) \ / \ \forall u \in \sigma : f(u) < f(v)\}$. The *upper link* $Lk^+(v)$ is defined by: $Lk^+(v) = \{\sigma \in Lk(v) \ / \ \forall u \in \sigma : f(u) > f(v)\}$.

Definition 1 (Critical Point) *A vertex v of $\mathcal{S}$ is* regular *if and only if both $Lk^-(v)$ and $Lk^+(v)$ are simply connected, otherwise v is a critical point of f.*

If $Lk^-(v)$ is empty, v is a minimum. Otherwise, if $Lk^+(v)$ is empty, v is a maximum. If v is neither regular, nor a minimum nor a maximum, it is a saddle. Critical points can be classified by their *index*. For surfaces, minima, saddles, and maxima have index 0, 1 and 2 respectively. A sufficient condition for the above classification is that all the vertices of $\mathcal{S}$ admit distinct f values (no degenerate flat plateaus), which can be obtained easily with symbolic perturbation (Sec. 3). To simplify the discussion, we assume in the following that all of the saddles of f are simple (f is then a *Morse* function [14]) and that $\mathcal{S}$ is processed on a per connected component basis.

2.2 General simplification of scalar fields on surfaces

Definition 2 (General Topological Simplification) *Given a field $f : \mathcal{S} \to \mathbb{R}$ with its set of critical points $\mathcal{C}_f$, we call a* general simplification *of f a scalar field $g : \mathcal{S} \to \mathbb{R}$ such that the critical points of g form a sub-set of $\mathcal{C}_f$: $\mathcal{C}_g \subseteq \mathcal{C}_f$ (with identical indices and locations).*

It is often additionally desired that $||f - g||_\infty$ is minimized for data-fitting purpose. In other words, general simplification consists in constructing a close variant of the input field f from which a set of critical points has been removed. We describe the possible removals:

Closed surfaces The Morse-Euler relation defines a dependency between the number of critical points of f and $\chi_{\mathcal{S}}$ (the Euler characteristic of $\mathcal{S}$), where $\mathcal{C}_f^I$ is the set of critical points of f of index I:

$$\chi_{\mathcal{S}} = \sum_{I \in \{0,1,2\}} (-1)^I |\mathcal{C}_f^I| \ = \ \#_{min(f)} - \#_{saddles(f)} + \#_{max(f)} \quad (1)$$

It follows that removing only one extremum, such that the total number of critical points strictly decreases, implies the removal of one saddle (to maintain $\chi_{\mathcal{S}}$ invariant) and reciprocally. In other words, the removal of the saddles of f are dependent on the removal of its extrema. Certain saddles of f cannot be removed:

$$\chi_{\mathcal{S}} = 2 - 2g_{\mathcal{S}} = \#_{min(f)} - \#_{saddles(f)} + \#_{max(f)} \quad (2)$$

It follows that f counts exactly $2g_{\mathcal{S}}$ non-removable saddles, located along the $g_{\mathcal{S}}$ handles of the surface (Fig. 3(b)).

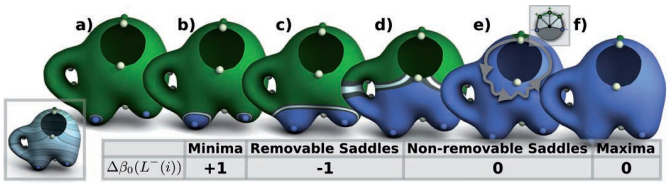
Surfaces with boundary The above properties are valid for surfaces with boundary. In addition, for each boundary component $\mathcal{B} \subseteq \partial\mathcal{S}$, certain saddles cannot be removed. Let $f_{\mathcal{B}}$ be the restriction of f to $\mathcal{B}$, and $\mathcal{C}_{f_{\mathcal{B}}}$ its critical points that we call *stratified* critical points. By construction, $\mathcal{B}$ is a closed PL 1-manifold. Then:

$$\chi_{\mathcal{B}} = \sum_{I \in \{0,1\}} (-1)^I |\mathcal{C}_{f_{\mathcal{B}}}^I| \ = \ \#_{min(f_{\mathcal{B}})} - \#_{max(f_{\mathcal{B}})} = 0 \quad (3)$$

It follows that $\mathcal{B}$ has an even number of stratified critical points. These cannot be regular points of f on $\mathcal{S}$. For instance, if a maximum of $f_{\mathcal{B}}$ is only surrounded on the interior of $\mathcal{S}$ by vertices with higher f values (Fig. 3(c), right), its lower link is by construction not simply connected; therefore it turns into a saddle of f. If it is only surrounded by vertices with lower f values (Fig. 3(d), right), then it turns into a maximum of f (otherwise it is a multi-saddle but f is assumed so far to have only simple saddles). The symmetric property holds for the minima of $f_{\mathcal{B}}$. Since f is required to admit distinct values for each vertex, $f_{\mathcal{B}}$ admits a pair of global stratified extrema (Fig. 3(c), middle). Consequently, each boundary component of $\mathcal{S}$ necessarily has a pair of critical points of f. If these two points are extrema, they can be removed only if they are not the only extrema of f, leaving a necessary saddle in place in exchange. Otherwise, these necessary critical points are non-removable saddles of f (Fig. 3(c), left). In conclusion, the removal of the saddles of f is completely dependent on the removal of its extrema, and for particular cases (summarized in Fig. 3) certain critical points are non-removable.

2.3 Surface scalar field constrained topology

As discussed in the previous sub-section, the removal of the saddles of f is dependent on the removal of its extrema. Then, given the sets of input constraints $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$ (the extrema of g), the target of general simplification is to constrain the topology of the level lines of f such that the output field g is close to f and admits as saddles a valid subset of $\mathcal{C}_f^1$ (such that the Morse-Euler relation still holds, Eq. 1). To constrain the topology of the level lines $L(i)$ of f, our strategy is to constrain the topology of both the sub- and sur-level sets of f ($L^-(i)$ and $L^+(i)$ respectively), which is more practical to achieve. We show in the following that, for surfaces, controlling the connectivity only of



	Minima	Removable Saddles	Non-removable Saddles	Maxima
$\Delta\beta_0(L^-(i))$	+1	-1	0	0

Fig. 4. Given a Morse function f^- admitting one maximum and several minima (left inset), the number of connected components of the sub-level set (in blue) increases when passing a minimum (b), decreases when passing a removable saddle (c) and does not change when passing the non-removable saddles (d, e) and the maximum (f).

$L^-(i)$ and $L^+(i)$ is sufficient to enforce the removal (or the preservation) of the removable critical points of f.

Let $f^- : \mathcal{S} \to \mathbb{R}$ be a Morse function with only one maximum and several minima (Fig. 4). Let $\beta_0(\mathcal{X})$ be the number of connected components of $\mathcal{X}$. Since f^- has only one maximum, $\beta_0(L^+(i)) = 1$ for all the i values under the maximum (each vertex of $L^+(i)$ has a non-empty upper link and thus admits a connected path to the maximum).

Minima A minimum at isovalue i has an empty lower link; then there exists no connected path on $L^-(i)$ linking this minimum to other lower minima. Thus, as i changes continuously in $\mathbb{R}$, when passing through a minimum of f^-, a new connected component of $L^-(i)$ has to be created and $\beta_0(L^-(i))$ increases (Fig. 4(b)).

Regular vertices The lower link of a regular vertex at isovalue i is made of one connected component. Then, *(a)* it cannot merge distinct components of $L^-(i)$ and *(b)* there exists connected paths on $L^-(i)$ linking it to lower minima. Thus, passing through a regular point does neither *(a)* decrease nor *(b)* increase $\beta_0(L^-(i))$.

Interior saddles By construction, the lower link of a simple saddle on the interior of $\mathcal{S}$ is made of two connected components (Fig. 2(c)). These components can either be linked to *(a)* the same or to *(b)* distinct connected components of $L^-(i)$. In the latter case *(b)*, $\beta_0(L^-(i))$ decreases when passing the saddle. In the former case *(a)*, neither $\beta_0(L^+(i))$ nor $\beta_0(L^-(i))$ changes ($\beta_0(L^+(i)) = 1$ for all i below the maximum). However, since f^- is Morse, $L(i)$ changes its topology at the saddle. In the interior of surfaces, the only possible topological change of $L(i)$ is a change of $\beta_0(L(i))$. Saddles which change $\beta_0(L(i))$ while preserving $\beta_0(L^+(i))$ and $\beta_0(L^-(i))$ have been shown to correspond to the saddles opening or closing the loops of the Reeb graph of the function [20] and for surfaces, these loops correspond to the handles of the surface [7]. Thus, the only interior saddles of f^- for which $\beta_0(L^-(i))$ does not change are the $2g_{\mathcal{S}}$ non-removable saddles (Fig. 3(b) and Fig. 4(d)).

Boundary saddles Simple boundary saddles can be classified in two categories. In the first case (called *join* saddles), the lower link is made of two components (each lying on the same boundary component $\mathcal{B} \subseteq \partial\mathcal{S}$) and the upper link of one (Fig. 4(e)). The second case (*split* saddles) is symmetric: the lower link is made of one component and the upper link is made of two components on the boundary. Since $\beta_0(L^+(i)) = 1$ (f^- has only one maximum) and since their lower link is made of only one component, neither $\beta_0(L^+(i))$ nor $\beta_0(L^-(i))$ changes when passing split saddles. For a join saddle, noted s_j, the two components of the lower link can either be linked to *(a)* the same or to *(b)* distinct connected components of $L^-(i)$. In the latter case *(b)*, $\beta_0(L^-(i))$ decreases when passing s_j. Otherwise *(a)*, neither $\beta_0(L^-(i))$ nor $\beta_0(L^+(i))$ changes and there exists a connected path on $L^-(i)$ connecting the two components of the lower link of s_j. This path encloses the boundary component $\mathcal{B}$ on which s_j lies (Fig. 4(e)). This implies that $\{\mathcal{B} - s_j\} \subset L^-(i)$ since $L^+(i)$ is made of only one component (the presence of a vertex on $\mathcal{B}$ with a value higher than i would then imply that $\beta_0(L^+(i)) > 1$). Thus, s_j is the *stratified* global maximum of $f^-_{\mathcal{B}}$. In other words, for each boundary component $\mathcal{B} \subseteq \partial\mathcal{S}$, the only join saddle of f^- for which $\beta_0(L^-(i))$ does not change is a non-removable saddle (Fig. 3).

Maximum The lower link of a maximum is made of only one connected component. Then, a maximum cannot merge or create a new component of $L^-(i)$. Thus $\beta_0(L^-(i))$ does not change when passing through a maximum (Fig. 4(f)).

The symmetric properties hold for a Morse function $f^+ : \mathcal{S} \to \mathbb{R}$ admitting only one minimum and several maxima. Since the input fields are assumed to be Morse functions, when passing through a given critical point, only one topological event can occur at a time on the level set [14], which enables the viewing of each critical point as an instance of the cases reviewed above. Then, in conclusion, the only critical points of the field for which both $\beta_0(L^-(i))$ and $\beta_0(L^+(i))$ do not evolve are the non-removable saddles due to the topology of $\mathcal{S}$ (Sec. 2.2); for the removable critical points of f, either $\beta_0(L^-(i))$ or $\beta_0(L^+(i))$ changes. Thus, it is possible to constrain the topology of the output field g by only controlling the connectivity of $L^-(i)$ and $L^+(i)$. The next section presents an algorithm exploiting this property.

3 Algorithm

Our algorithm naturally follows from the properties discussed in Sec. 2. Given the constraints $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$, it iteratively reconstructs the corresponding sub- and sur-level sets, while removing the optimal set of saddles with regard to the L_∞ norm. The remainder of this section is organized as follows. The algorithm, which consists in alternating sub- and sur-level set constrained reconstruction, is described in Sec. 3.1, where it is shown to converge to an output field whose topology conforms to the input constraints $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$. Then, in Sec. 3.2, the properties of the algorithm are discussed.

Algorithm 1: Sub-level set constrained reconstruction

input : Scalar field $f : \mathcal{S} \to \mathbb{R}$ (with n scalar (f) and offset ($\mathcal{O}$) values);
input : Set of minima constraints to enforce $\mathcal{C}_g^0$;
output: Scalar field $g : \mathcal{S} \to \mathbb{R}$ with enforced minima in $\mathcal{C}_g^0$.

```
1  begin
2      // T: set of vertices (self-balancing binary search tree).
3      T ← ∅;
4      // i: time (integer) when a vertex was last processed.
5      i ← 0;
6
7      // Initialize T with the minima constraints.
8      foreach m ∈ C_g^0 do T ← {T + m};
9
10     repeat
11         v ← argmin_{x∈T} f(x);
12         T ← {T − {v}};
13         mark v as visited;
14         // Add unvisited neighbors.
15         T ← {T ∪ {v_n ∈ Lk(v) | v_n is not visited}};
16         A[i] ← v;
17         i ← i + 1;
18     until T = ∅;
19
20     // Scalar and offset value update, for all the vertices.
21     // Make the ordering on g (scalar and offset values) consistent with the order of visit.
22     for j ← 0 to n do
23         if j ≠ 0 && f(A[j]) < g(A[j − 1]) then
24             g(A[j]) ← g(A[j − 1]);
25         else
26             g(A[j]) ← f(A[j]);
27         O(A[j]) ← j;
28 end
```

3.1 Algorithm description

In the following, we start by describing the algorithm for sub-level set constrained reconstruction.

Sub-level set constrained reconstruction The pseudo-code for sub-level constrained reconstruction is given in Algorithm 1. To guarantee that the input field admits distinct values on each vertex, symbolic perturbation is used. In addition to its scalar value, each vertex v is associated with an integer *offset* (noted $\mathcal{O}(v)$) initially set to the actual offset of the vertex in memory. When comparing two vertices (for critical point classification for instance), if these share the same scalar value, their order is disambiguated by their offset $\mathcal{O}$. Algorithm 1 modifies both vertex scalar values and offsets.

The algorithm starts by pushing the minima constraints $\mathcal{C}_g^0$ into a self-balancing binary search tree (noted $\mathcal{T}$, line 8) ordered by scalar value and offset. Then, the sub-level sets are iteratively reconstructed

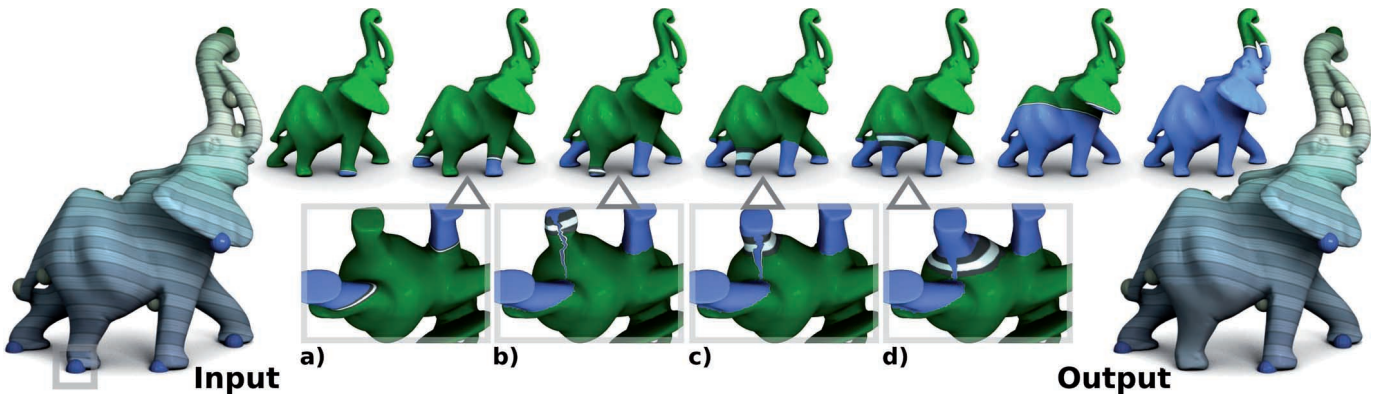


Fig. 5. Removing the lowest minimum (small box on the input) by sub-level set constrained reconstruction (in blue). The algorithm enforces the minima constraints while implicitly removing saddles.

(line 10 to 18), one vertex at a time, in a flooding fashion: the unvisited neighbors of the visited vertex are added to $\mathcal{T}$ and the vertices of $\mathcal{T}$ are uniquely visited in increasing order of scalar value (and offset) until the entire domain is processed. For instance, Fig 5 shows the removal of the lowest minimum of f. Hence, all the minima of f have been added to $\mathcal{C}_g^0$ except for the global minimum. The corresponding flooding is progressively shown in the middle of Fig. 5, where the visited and unvisited vertices appear in blue and green respectively. The resulting order of visit of each vertex is stored in the array $\mathcal{A}$. The last step of the algorithm (line 22 to 27) traverses $\mathcal{A}$ and updates the vertex scalar values and offsets such that the order defined by the output field is equivalent to the order of visit of the vertices (then the sub-level sets $L^-(i)$ of the output field indeed correspond to the iteratively reconstructed sub-level sets). As shown in Fig. 5 (right), this strategy for function value update has the effect of flattening the output in the vicinity of the removed minima.

Since the sub-level sets are grown by adding the vertices of $\mathcal{T}$ with smallest function value first, if a connected component of $L^-(i)$ were to hit a minimum constraint $m \in \mathcal{C}_g^0$ before the latter was popped out of $\mathcal{T}$, this would imply that m had a neighbor with lower initial function value, through which the component entered the link of m. This implies that m was not a minimum in the input, which is an invalid constraint. Then, for each $m \in \mathcal{C}_g^0$, m is visited before the vertices of its link; hence the vertices of $\mathcal{C}_g^0$ are all minima in the output. The other vertices which do not belong to $\mathcal{C}_g^0$ can only be visited by the iterative growth of $L^-(i)$, after that one of the vertices of their link has been visited. Hence, their lower link is not empty in the output and the vertices $m \in \mathcal{C}_g^0$ are then the only minima of the output (Fig. 5).

When passing a saddle s of the input which used to join distinct components of sub-level sets, if only one component of $L^-(i)$ hits the saddle, this implies that the minimum which created initially the other component does not belong to the constraints $\mathcal{C}_g^0$. Then, the component of $L^-(i)$ traverses s and continues to grow by visiting vertices with smallest values first, eventually sweeping the removed minimum (Fig. 5(b)-(d)). Otherwise, s is maintained as a saddle in the output.

Hence, the algorithm implicitly removes saddles given the extrema constraints and guarantees a valid topology of the output. Note that, since the algorithm visits the vertices of $\mathcal{T}$ with smallest value first, the saddle removed with one minimum m is the lowest saddle s which used to join the sub-level set component created by m in the input (i.e. the next saddle from m up the join tree [5]): thus the algorithm minimizes $|f(m) - f(s)|$ when removing one saddle s along with one minimum m. Since the update of function value will lift m up to the level of s ($g(m) \leftarrow f(s)$, line 22 to 27 of algorithm 1), $||f-g||_\infty$ will be equal to $|f(m) - f(s)|$ (for instance, in Fig. 6, B is lifted up to the level of E). Thus the algorithm removes the optimal set of saddles with regard to $||f-g||_\infty$, hence guaranteeing a small distance between the input and the output (for the regions where no simplification is needed, the function is unaltered).

Sur-level set constrained reconstruction The constraints $\mathcal{C}_g^2$ are enforced with the symmetric algorithm: the vertices of $\mathcal{C}_g^2$ are initially pushed in $\mathcal{T}$ and the vertices of $\mathcal{T}$ are visited in decreasing order of function value (the update of the function values is also symmetric).

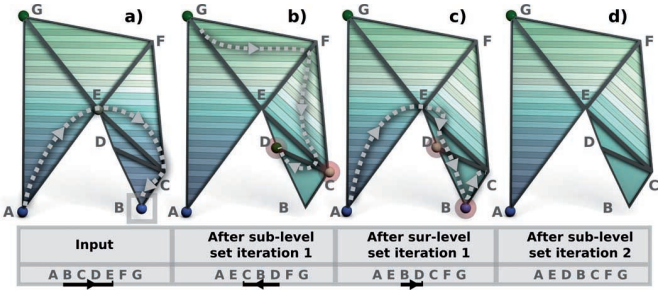


Fig. 6. Sub-level set constrained reconstruction can introduce residual maxima (red spheres): in (a), all the neighbors of D are visited before it, hence yielding a maximum (b). Symmetrically, in (b), all the neighbors of B are visited before it, yielding a minimum (c). Alternating sub- and sur-level set reconstruction reduces the (offset) function difference between the residual extrema and their corresponding saddle (cf. vertex ordering, bottom), and converges to the removal of all the residuals.

Overall algorithm As shown in Fig. 6, while algorithm 1 guarantees that the constraints $\mathcal{C}_g^0$ will be the only minima of the output, it does not guarantee that $\mathcal{C}_g^2$ will be the only maxima. When the reconstructed sub-level set removes a saddle, the algorithm visits the vertices of lowest function value in priority, possibly leaving islands of non-visited vertices behind (Fig. 6(a)), yielding residual maxima in the output function (Fig. 6(b)). The symmetric remark goes for the sur-level set reconstruction regarding minima constraints. To remove these residuals, our overall algorithm successively alternates sub- and sur-level set reconstructions until $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$ are the only extrema. We show in the following that this process converges.

Convergence Algorithm 1 lifts up the minima to remove, since they are visited after their associated saddle (Fig. 6(a)). Then, when removing a minimum m (B, Fig. 6(a)), residual critical points can only occur higher than the minimum's associated saddle (E, Fig. 6(a)), but lower than its next vertices in the global vertex ordering (F and G, Fig. 6(a)). Symmetrically, sur-level set reconstruction pushes down the maxima to remove. Then new residual critical points (B and D, Fig. 6(c)) occur lower than the residual saddle of the previous step (C, Fig. 6(b)), but still higher than the original (E, Fig. 6(a)). Alternating sub- and sur-level set reconstruction will keep on reducing the function range where the residual extremum and its corresponding saddle appear. Eventually these will be consecutive in the global vertex ordering (Fig. 6, bottom) leaving no more room for new residuals at the next iteration.

From symbolic to numerical perturbation After convergence, it may be useful to convert the symbolic perturbation (vertex offset $\mathcal{O}(v)$) into numerical perturbation, to represent the output field g with a numerical value only for each vertex. The final array $\mathcal{A}$ (algorithm 1)

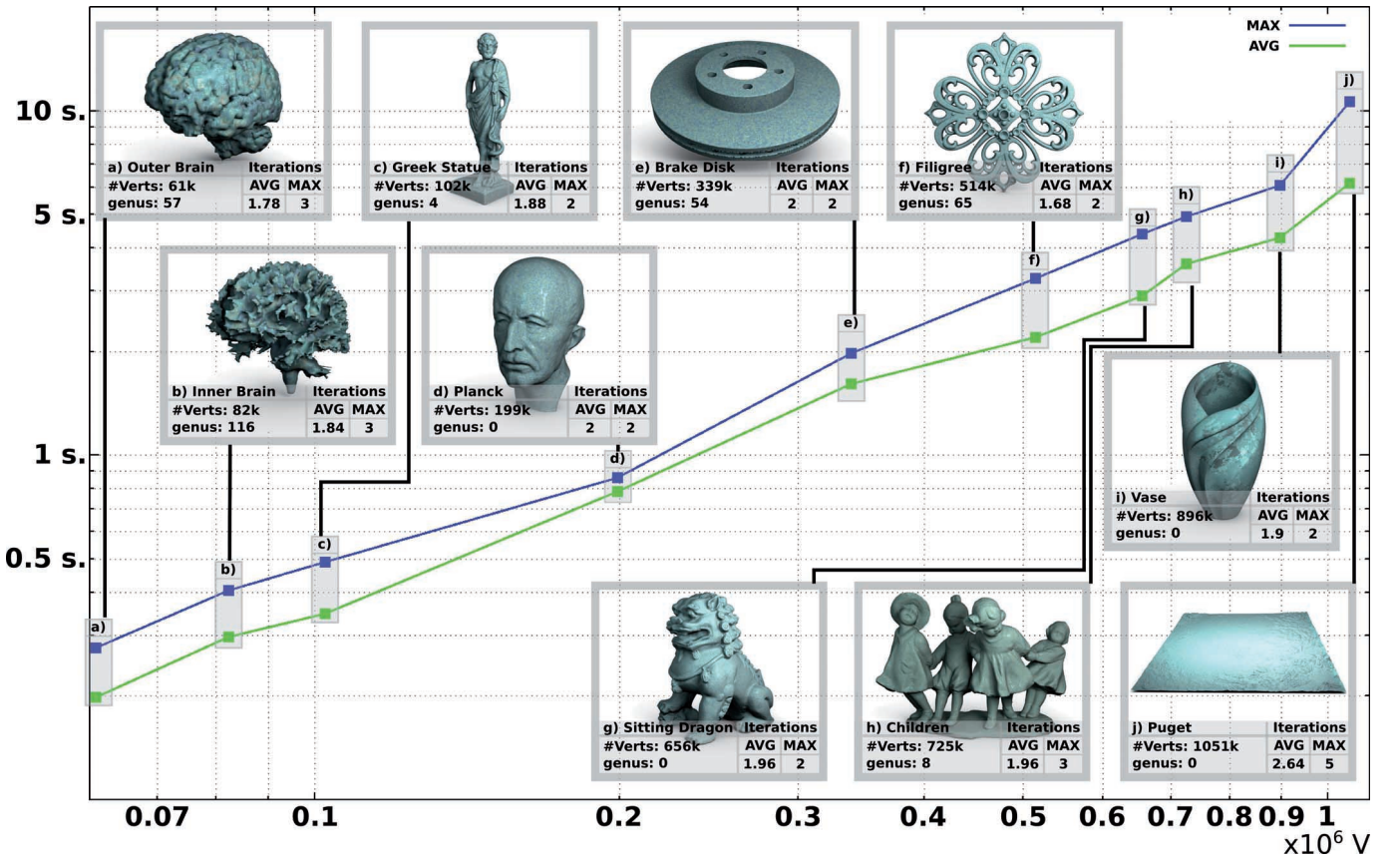


Fig. 8. Running times (log scale) for the simplification of random functions, with a random constraint selection ($\mathcal{C}_g^0$, $\mathcal{C}_g^2$), 50 runs per data-set.

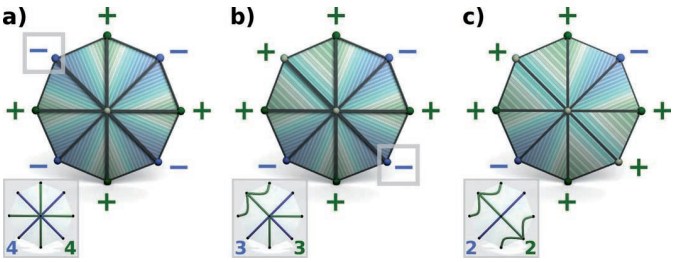


Fig. 7. Simplifying a saddle with high multiplicity (a): 4 components of sub- and sur-level sets merge in the saddle (inset Reeb graph: sub- and sur- level sets are marked in blue and green respectively). Removing one extremum (box in (a)) decreases the number of components of the lower and upper links by 1 (b). Removing other extrema (box in (b)) eventually decreases this number to 2, yielding a simple saddle (c).

is traversed in increasing order and whenever a vertex is at the same value (or lower) than its predecessor ($g(A[i]) \leq g(A[i-1])$), its function value is increased by an arbitrarily small value ξ: $g(A[i]) \leftarrow g(A[i-1]) + \xi$. This numerical perturbation should be restricted only where it is needed (flat regions of g) to maintain $||f-g||_\infty$ to a small value. For instance, in Fig. 6, the vertices D, B and C should all have a final function value in the interval $(f(E), f(F))$. Hence, ξ should be smaller than $\frac{\delta_f}{n}$, where δ_f is the smallest (non-zero) function value absolute difference in the input and n is the number of vertices in $\mathcal{S}$.

3.2 Algorithm properties

Relation to ϵ-simplification The implicit pairing performed by our algorithm is compatible with the pairing of critical points based on persistence: given one extremum removal, it pairs a minimum (respectively, a maximum), with its closest saddle up the join tree [5] (respectively, down the split tree). Moreover, given one extremum removal, $||f-g||_\infty$ will be equal to the absolute difference in function value between the extremum and its paired saddle (line 22 to 27 of algorithm 1). For instance in Fig. 6, B is paired with E and $||f-g||_\infty$ is equal to $|f(B) - f(E)|$. Thus, if the input constraints are selected according to topological persistence (the persistence of the pairs associated with each critical point of $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$ is higher than ϵ), then $||f-g||_\infty \leq \epsilon$.

Non-Morse inputs Multi-saddles may occur in the input, preventing f from being a Morse function. For these, the lower and upper links can be made of more than two components. Our algorithm handles these degenerate cases with no modification: removing one extremum associated with a multi-saddle will simply decrease the saddle's multiplicity in the output by one (Fig. 7).

4 Results and Discussion

In this section, we present practical results of our algorithm obtained with a C++ implementation on a computer with an i7 CPU (2.93 GHz).

4.1 Time requirement

The algorithm uses no computationally expensive topological abstraction, such as a Morse-Smale complex or even a contour tree. Therefore each iteration is extremely fast in practice. Given a surface with n vertices, inserting and removing a vertex from the self-balancing binary search tree $\mathcal{T}$ takes $O(log(n))$ time. Each vertex is uniquely visited. Thus, the complexity of an iteration is $O(n\ log(n))$, irrespectively of the number of critical points to remove. In theory, the number of iterations required for the algorithm to converge could possibly be non-negligible. Given one extremum removal, after each reconstruction, the distance in the global vertex ordering which separates a new residual extremum from its corresponding saddle decreases at least by one (this distance is illustrated by a black arrow in Fig. 6, bottom). As this distance can initially be close to the number of vertices in the mesh, n reconstructions could be required, yielding an $O(n^2\ log(n))$ worst case complexity, irrespectively of the number of removed extrema.

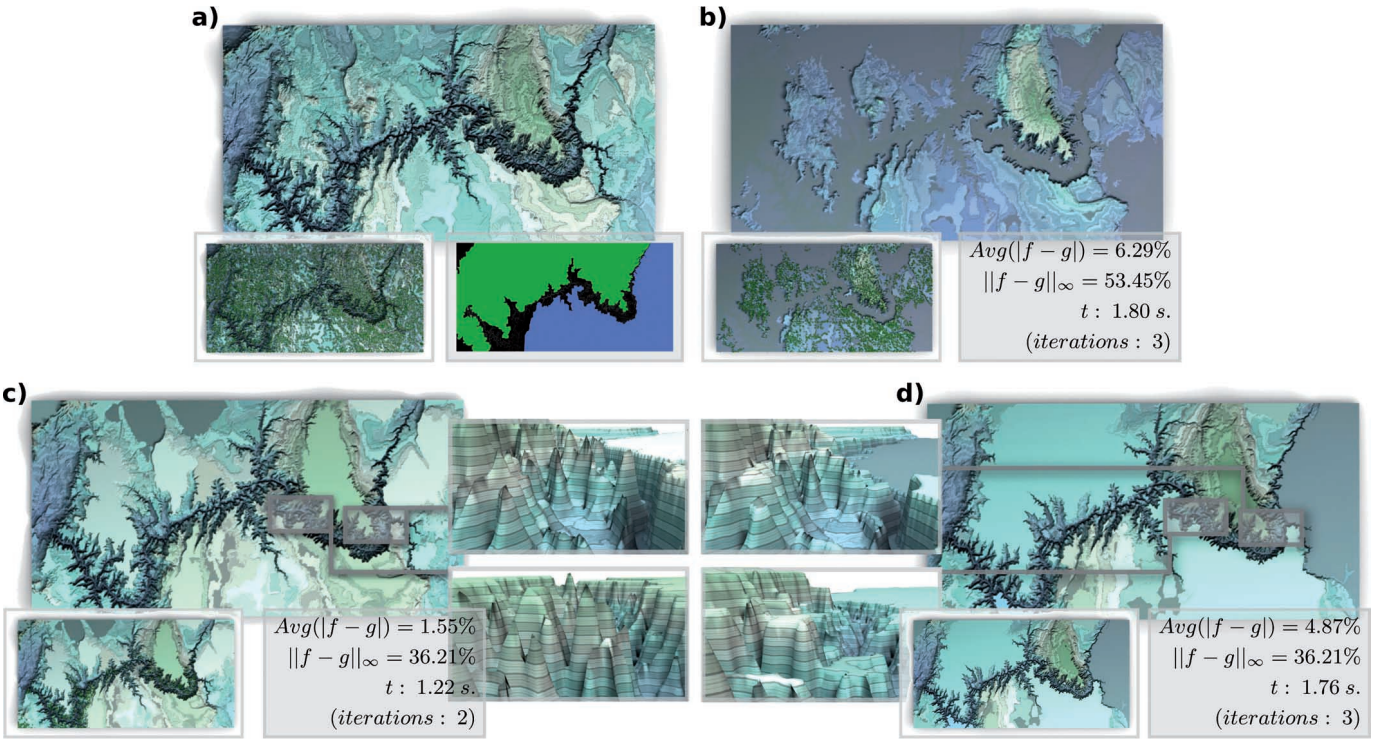


Fig. 10. Simplifying the Grand Canyon ((a), 500k vertices, 65,526 critical points, bottom) with a location driven feature selection. The terrain is decomposed into three geographically meaningful regions: the canyon, its north rim and its south rim (in black, green, blue, (a) bottom). (b) Maintaining only one minimum and removing all the critical points from the canyon (16,756 critical points remain) emphasizes the topological features of the rims and simulates a massive flooding of the canyon. (c) Removing all the critical points from the rims (2,296 remaining) emphasizes the topological features inside the canyon in a way that is suited for an interactive fly-through within the canyon. An ϵ-simplification with compatible L_∞ norm (d) completely discards the features irrespectively of their location (zoom insets) while yielding a worse average data fitting ($Avg(|f-g|)$).

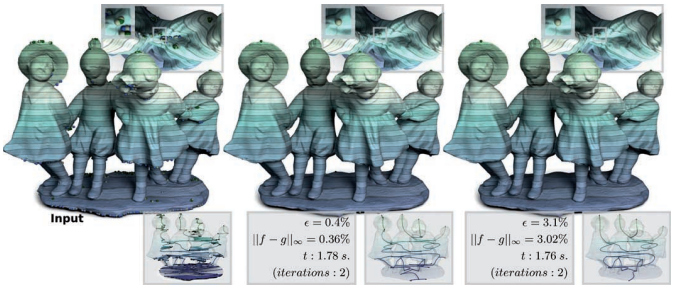


Fig. 9. In the special case where the critical points in $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$ are all more persistent than ϵ, our algorithm produces an ϵ-simplification ($||f-g||_\infty \leq \epsilon$). Both the (vertex) position and the function value of the remaining removable critical points are preserved after simplification (top insets), even in the presence of multi-saddles (6 in the input). The topology of the resulting field is summarized with the inset Reeb graph for illustration purpose (input surface: 725k vertices).

Traditionally, random scalar fields are considered as relevant approximations of worst-case scenarios. Moreover, considering multiple instances increases the chances to get a proper worst-case approximation. We show in Fig. 8 the average and maximum running times (in log scale) for the algorithm to achieve convergence on a set of meshes (including examples with high genus, up to 116), for which 50 instances of random fields have been considered and for which the constraints $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$ are random subsets of the fields' extrema. In particular, critical points from $\mathcal{C}_f^0$ are uniquely added to $\mathcal{C}_g^0$ in random order until $|\mathcal{C}_g^0|$ is equal to a random fraction of $|\mathcal{C}_f^0|$ (the constraint set $\mathcal{C}_g^2$ is constructed similarly). For most data-sets, the average number of required iterations is smaller than or equal to 2 and the maximum number of iterations is never greater than 5. This shows that, from a practical point of view, the number of required iterations is negligible with regard to n, hence yielding $O(n\ log(n))$ practical running time. In our experiments, the algorithm took at most 10.7 seconds to compute on a mesh with 1 million vertices (6.3 million simplices total).

4.2 Discussion

A unique aspect of our algorithm is its ability, given the constraints $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$, to automatically identify and remove the optimal set of saddles with regard to the L_∞ norm. Moreover, this is accomplished without the need to carry a union-find data-structure unlike previous techniques. Although this data-structure has nearly linear time complexity in theory, practically it could cause slowdowns by a non-negligible constant factor, given the algorithm's low resource requirement. Also, after simplification, the (vertex) position of the remaining critical points is preserved. In the special case where the critical points of $\mathcal{C}_g^0$ and $\mathcal{C}_g^2$ are selected based on topological persistence, the algorithm produces a standard ϵ-simplification, as shown in Fig. 9. In contrast to previous approaches, our algorithm directly works on a PL representation of the field, which is more acceptable application-wise. Importantly, it is also more general as critical points can be removed arbitrarily (at the exception of the non-removable critical points summarized in Fig. 3), irrespectively of the employed feature-selection strategy.

4.3 Limitations

Given our formulation of general simplification (Sec. 2.2), for specific constraint configurations, the value of the remaining critical points may change after simplification. For instance, if the lowest minimum in $\mathcal{C}_g^0$ is initially higher than the highest maximum in $\mathcal{C}_g^2$, the algorithm will change their values to satisfy the topological constraints. Also, when removing only one extremum from the boundary, it will be replaced by a boundary saddle if it is associated by the algorithm (with regard to the L_∞ norm) with an interior saddle (Fig. 7). This is due to the fact that the number of critical points must be even on each

boundary component (Sec. 2.2), which may be counter-intuitive from a user's perspective. Moreover, whereas our algorithm removes the optimal set of saddles with regard to $||f-g||_\infty$ given some extrema constraints, our strategy for function value update (which is purely based on *flooding*) does not guarantee a minimization of $||f-g||_\infty$, although the resulting L_∞ norm is close to the theoretical minimum. In the context of persistence driven simplification, Bauer et al. [3] showed that optimality could be reached by a combination of *carving* (i.e. pulling the saddles halfway towards their associated extremum) and *flooding* (i.e. pushing the extrema halfway towards their associated saddle) at the expense of no longer guaranteeing a function value lock on the maintained critical points. By simplifying all the pairs less persistent than ϵ, their approach yields $||f-g||_\infty \leq \frac{\epsilon}{2}$. In contrast, like in [10], our approach yields $||f-g||_\infty \leq \epsilon$ but locks the maintained critical points in terms of function value. However, in the context of general simplification, the optimal balance between carving and flooding will be more difficult to evaluate, as it is no longer a local decision which depends only on the pair of removed critical points (excessive carving could break the enforcement of near-by extrema located in the vicinity of the removed saddles). Finally, like any combinatorial approach, our algorithm provides strong guarantees on the topology of the output at the expense of its geometrical smoothness. Unlike previous combinatorial algorithms using *carving* [2, 3, 10], our algorithm uses *flooding* only. Hence, it does not suffer from the usual artifacts of carving (visible thin paths linking sets of removed critical points), but from those of flooding (flat regions in the output, Fig. 5, right). In our experiments, we found that these artifacts were usually little noticeable, unless the features removed by the user span a large region of the domain (Fig. 5, right). If smoothness is desired, our approach can be combined with a numerical technique (cf. Sec. 1.1) to provide smooth outputs that still benefit from the topological guarantees of our algorithm (see Sec. 5.2).

5 Applications

To demonstrate the utility of our algorithm, we present two applications benefiting from a general scalar field simplification approach.

5.1 Terrain simplification with ad-hoc feature selection

Topological simplification of terrains can be particularly useful for topographic analysis or water flow simulations as discussed by Bremer et al. [4]. However, the original topological persistence measure can be unsatisfactory for selecting features in such a context, as the characterization of features of interest is application dependent and at times can also be data-set dependent. Fig. 10 exemplifies this observation on the Grand Canyon elevation data-set, which initially counts 65,526 critical points. The Grand Canyon can be decomposed in three major regions from a geographic point of view: the canyon itself, its north rim and its south rim. These regions (in black, green and blue respectively in Fig. 10(a)) have been initially extracted by segmenting the image along high elevation gradient and interactively completed by the user. Based on this initial decomposition, we present two simplification scenarios. First, in Fig. 10(b), the topology of only the rims has been emphasized: only the lowest minimum above 53% of the total elevation difference has been maintained, the critical points inside the canyon have been selected for removal and all the maxima on the rims above 53% of elevation have been maintained. In less than 2 seconds, our algorithm constructs the corresponding *flooded* Grand Canyon, while enforcing the preservation of the selected topological features on the rims. A contrary scenario would consist in emphasizing the peaks inside the canyon. For instance, the result of such a simplification strategy can drive a mesh simplification procedure for an interactive fly-through within the canyon. In that case (Fig. 10(c)), only the global minimum has been maintained and all the maxima outside of the canyon have been selected for removal. Note that in comparison, a standard ϵ-simplification with compatible L_∞ norm (Fig. 10(d)) is unsatisfactory as it completely discards the topological features irrespectively of their location (zooms in Fig. 10), while yielding a worse average data-fitting ($Avg(|f-g|)$).

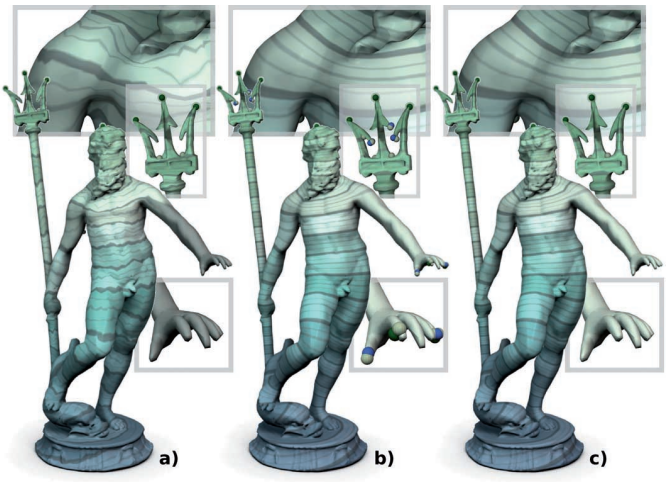


Fig. 11. Solving the Laplace equation with topological guarantees (transparent spheres are Dirichlet constraints). The combinatorial Laplacian (a) satisfies the topological properties of the solution, but it has a poor geometrical accuracy (level lines, top). The cotangent weight Laplacian (b) provides an improved geometrical approximation (top) but is numerically sensitive and generates invalid additional singularities (inset zooms). Our algorithm can be applied as a post-process (c), with no threshold parameter, to remove these inconsistent critical points. The resolution of the equation took 0.14 s. (surface: 25k vertices), while the combinatorial simplification took 0.02 s. (1 iteration). Our algorithm provides a solution (c) which both benefits from the geometrical approximation quality of cotangent weights ($||f-g||_\infty = 0.12\%$) and from the topological stability of the combinatorial Laplacian.

5.2 Scalar field design with topological guarantees

Many geometry processing problems involve numerically sensitive tasks such as partial differential equation resolution, gradient field integration, or scale-space computation. In many cases, the topology of the numerical solution is a major consideration. In meshing for instance, extraordinary vertices often correspond to singularities and these important constraints must be respected. However, numerical noise often occurs and can alter the topology of the solution. We illustrate this issue with the Laplace equation subject to Dirichlet boundary conditions. Beyond its ubiquity in geometry processing, this equation plays an important role in electromagnetism, astronomy and fluid dynamics. Given a finite set of extrema constraints $\mathcal{D}$ along with corresponding target values, the solution f to this equation is defined as follows:

$$f(d_i) = f_{d_i} \quad \forall d_i \in \mathcal{D} \tag{4}$$

$$\Delta f(v) = 0 \quad \forall v \notin \mathcal{D} \tag{5}$$

where Δ stands for a discretization of the Laplacian operator on surfaces. An important property of this equation is that the Dirichlet constraints $\mathcal{D}$ should be the only extrema of the solution.

However, since the Laplacian is a second-order operator, it is difficult to discretize for piecewise linear functions. Hence several discretization strategies have been proposed (see [21] for a comprehensive discussion). Fig. 11 shows the solution of this equation for two discretizations of the operator, obtained by least-squares optimization with the penalty method [23]. The combinatorial Laplacian [21] (Fig. 11(a)) is a straightforward discretization which exhibits robust topological properties. However, as it is strongly biased by the discretization of the mesh, it fails at generating smooth level sets. In contrast, the discretization based on cotangent weights [18] produces much smoother level sets. However, in practice, surface triangulations often include many sharp triangles. As edge angles get closer to zero, the numerical error on their tangent evaluation can be arbitrarily amplified when used as the denominator for the cotangent computation, hence yielding an error of arbitrarily high amplitude in the solution in the vicinity of sharp triangles. As shown in Fig. 11(b), this numerical

error generates additional critical points (with non-zero persistence), which prevents the solution from conforming to its formal description.

Our algorithm can be used in a straightforward manner to fix these numerical instabilities by using the Dirichlet constraints as topological constraints ($\mathcal{C}_g^0$ and $\mathcal{C}_g^2$). In practice, our algorithm is around an order of magnitude faster than the actual numerical optimization (using CholMod). Thus it can be used as a post-process with a negligible computation time overhead. Note that since our algorithm uses no threshold parameter, it can be used in a robust manner, irrespective of the amplitude of the numerical error. As shown in Fig. 11(c), our algorithm automatically removes topological noise while minimally affecting the function. Therefore, our approach can be used to generate a solution with both the geometrical accuracy of the cotangent weight Laplacian and the topological robustness of the combinatorial Laplacian, yielding a solution with topological guarantees that is exploitable for *certified* geometry processing.

Note however that due to its combinatorial nature, our algorithm will locally break the harmonicity of the function in the vicinity of the removed critical points by flattening the area. This drawback has only a local impact and a small amplitude (in Fig. 11(c), $||f - g||_\infty = 0.12\%$) and thus will be acceptable for most applications (like quad-mesh design for example). However, it might be a limitation in specific applications where harmonicity is a critical feature that must be enforced everywhere, like harmonic parameterization for instance.

6 Conclusion

In this paper, we have presented a combinatorial approach for the general simplification of piecewise linear scalar fields on surfaces. By abstracting our approach from the concepts of persistent homology, we believe to have presented a simpler, more intuitive and more general description of scalar field topological simplification. Also, we enumerated all the configurations for which critical points were *non-removable* given the topology of the domain, for surfaces with or with out boundary. From this, we derived a strategy that allows for the arbitrary suppresion of the *removable* critical points of the field.

We presented a simple iterative algorithm for general topological simplification which, given some constraints on the extrema of the output field, implicitly identifies and removes the optimal set of saddles with regard to the L_∞ norm, hence guaranteeing a small distance $||f - g||_\infty$. Although it is iterative, extensive experiments on approximated worst-case scenarios showed that the algorithm rarely takes more than 2 iterations to converge. The algorithm uses no computationally expensive topological abstraction, such as a Morse-Smale complex or even a contour tree; hence it is very fast in practice. It can be used as a complement to existing numerical approaches to provide strict topological guarantees on the output. In contrast to previous combinatorial approaches, our approach works directly on a PL representation of the field (which is more acceptable application-wise), is robust against multi-saddles, and handles surfaces with or without boundary. Moreover, our approach solves a more general problem, for which ϵ-simplifications have been shown to be a special case.

We demonstrated the utility, accuracy, and efficiency of our contribution in two applications. In the context of scalar field design, we showed that our algorithm could combine the practical individual advantages of different discretizations of the Laplace operator, hence providing, without any threshold parameter, topological guarantees for certified geometry processing. Other numerical problems on surfaces, including other partial differential equation systems, gradient field integration or scale-space computations can benefit from this post-processing black-box and we refer the reader to the C++ code provided in appendix for usage examples.

A natural direction for future work is the extension of this approach to volumetric data-sets. However, as the dimension of the domain increases, new types of saddle points appear and more subtle topological transitions occur on the level sets. Hence, enforcing the connectivity of the sub- and sur-level sets is insufficient; the genus of the iso-surfaces also needs to be efficiently controlled.

Acknowledgements

The authors are grateful to Delila Omerbašić and Brian Summa for their careful proofreading and Sophie Masse for the graphical design of our demo program's user interface. We would also like to thank the anonymous reviewers for their thoughtful remarks and suggestions. Data-sets are courtesy of AIM@SHAPE and the Large Geometric Models Archive at Georgia Tech.

References

[1] P. Agarwal, L. Arge, and K. Yi. I/O-efficient batched union-find and its applications to terrain analysis. In *ACM Symp. on Comp. Geom.*, pages 167–176, 2006.

[2] D. Attali, M. Glisse, S. Hornus, F. Lazarus, and D. Morozov. Persistence-sensitive simplification of functions on surfaces in linear time. In *TopoInVis Workshop*, 2009.

[3] U. Bauer, C. Lange, and M. Wardetzky. Optimal topological simplification of discrete functions on surfaces. *Discrete and Computational Geometry*, pages 347–377, 2012.

[4] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Trans. on Vis. and Comp. Graph.*, 10:385–396, 2004.

[5] H. Carr, J. Snoeyink, and A. Ulrike. Computing contour trees in all dimensions. In *Proc. of Symposium on Discrete Algorithms*, pages 918–926, 2000.

[6] H. Carr, J. Snoeyink, and M. van de Panne. Simplifying flexible isosurfaces using local geometric measures. In *Proc. of IEEE VIS*, pages 497–504, 2004.

[7] K. Cole-McLaughlin, H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Loops in Reeb graphs of 2-manifolds. In *ACM Symp. on Comp. Geom.*, pages 344–350, 2003.

[8] C. Correa, P. Lindstrom, and P. Bremer. Topological splines: A structure-preserving visual representation of scalar fields. *IEEE Trans. on Vis. and Comp. Graph. (Proc. of IEEE VIS)*, 17:1842–1851, 2011.

[9] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete & Computational Geometry*, 28:511–533, 2002.

[10] H. Edelsbrunner, D. Morozov, and V. Pascucci. Persistence-sensitive simplification of functions on 2-manifolds. In *ACM Symp. on Comp. Geom.*, pages 127–134, 2006.

[11] R. Forman. A user's guide to discrete Morse theory. *Advances in Mathematics*, 134:90–145, 1998.

[12] Y. Gingold and D. Zorin. Controlled-topology filtering. *Computer-Aided Design*, pages 676–684, 2006.

[13] A. Gyulassy, P.-T. Bremer, B. Hamann, and P. Pascucci. A practical approach to Morse-Smale complex computation: scalabity and generality. *IEEE Trans. on Vis. and Comp. Graph. (Proc. of IEEE VIS)*, pages 1619–1626, 2008.

[14] J. Milnor. *Morse Theory*. Princeton University Press, 1963.

[15] X. Ni, M. Garland, and J. Hart. Fair Morse functions for extracting the topological structure of a surface mesh. *ACM Trans. on Graph. (Proc. of ACM SIGGRAPH)*, 23:613–622, 2004.

[16] V. Pascucci, G. Scorzelli, P. T. Bremer, and A. Mascarenhas. Robust online computation of Reeb graphs: simplicity and speed. *ACM Trans. on Graph. (Proc. of ACM SIGGRAPH)*, 26:58.1–58.9, 2007.

[17] G. Patanè and B. Falcidieno. Computing smooth approximations of scalar functions with constraints. *Computers and Graphics*, 33:399–413, 2009.

[18] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Exp. Math.*, pages 15–36, 1993.

[19] P. Soille. Optimal removal of spurious pits in digital elevation models. *Water Resources Research*, 40, 2004.

[20] J. Tierny, A. Gyulassy, E. Simon, and V. Pascucci. Loop surgery for volumetric meshes: Reeb graphs reduced to contour trees. *IEEE Trans. on Vis. and Comp. Graph. (Proc. of IEEE VIS)*, 15:1177–1184, 2009.

[21] M. Wardetzky, S. Mathur, F. Kalberer, and E. Grinspun. Discrete Laplace operators: no freen lunch. In *Proc. of SGP*, pages 33–37, 2007.

[22] T. Weinkauf, Y. Gingold, and O. Sorkine. Topology-based smoothing of 2D scalar fields with C^1-continuity. *Comp. Graph. Forum (Proc. of EuroVis)*, 29:1221–1230, 2010.

[23] K. Xu, H. Zhang, D. Cohen-Or, and Y. Xiong. Dynamic harmonic fields for surface processing. *Comput. Graph. (Proc. of SMI)*, 33:391–398, 2009.