Combinatorial Exploration of Morse–Smale Functions on the Sphere via Interactive Visualization

Youjia Zhou, Jānis Lazovskis, Michael J. Catanzaro, Matthew Zabka, and Bei Wang

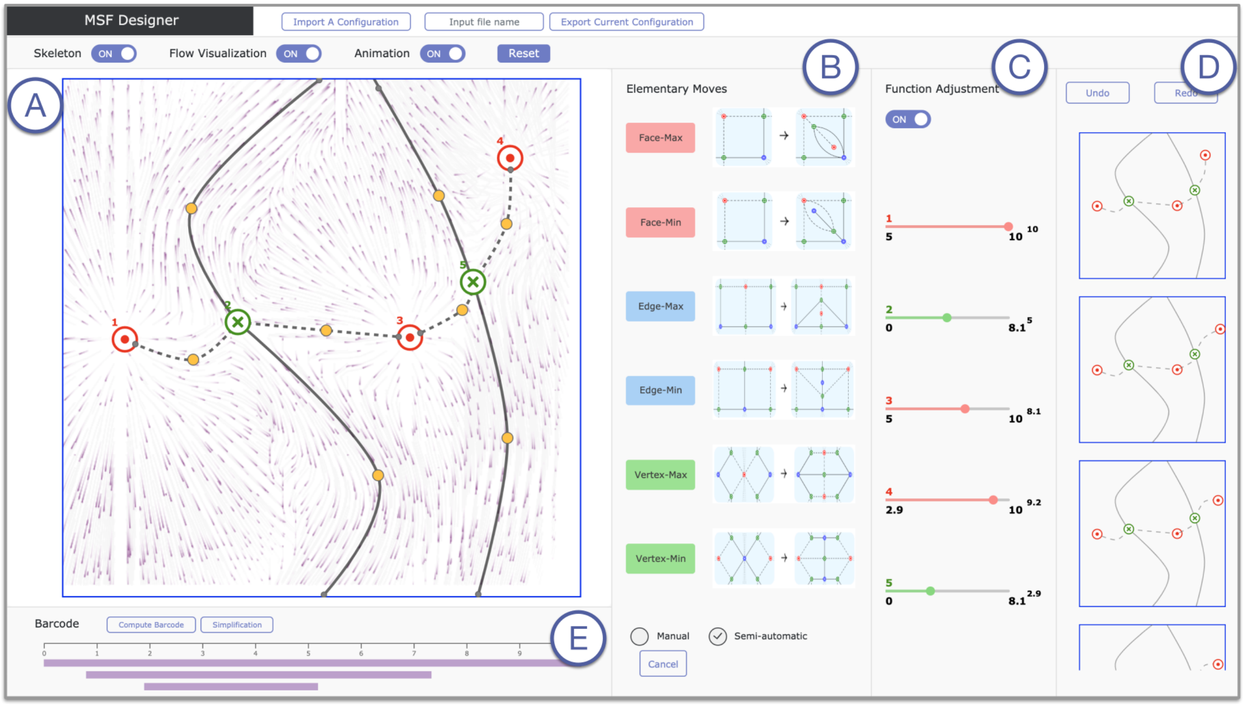

Fig. 1. With MSF Designer, users can interact with a Morse–Smale function f on the sphere visually and manipulate it structurally. The interface consists of several panels. (A) **Function and flow visualization** panel supports modifying the topology and geometry of the Morse–Smale graph of f and visualizes the dynamics of its underlying gradient vector field ∇f via animation. (B) **Elementary moves** panel provides a set of elementary moves as fundamental building blocks of a Morse–Smale function. (C) **Function adjustment** panel allows modifying the function values at singularities. (D) **History** panel provides undo and redo features to remove or repeat single or multiple operations. (E) **Barcode** panel computes and displays persistent homology barcodes to guide persistence simplification.

Abstract— In this paper, we are interested in the characterization and classification of Morse–Smale functions. To that end, we present MSF Designer, an interactive visualization tool that supports the combinatorial exploration of Morse–Smale functions on the sphere. Our tool supports the design and visualization of a Morse–Smale function in a simple way using fundamental moves, which are combinatorial operations introduced by Catanzaro et al. that modify the Morse–Smale graph of the function. It also provides fine-grained control over the geometry and topology of its gradient vector field. The tool is designed to help mathematicians explore the complex configuration spaces of Morse–Smale functions, as well as their associated gradient vector fields and Morse–Smale complexes. Understanding these spaces will help mathematicians expand their applicability in topological data analysis and visualization. In particular, our tool helps topologists, geometers, and combinatorialists explore invariants in the classification of vector fields and characterize Morse functions in the persistent homology setting.

Index Terms—Morse–Smale functions, Morse functions, Morse–Smale complexes, Morse–Smale Vector fields, persistence

1 Introduction

Many popular tools from topology in visualization—merge trees, contour trees, Reeb graphs, Morse and Morse–Smale complexes—originate from the Morse theory. In this paper, we help mathematicians explore the interplay among Morse functions, their gradient vector fields, and Morse–Smale complexes via interactive visualization.

Morse theory studies the relation between the shape of a space and functions on the space. According to Matsumoto, it describes "how the critical points of a function defined on a space affect the topological shape of the space, and conversely, how the shape of a space controls the distribution of the critical points of a function" [20]. A smooth function defined on a manifold is a Morse function if all its critical points are nondegenerate. Morse theory associates the topological changes

- *Y. Zhou and B. Wang are with the University of Utah. E-mails: zhouyj96180@gmail.com, beiwang@sci.utah.edu*
- *J. Lazovskis is with the Riga Technical University. E-mail: janis.lazovskis_1@rtu.lv*
- *M. J. Catanzaro is with Geometric Data Analytics. E-mail: mikecat5@gmail.com*
- *M. Zabka is with Southwest Minnesota State University. E-mail: Matthew.Zabka@smsu.edu*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

of a manifold with the critical points of a Morse function defined on the manifold. The gradient of a Morse function with respect to some Riemannian metric on the manifold is a vector field. The Morse–Smale complex [6, 7] of a Morse function captures the characteristics of this vector field by decomposing the manifold into cells of uniform flow behavior. The Morse–Smale complex is a type of topological descriptor that provides a compact and abstract representation of scalar functions, which has been shown to be effective for identifying, ordering, and selectively simplifying features of data across a wide range of applications ranging from cosmology simulations [15] to material science [13, 14]; see [46] for a survey.

Given the connections among Morse functions, their gradient vector fields, and Morse–Smale complexes, we introduce a design tool that enables mathematicians to perform fine-grained analysis and exploration of these objects interactively.

Contributions. We present MSF Designer, a design tool that helps mathematicians characterize and classify the spaces of Morse–Smale functions (i.e., a more restricted version of a Morse function) on the sphere, see Fig. 1 for its visual interface. Specifically,

- MSF Designer helps users design a Morse–Smale function on a sphere using fundamental moves of Catanzaro et al. [3], thus exploring the space of these functions combinatorially and visually.
- MSF Designer provides fine-grained control over the topology and geometry of a Morse–Smale function, and supports the exploration of its associated gradient vector field and Morse–Smale complex.
- MSF Designer computes and visualizes the persistent homology barcode [11] of a Morse–Smale function to offer a global summary of its features. It also allows the adjustment of function values at critical points to explore diverse functions. Furthermore, it creates a one-to-one mapping between the topological features in the domain with bars in the barcode to guide interactive persistence simplification.
- MSF Designer offers a set of applications in mathematics, specifically, it helps mathematicians:
  - Construct and explore topological invariants in the classification of Morse–Smale flows on surfaces;
  - Characterize various equivalence classes of Morse functions through persistence;
  - Study the inverse problem of generating Morse functions (and their gradients) from a given barcode;
  - Explore the combinatorics of gradient vector fields;
  - Create collections of Morse–Smale functions/complexes for ensemble analysis, uncertainty quantification, and visualization.

2 Technical Background

We review relevant notions on Morse and Morse–Smale functions, Morse–Smale complexes, vector field topology, and persistence.

Terminology. Some of the terminology used is unintuitive, so we prepare the reader with a brief overview in Figure 2, ahead of detailed definitions further in this section.

A *smooth function* is	if
Morse [22]	all its critical points are nondegenerate
Morse–Smale [5]	it is Morse **and** its (un)stable manifolds intersect transversally

A *vector field* is	if
Morse–Smale [27]	its trajectories satisfy three conditions
Morse [27]	it is Morse–Smale **and** it has no periodic trajectories

Fig. 2. An overview of some key terms. Our work is mostly concerned with Morse–Smale functions and Morse vector fields.

Moreover, some authors consider "Morse vector fields" as gradient vector fields of Morse functions [4], whereas others consider "Morse–Smale vector fields" as gradient vector fields of Morse–Smale functions [3]. We avoid this implied link and use precise definitions, which we now introduce.

Morse functions and Morse–Smale functions. Given a smooth function $f : \mathbb{M} \to \mathbb{R}$ defined on a manifold $\mathbb{M}$, a point $x \in \mathbb{M}$ is a *critical* point of f if and only if the partial derivatives at x are zero; otherwise, it is *regular*. A critical point x is *nondegenerate* if the Hessian (the matrix of second derivatives) at x is nonsingular. The function f is a *Morse function* if all its critical points are nondegenerate. Morse functions are considered to be a well-behaved functions, since their critical points are isolated and stable with respect to small perturbations. They play an essential role in helping mathematicians understand the topology of a manifold.

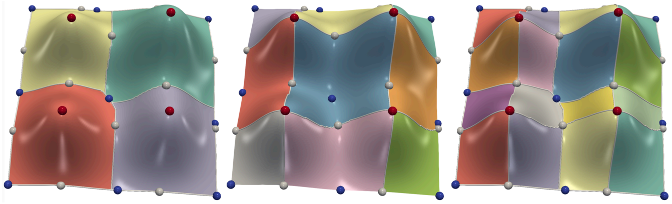

Fig. 3. A Morse–Smale function f on a 2-dimensional manifold. The stable manifolds of f surrounding local maxima (left) and unstable manifolds surrounding local minima (middle) intersect transversally. These intersections produce a Morse–Smale complex (right). Local maxima, local minima, and saddles of f are in red, blue, and white, respectively.

Let f be a Morse function on $\mathbb{M}$ and ∇f be its gradient. An *integral line* at a regular point is a maximal path in $\mathbb{M}$ whose tangent vectors agree with the gradient. The *stable manifold* $S(x) \subseteq \mathbb{M}$ of a critical point x of f is the point itself together with all regular points whose integral lines end at x. The *unstable manifold* $U(x) \subseteq \mathbb{M}$ of x is the point itself together with all regular points whose integral lines originate at x [5, page 131]. A *Morse–Smale function* is a Morse function whose stable and unstable manifolds intersect transversally [5, page 132]; see Fig. 3 for an example.

Morse–Smale complexes. For a given Morse–Smale function f, by intersecting the stable and unstable manifolds, we obtain the *Morse–Smale cells* as the connected components of the set $U(x) \cap S(y)$ for all critical points $x, y \in \mathbb{M}$ [7]. The *Morse–Smale complex* (MSC) is the collection of Morse–Smale cells [7], shown in Fig. 3 (right).

We define the *Morse–Smale graph* of f to be the 1-skeleton of the MSC, that is, the union of the 0-dimensional (vertices) and 1-dimensional (edges) cells. The Morse–Smale graph of f is equivalent to the topological skeleton [16] of the gradient vector field of f defined below.

Vector field topology. A two-dimensional vector field v is a smooth mapping $v \colon \mathbb{M} \to \mathbb{R}^2$ defined on a manifold $\mathbb{M}$. *Critical points* (or *singularities*) are elements of $\mathbb{M}$ at which the vector field values are zero; they include sources, saddles, and sinks. A *streamline* is a line that is tangential to the instantaneous velocity direction. A *topological skeleton* [16] of a vector field consists of critical points and *separatrices* (i.e., streamlines connecting the critical points), which decomposes the domain into different modes of flow behavior.

Morse and Morse–Smale vector fields. Let $\mathrm{T}\mathbb{M}_p$ denote the tangent space of $\mathbb{M}$ at p. A vector field v on $\mathbb{M}$ associates a vector $v(p) \in \mathrm{T}\mathbb{M}_p$ to each point $p \in \mathbb{M}$. An *integral curve* of v through a point $p \in \mathbb{M}$ is a smooth map $\gamma \colon I \to \mathbb{M}$ such that $\gamma(0) = p$ and $\gamma(t) = v(\gamma(t))$ for all $t \in I = [0, 1]$. The image of an integral curve is called a *trajectory* [28, page 10]. Two vector fields $v_1 : \mathbb{M}_1 \to \mathbb{R}^2$ and $v_2 \colon \mathbb{M}_2 \to \mathbb{R}^2$ are considered *topologically trajectory equivalent* if there is a homeomorphism $h \colon \mathbb{M}_1 \to \mathbb{M}_2$ that transforms the trajectories of v_1 into the trajectories of v_2, preserving the orientations of the trajectories [27, Definition 1.1]. A vector field v is *structurally stable* if the topological behavior of its trajectories is preserved under small perturbations of v [27, Definition 1.2], that is, if the perturbed and the initial vector fields are trajectory topologically equivalent.

Suppose $\mathbb{M}$ is compact. Then there exists a *global flow* $\phi \colon \mathbb{R} \times \mathbb{M} \to \mathbb{M}$ determined by v such that $\phi(0, p) = p$ and $(\partial/\partial t)\phi(t, p) = v(\phi(t, p))$ [28, Proposition 1.3]. For each $t \in \mathbb{R}$, the map $v_t : \mathbb{M} \to \mathbb{M}$ is defined as $v_t(p) = \phi(t, p)$. The *ω-limit set* of a point $p \in \mathbb{M}$ is $\omega(p) = \{q \in \mathbb{M} \mid v_{t_n}(p) \to q \text{ for some sequence } t_n \to \infty\}$. The α-

limit set of p is $\alpha(p) = \{q \in \mathbb{M} \mid X_{t_n} \to q \text{ for some sequence } t_n \to -\infty\}$.

A vector field v on a closed two-dimensional surface is called a *Morse–Smale vector field* [27, Definition 1.4] if

- v has finitely many singular points and periodic trajectories, which are all hyperbolic;
- There are no trajectories from a saddle to a saddle;
- The α-limit set and the ω-limit set of each trajectory of v is either a singular point or a periodic trajectory (a limit cycle).

A vector field is a *Morse vector field* if it is a Morse–Smale vector field without periodic trajectories.

Gradient vector fields of Morse–Smale functions. In this paper, we focus on the gradient vector fields of Morse–Smale functions f on the sphere $\mathbb{M} = \mathbb{S}^2$, denoted as ∇f. These gradient vector fields are by definition Morse vector fields. Although this formulation might seem restrictive, the study of Morse–Smale functions on the sphere under the persistence setting is nontrivial and under active investigation [3, 9, 25].

Fig. 4 (left) illustrates the topological skeleton of a gradient vector field ∇f on the sphere, where the blue boundary indicates a (global) sink. Here, and further in this paper, we follow [7] in using ⊙ to indicate a source, ◎ for a sink, and ⊕ for a saddle. Separatrices are either dashed lines (saddle-source connections) or solid lines (saddle-sink connections). ⊙, ◎, and ⊕ also correspond to the local maxima, local minima, and saddles of a Morse–Smale function f.

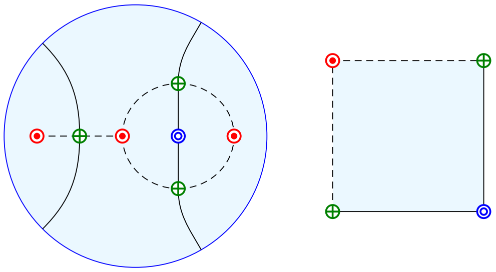

Fig. 4. Left: a topological skeleton of a gradient vector field ∇f on the sphere with boundary identified to a point. Equivalently, this is a Morse–Smale graph of f. Right: a quadrangle (or cell) of ∇f.

Finally, by [7, Quadrangle Lemma], every Morse vector field can be decomposed into regions, referred to as *quadrangles* or *cells*, as illustrated in Fig. 4 (right). In a nongeneric or boundary setting, a quadrangle may become degenerate, that is, a separatrix may have the same quadrangle on both of its sides. Equivalently, every cell of a Morse–Smale graph has four edges, counting an edge twice if the face is on both sides of the edge [3].

Persistence and persistence simplification. Persistent homology is a powerful tool in topological data analysis that is applicable in both data summarization and simplification. In its standard setting, persistent homology can be considered as an extension of Morse theory, in the sense that it studies homology groups of the sublevel sets connected by inclusions, $\mathbb{M}_a \hookrightarrow \mathbb{M}_b$ for $a \leq b$, where $\mathbb{M}_a := f^{-1}(-\infty, a]$. In other words, it computes and summarizes topological features of a space across multiple scales. The importance of a feature can be quantified via the notion of *persistence*, that is, the amount of change to f necessary to eliminate the feature [9]. Persistence is also useful in simplifying a function f in terms of removing topological noise as determined by its persistence diagram or barcode [8, 9].

We consider only Morse–Smale functions f on the sphere and their gradient vector fields ∇f (refer to as vector fields) throughout the remainder of this paper. Given such a Morse–Smale function, we can construct a hierarchy by successive simplification based on persistence. Each step in the process cancels a pair of adjacent critical points and the sequence of cancellations is determined by the persistence of the pairs. A pair of critical points of f are adjacent if they are connected by an edge in the Morse–Smale graph of f. Equivalently, a pair of critical points of ∇f are adjacent if they are connected by a separatrix.

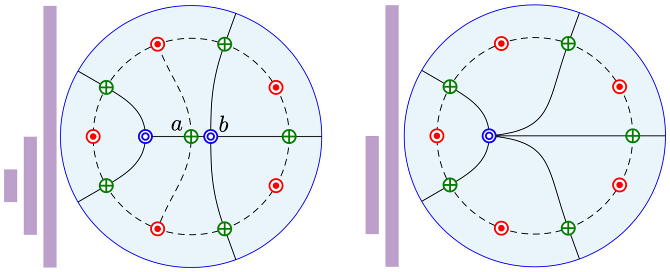

Fig. 5. The topological skeleton and persistence barcodes before (left) and after the cancellation (simplification) of critical points a and b. Barcodes before and after simplifications are shown vertically in purple. This figure is an adaptation of [7, Figure 17].

We compute the pairing of critical points and their persistence using algorithms in [7]. Heuristically, saddles of a Morse–Smale function f cancel with local maxima or minimum; correspondingly, saddles of ∇f cancel with sources or sinks. The sequence of cancellations is in the order of increasing persistence, and the process can be simulated combinatorially. Fig. 5 illustrates the cancellation of a saddle-min pair of f (equivalently, a sink-saddle of ∇f). Note that for every positive i, the i-th pair of critical points ordered by persistence forms an arc in the topological skeleton obtained by canceling the first $i - 1$ pairs [7, Adjacency Lemma]. This means that, in general, not all critical points paired by the persistence algorithm are adjacent; however, they will become adjacent after enough cancellations of other pairs [7]. Persistence simplification provides an important functionality in our interactive system.

3 Related Work

Vector field design systems. A few vector field design systems have been created for domain-specific applications, such as graphics [17, 18, 31, 39, 43] and fluid simulation [35]. The techniques can be roughly classified into projection, diffusion, and interpolation. In the first approach, a three-dimensional vector field is specified and projected onto the surface [41]. This approach is fast and simple, but it is difficult to achieve fine-grained control. In the second approach, the system performs relaxation based on a set of user-specified vectors, where known vector values are propagated like diffusion across the remaining surface [39, 43]. In the third approach, a global vector field is built by interpolating a few user-specified vector values using basis functions [31, 40].

To provide more control over vector field topology, other design systems focus on specifying the number, types, and locations of singularities or to a larger extent, the topological skeleton. A simple way to design and control a vector field begins with a set of *flow primitives* or *building blocks* [44], and such primitives are combined into a global vector field. For instance, a primitive can be a simple vector field in the local neighborhood of a user-specified singularity, and multiple primitives can be combined using radial basis functions [40]. Furthermore, singularities can be added, removed, or edited by users [32]. The users can also specify the entire topological skeleton of the desired vector field [36], although a complete specification may be inefficient.

In terms of vector field simplification, techniques are often based on Laplacian smoothing [30, 37, 45], whereas topology-based techniques originate from the study of Morse theory and gradient vector fields of Morse functions. By specifying the number and configuration of the critical points of a Morse function and running multi-grid relaxation, the design of Morse functions over a surface is equivalent to the design of their gradient vector fields [24]. The gradient vector field of a scalar function can be simplified by modifying function values near the singularity pair [7, 8]. A singularity pair can also be simplified directly by performing nonlinear optimization surrounding the pair [38] or using Conley index theory [47].

Our system is topology-based and shares some similarities with the above systems, with three main distinctions:

1. Elementary moves (Sect. 4) are used as fundamental building blocks in designing the Morse–Smale function and its gradient vector fields incrementally.
2. Much finer control is given to the vector field topology (respectively, Morse–Smale graph), in particular, adjusting the geometry of separatrices (respectively, edges in a Morse–Smale graph).
3. Persistence barcodes are used explicitly to guide interactive simplification and exploration.

Topological classification of vector fields. Our work is related to the topological classification of Morse and Morse–Smale vector fields with one crucial difference: we are interested in the equivalence classes of Morse functions (and their gradient vector fields) that share the same persistence barcode. That is, their Morse–Smale complexes are topologically indistinguishable, and we work to distinguish them.

One of the first invariants defined for Morse–Smale vector fields on a closed surface is Peixoto's *distinguished graph*, that is, a graph together with a distinguished set of edges satisfying some conditions [29]. The distinguished graph provides an invariant for Morse–Smale vector fields on a surface, but has a very technical description. The graph corresponds to connected components of what is left after deleting saddle singularities and their associated stable and unstable manifolds from the surface. Several simpler invariants have been proposed based on Peixoto's work. *Cyclic distributions of colored points* introduced by Fleitas are a simplification of the distinguished graph invariant [10], and the *colored dual graph* by Wang is a variant for orientable closed surfaces [42]. All three mentioned above are invariants of Morse–Smale vector fields on closed surfaces, and they become complete invariants when restricted to Morse vector fields (see [27] for an extended comparison of these invariants).

Related to classification of gradient-like vector fields on a surface is the classification of functions themselves. An a-*function* is a Morse function on a surface with exactly three critical values. The f-*invariant* was constructed to classify a-functions up to conjugacy [26], and hence their vector fields up to topological equivalence [26, Remark 2.8]. In addition, there has been work classifying Morse functions on surfaces, albeit from a much different perspective [2, 25, 33].

Some of the mentioned invariants could also be used in the design of vector fields, left open as an avenue for future research; however, they do not lead to as simple and efficient operations as the elementary moves (Sect. 4) employed in this paper.

4 Methods

To design a Morse–Smale function on a sphere, we describe the main analytic components within MSF Designer, including elementary moves as fundamental building blocks, vector field construction with basis functions, and geometric control of Morse–Smale graphs using splines.

4.1 Elementary Moves

Since the Euler characteristic of the sphere is two, the simplest function f on a sphere (that is not a constant function) is one with a single maximum and a single minimum. We visualize the minima as a boundary cycle in blue: imagine "flattening" the sphere onto a disk by expanding a rubber band surrounding the minimum; see Fig. 6 (left). Equivalently, the gradient vector field ∇f (and the simplest Morse vector field on a sphere) is one with a single source and a single sink.

However, neither f nor ∇f is Morse–Smale. A Morse–Smale function on the sphere must contain at least one saddle [7, Quadrangle Lemma], which must have four edges coming out of it that cannot be identified with each other. Such a configuration is realizable on the sphere containing two local maximum, one local minimum, and one saddle, shown in Fig. 6 (right), which we use as the default configuration for MSF Designer.

Using MSF Designer, the design of a Morse–Smale function and its gradient vector field could be done interchangeably, as the Morse–Smale graph of f is the topological skeleton of ∇f. Therefore, we first describe the design of a Morse-Smale graph via elementary moves.

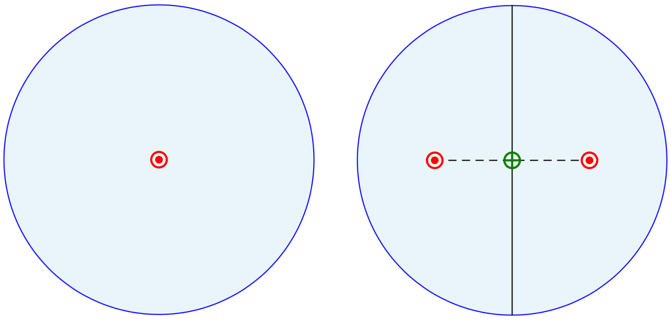

Fig. 6. The simplest function that is not a constant on a sphere (left) and the simplest Morse–Smale function on a sphere (right).

Our visualization framework is based upon understanding how cells, generically as quadrangles, in Fig. 7 (top left), can fit together on a surface and how they change when a pair of singularities is added to their interiors and boundaries; such operations are referred to as *elementary moves*. An elementary move is an action that initiates or advances our design process. Elementary moves originate from a mathematical framework of Catanzaro et al. [3] that studies different notions of equivalence for Morse functions on the sphere in the context of persistent homology. The following theorem [3, Theorem 3] describes all possible ways to create a new Morse–Smale graph via elementary moves. These moves are illustrated in Fig. 7.

Theorem 1 (Elementary Move Theorem)**.** *The Morse–Smale graph of any two Morse–Smale functions is related by a sequence of elementary moves [3, Theorem 3].*

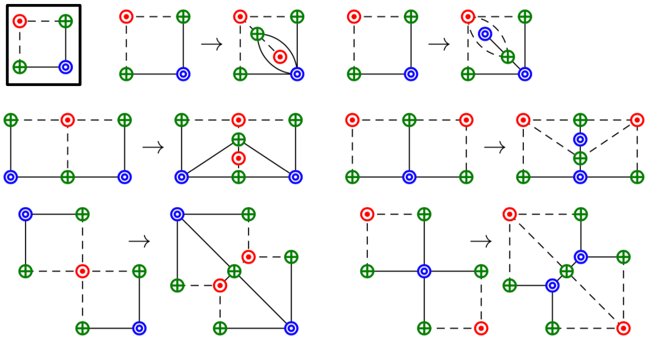

Fig. 7. A single cell (top left in black square) and elementary moves. The elementary moves (left to right) comprise face-max and face-min moves (top row), edge-max, and edge-min moves (middle row), and vertex-max and vertex-min moves (bottom row).

By the Quadrangle Lemma [7], every cell of a Morse–Smale graph is bounded by four edges (counting an edge twice if the cell is on both sides of the edge). This lemma allows us to describe changes to the Morse–Smale graph as a composition of moves. A *face move* adds one saddle-maximum or saddle-minimum pair in the interior of a cell. An *edge move* adds one saddle-maximum or saddle-minimum pair on the edge of a cell. A *vertex move* adds one saddle-maximum or saddle-minimum pair at an existing singularity. All moves add two cells to the Morse–Smale graph. Everything outside the modified region stays the same between moves. Notice that all moves are, in fact, reversible; that is, the addition of a pair of singularities translates to the removal of a pair of singularities in the reverse direction. The face, edge, and vertex moves are ways of manipulating a Morse–Smale graph to obtain a different Morse–Smale graph. These moves by themselves do not have functional values associated with the singularities.

However, if we combine the modification of the Morse–Smale graph with the modification of function values at the singularities (see Sect. 5), we could now construct and explore the space of Morse–Smale functions. We could equivalently construct and explore its gradient vector field.

4.2 Initialization, Control, Debugging, and Simplification

Initial vector field design. Once the user specifies the types and locations of singularities via the elementary moves, we consider these singularities as sources, sinks, and saddles of the vector field and we use the framework of Zhang et al. [47] to construct an initial vector field. The vector field is represented as a triangulated mesh with vector values assigned at the vertices of the mesh. We attach a *basis vector field* to each user-specified singularity, and construct a vector field as the truncated sum of these basis vector fields. For instance, a basis vector field centered at a source $\mathbf{p}_0 = (x_0, y_0)$ is defined as:

$$V(\mathbf{p}) = e^{-d\|\mathbf{p}-\mathbf{p}_0\|^2} \begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix},$$

for any point $\mathbf{p} = (x, y) \in \mathbb{R}^2$, where d is a constant that is used to control the amount of influence of the basis vector field, and the matrix $\begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix}$ indicates the type of the singularity. For sinks and saddles, we replace this matrix with $\begin{pmatrix} -k & 0 \\ 0 & -k \end{pmatrix}$ and $\begin{pmatrix} k & 0 \\ 0 & -k \end{pmatrix}$, respectively.

Geometric control. To provide a geometric control of the separatrices (which correspond to edges of the Morse–Smale graphs), we approximate their geometry using cubic cardinal splines with tension 0. In addition to the initial vector field, we also generate another auxiliary vector field that captures the flow along separatrices. The final vector field is a weighted sum of the initial and the auxiliary vector field, where the weight is computed from the distance between a mesh vertex and its closest separatrix. We apply additional smoothing in the neighborhood of separatrices to prevent the creation of spurious singularities, by replacing the function value of each mesh vertex in the neighborhood of separatrices with the weighted average function value of its neighbors. The weights are inversely proportional to the distances between the vertex and its neighbors.

Debugger. MSF Designer provides great flexibility for a user to control geometric details involving separatrices (equivalently, edges from Morse–Smale graphs). A key component is the *debugger*, which detects invalid configurations. Invalid configurations may arise due to user operations, semiautomatic modes, simplification, or boundary conditions. Flow animation is not allowed when the debugger detects a configuration to be invalid.

In this paper, we assume all saddles are simple and all higher-order saddles can be unfolded into simple saddles. Every saddle, therefore, is of degree four, and the endpoints of its four adjacent separatrices alternate between connections with sources and sinks, as in Fig. 4, for example. Sources and sinks may have arbitrary degrees. A debugger will report an invalid configuration if:

- The separatrices adjacent to a saddle do not follow the appropriate *alternating* order. Recall a saddle-source connection is indicated by a solid line, and a saddle-sink connection is marked by a dashed line. The configuration of a saddle is invalid if its adjacent separatrices are not in alternating solid and dashed lines.
- The end point of a separatrix is not properly attached to a singularity or the boundary (the global sink).
- The separatrices are crossing.
- A singularity is isolated without any separatrix attachment, with the exception of the minimal configuration in Fig. 6 (left).
- There is a saddle-saddle separatrix.
- A singularity is dragged outside the boundary.

See Sect. 5 for examples of invalid configurations detected by the debugger.

Persistence simplification. We use Perseus [23] for efficient computation of persistent homology, which gives rise to persistence pairings of singularities. Since some of these pairs are in fact adjacent, based on [7, Adjacency Lemma] (Sect. 2), they can be simplified by modifying the basis functions defining these singularities. Each simplification operation removes a pair of adjacent singularities ranked by persistence, together with edges adjacent to the pair, resulting in a simplified topological skeleton. See Fig. 5 for an example of simplifying a saddle-minimum (equivalently saddle-sink) pair (a, b) connected by an edge.

4.3 System Design

MSF Designer is web-based and can be accessed from any modern web-browser (tested using Google Chrome and Mozilla Firefox). MSF Designer is implemented in HTML, CSS, and JavaScript, with Python and Flask as the backend server to handle requests from the browser. The module collection `D3.js` is used for rendering SVGs, and the flow animation is generated with a Canvas element. Perseus [23] is used to compute persistence homology and produce barcode. MSF Designer is provided open-sourced via GitHub[1]. We also provide a supplementary video that demonstrates the capabilities of MSF Designer.

5 MSF Designer User Interface

The user interface of our design system is provided in Fig. 1; see the supplementary video for a demo. The system contains five main interactive panels: the function and flow visualization panel, the elementary moves panel, the function adjustment panel, the history panel, and the barcode panel.

Function and flow visualization panel (A) supports modifying the topology and geometry of the Morse–Smale graph of f (resp.the topological skeleton of ∇f). In particular, as illustrated in Fig. 8, users can modify the locations of singularities using "drag and drop". Since edges of Morse–Smale graphs (resp., separatrices) are modeled as splines, users can also modify their geometry using yellow control points of the splines.

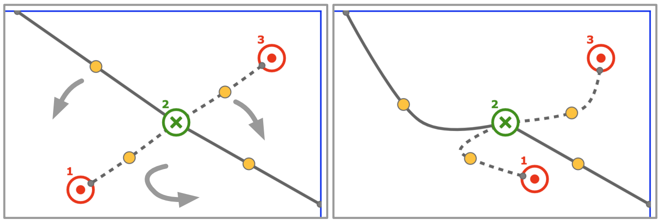

Fig. 8. Control points (in yellow) are used to modify (left to right) the geometry of edges surrounding a saddle point.

The panel supports both static and dynamic flow visualization. The topological skeleton, the (static) flow visualization, and the animation of the current configuration can be enabled or disabled as desired. The animation is particularly useful for the user to get an intuitive sense of the dynamics of ∇f, as in Fig. 1 (A).

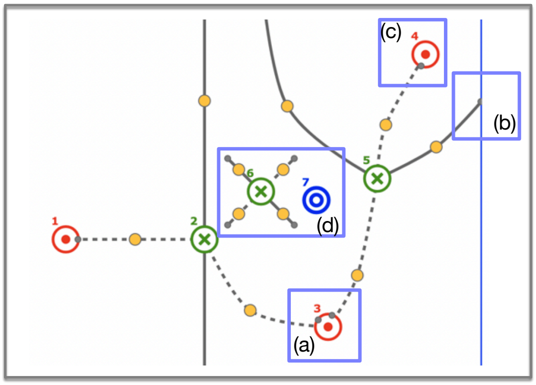

Fig. 9. Various ways of connecting pairs of singularities. (a) A source is attached to two saddles, and the attachment map is defined by gluing the end points (in gray) of saddle-source connections. (b) A saddle is attached to the boundary (the global sink). (c) A source is attached to a saddle via a single attachment point. (d) This is an initialization of a face-min move under manual mode.

[1] https://github.com/tdavislab/MSF-Designer

Elementary moves panel. Various elementary moves are provided via the elementary moves panel of Fig. 1 (B). Under *manual mode*, a user connects pairs of singularities manually and our system checks for valid configurations. Using *semiautomatic mode*, edges of Morse–Smale graphs are added automatically, followed by user adjustments. Fig. 9 shows various ways of connecting pairs of singularities. MSF Designer provides fine-grained control in attaching edges to singularities. Specifically, the end points (in gray) of each edge can be attached to the circular boundaries of the glyphs representing different types of singularities.

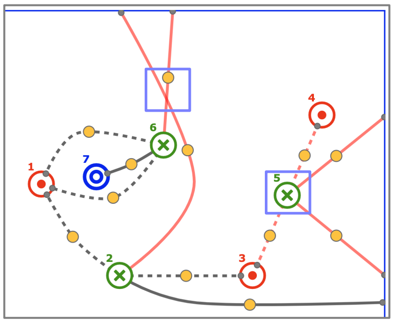

Fig. 10. Examples of invalid configuration, where invalid seperatrices are highlighted in red.

For both manual and semiautomatic modes, a key component connecting the function and flow visualization panel with the elementary moves panel is the *debugger* (Sect. 4). It detects invalid configurations when a user is modifying the geometry of the topological skeleton during the design process, or when the semiautomatic configuration of an elementary move does not resolve all errors. When an invalid configuration is detected, edges involved in the configuration are highlighted in red. Additional warning messages are provided to the user to guide necessary correction and adjustment operations; see Fig. 10 for an example, where blue boxes enclose regions of error either the edges are intersecting, or the edges surrounding a saddle are not in alternating order.

Function adjustment panel and history panel. Recall that the design of a Morse–Smale function is equivalent to the design of its gradient vector field, and a gradient vector field can be modified by modifying function values at the singularities. The function adjustment panel in Fig. 1 (C) enables a user to modify the function values at singularities. Such modification does not necessarily change the flow directions, but may change the flow magnitude. MSF Designer automatically checks for constraints in terms of function values during the adjustment operation, that is, it ensures the function value of saddle is bounded above by function values of its adjacent local maxima, and bounded below by its adjacent local minima. The history panel in Fig. 1 (D) stores all valid and invalid operations during manipulations and supports redo and undo operations.

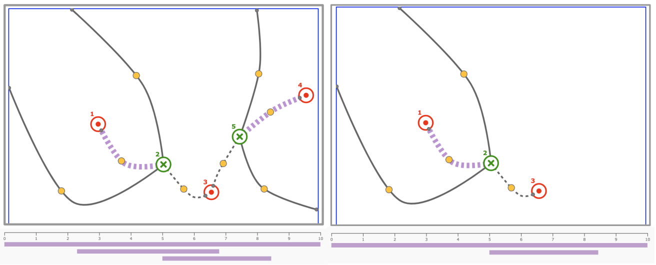

Fig. 11. All simplification candidate pairs are connected by purple dotted lines (left), any of which may be chosen for simplification (right).

Barcode panel. Finally, using the barcode panel in Fig. 1 (E), we provide the persistence barcode of the current configuration, as well as interactive capabilities for persistence-based simplification. As illustrated by a simple example in Fig. 11, when the simplification button is clicked (left), the funding and flow visualization panel highlights adjacent pairs of singularities that are eligible for simplification, marked by purple dotted lines. Selecting a particular bar leads to the highlighting of a potential candidate pair, if one exists, and clicking on an eligible pair will simplifying the underlying field (right). The remaining configuration contains a last candidate pair for simplification, and simplifying it will result in the default configuration of Fig. 6 (left).

6 Usage Scenarios

We describe various usage scenarios to illustrate the capabilities of MSF Designer for mathematicians. These usage scenarios are created with our collaborating mathematicians. The tool helps topologists, geometers, and combinatorialists explore invariants in the classification of flows and characterize Morse functions in the persistence setting.

6.1 Studying Topological Invariants in Flow Classification

Qualitative analysis of dynamical systems and the classification of such systems is an active area of research, which has widespread applications in mathematics, physics, biology, economics, and medicine. Users of MSF Designer can study various topological invariants employed in the classification of flows on surfaces, as the elementary moves give insight into the construction of certain invariants and highlight the differences between them (Sect. 3).

Three-color graphs. Oshemkov and Sharko [27] studied an invariant of a Morse–Smale flow called a *three-color graph* and proved that such an invariant classifies Morse flows on two-dimensional surfaces up to trajectory topological equivalence. We replicate Example 1.8 of [27] in Fig. 12 since it is simple, yet still shows enough complexity of the general theory.

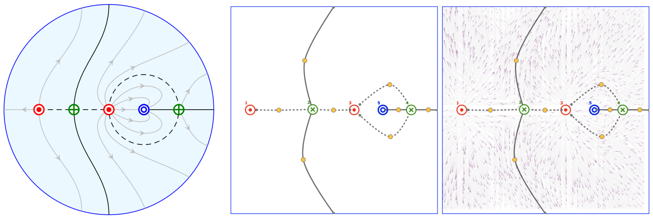

Fig. 12. The design of a vector field on the sphere. An example (left) reproduced from [27, Example 1.8] used to study the three-color graph, a construction (middle) of the same example using MSF Designer, and its flow visualization (right).

Example 1.8 from [27], reproduced in Fig. 12 (left), is a flow defined on a sphere containing two sources, two saddles, and two sinks. One of the sinks is on the reverse side of the sphere, which is represented by the blue boundary circle; separatrices are shown in bold. Here, we demonstrate that MSF Designer can easily reconstruct interesting flows in the literature. In this case, we use a single elementary move and a face-min move on the default configuration, which is a standard height function on the sphere (a pair of sources, a boundary sink, and a saddle), to obtain a vector field with two sources, two sinks, and two saddles, see Fig. 12 (middle). Its gradient vector field is visualized in Fig. 12 (right).

Colored dual graphs. Other authors have constructed different invariants for Morse and Morse–Smale flows on (orientable) surfaces. In addition to the three-color graph, Peixoto [29] introduced a *distinguished graph*, Fleitas [10] introduced *cyclic distributions of coloured points*, and Wang [42] introduced a *coloured dual graph* - all of which may be described and their effects explored using MSF Designer.

We now replicate two examples of colored dual graphs, Example 2.6 and Example 2.14, from [42]. Example 2.6, which is reproduced in Fig. 13 (left), is a flow defined on a sphere containing four sources, three saddles and one sink. The configuration can be generated semiautomatically by two edge-max moves; see Fig. 13 (middle). Its gradient vector field is visualized in Fig. 13 (right). Example 2.14 is the “Monkey seat” containing two sources, two saddles and two sinks Fig. 14 (a). By unfolding the surface from the bottom, which is the sink, we are able to visualize the configuration in a 2D circle; see Fig. 14 (b). The configuration can then be generated using MSF Designer by a single face-min move; see Fig. 14 (c). The flow visualization is shown in Fig. 14 (d).

Distinguished graphs. We replicate an example from [29], which studies two flows of the equivalence classes. Fig. 15 (a) and (d) are reproduced from Figure 1 and 2 in [29], respectively, where vertex 1 stands for the North Pole, vertex 2 for the South Pole, and the dotted line for the equator. Both flows contain seven sources, six saddles and one sink. By unfolding the sphere from the sink, both configurations can be generated using MSF Designer by five face-max moves; see Fig. 15 (b) and (e). The flow visualizations are shown in Fig. 15 (c) and (f), respectively. By looking at the flow visualizations, the mathematicians are able to better compare and distinguish the equivalence classes.

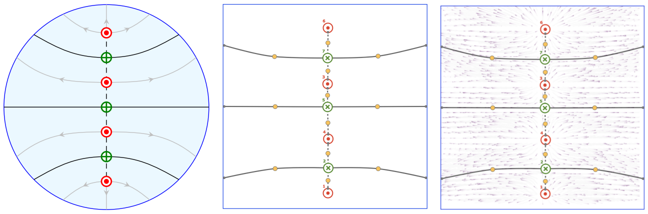

Fig. 13. An example (left) reproduced from [42, Example 2.6] used to study the coloured dual graph, a construction (middle) of the example using MSF Designer, and its flow visualization (right).

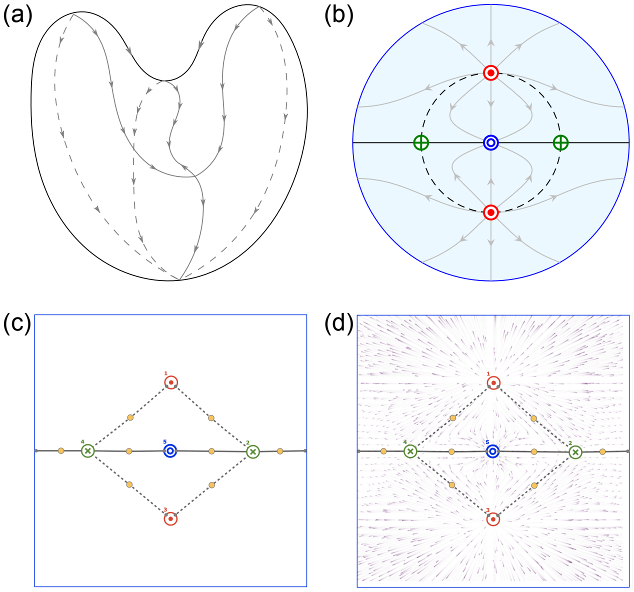

Fig. 14. An example (a) reproduced from [42, Example 2.14] used to study the colored dual graph, a 2D visualization by unfolding the surface from the bottom (b), a construction (c) of the example using MSF Designer, and its flow visualization (d).

6.2 Characterizing Morse Functions Through Persistence

Our original motivation for developing MSF Designer was inspired by a mathematical framework [3] that investigates different moduli spaces of Morse functions from the perspective of persistence. In this vein, we are interested in using MSF Designer to characterize the set of Morse functions that give rise to the same barcode. Two Morse functions may give rise to the same barcode, but they are not necessarily considered equivalent if taking one function to another requires a significant amount of deformation; see Fig. 16 for an example. Recall two Morse–Smale functions f, g are *graph equivalent* if there is a graph isomorphism between their Morse–Smale graphs. Consider the two Morse functions on the sphere in Fig. 16, which have the same barcode. Note that these functions are are not graph equivalent. That is, a deformation from the function given in Fig. 16 (left) to the function given in Fig. 16 (right) requires significant perturbation.

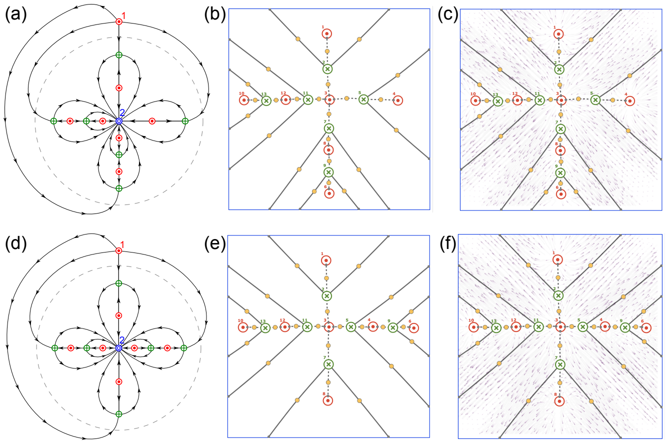

Fig. 15. Two examples (a, d) reproduced from [29, Figure 1, 2] used to study the distinguished graph, the constructions (b, e) of the example using MSF Designer, and their flow visualizations (c, f).

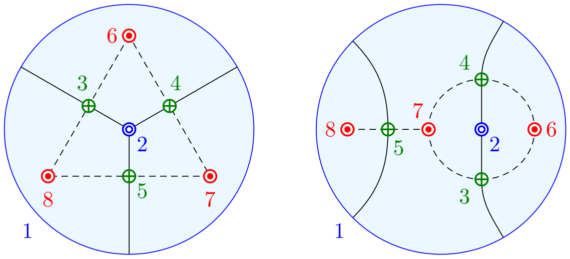

Fig. 16. An example of two Morse functions on the sphere that have the same sub-level set barcode, with critical values indicated. These are *not* graph equivalent functions.

Using MSF Designer, Fig. 17 shows we can easily recreate the two configurations. Starting from the default configuration, the configuration of Fig. 16 (left) can be created under the semiautomatic mode, using a face-max and an edge-min move, followed by geometric modifications to the separatrices. The configuration Fig. 16 (right) can be generated semiautomatically by an edge-min move and an edge-max move, in combination with geometric operations. Notice that the semiautomatic edge-max move creates an invalid configuration (detected by the debugger), which is subsequently corrected.

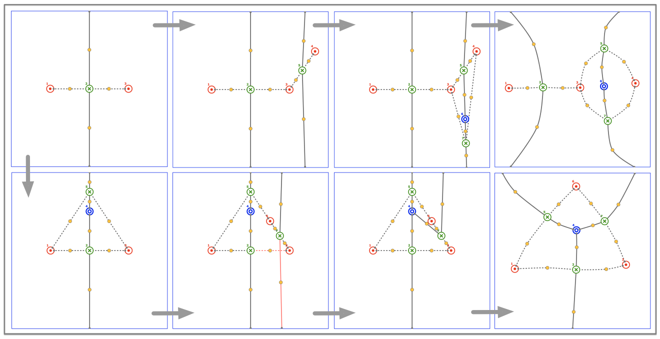

Fig. 17. Using MSF Designer to recreate Morse functions on the sphere that have the same barcode but are not graph equivalent.

Hence, through [3], we can explore the space of Morse functions (that give rise to the same barcode) by putting different equivalence relations on the space. Each choice of equivalence relation leads to a different moduli space structure on the space of Morse functions, and each equivalence class has an interesting combinatorial structure that can be used practically to enrich the barcode. Given MSF Designer, we can start to address the following question: Can we characterize equivalence classes of Morse functions on the sphere that have the same barcode?

6.3 Designing and Visualizing Morse–Smale Complexes

Given the connection between Morse functions, Morse vector fields, and MSCs, MSF Designer can also be utilized to design MSCs. The design process offers insights into structural variations of MSCs in terms of persistence simplification, which is useful in constructing MSC ensembles for uncertainty quantification and uncertainty visualization.

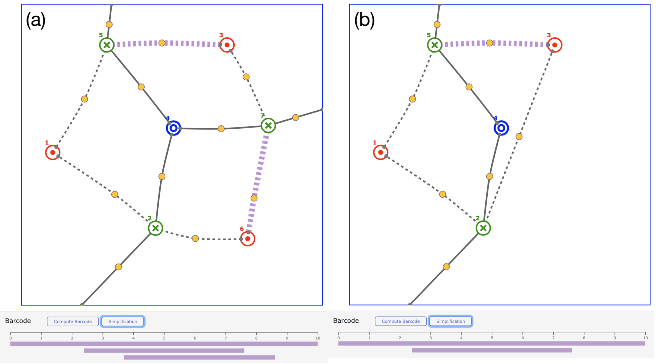

Fig. 18. Using MSF Designer to design Morse–Smale complexes. A Morse–Smale complex is shown together with its barcodes before (a) and after (b) simplification.

An example is given in Fig. 18 where a MSC arises from some underlying function. We illustrate the 1-skeleton of the MSC (equivalently, the topological skeleton of its gradient field) before and after simplification. Notice this approach is different from (and, in some sense, complementary to) the work on conforming MSCs [12], which focuses on constructing MSCs that conform with a user-specified map.

Related to the design of vector fields is the notion of persistence simplification [8]. MSF Designer allows users to view and simplify functions (resp., vector fields) with the simplification feature (highlighted in Sect. 4). This feature connects MSF Designer to persistence simplification, one of the initial motivations for studying persistence.

6.4 Inverse Problem: From Barcodes to Vector Fields

We can also use MSF Designer to study the rich area of inverse problems in topological data analysis. Specifically, given a persistence barcode, can we reverse engineer Morse functions or Morse vector fields that give rise to the given barcode?

Authors in [3, Figure 13] studied an interesting combinatorial question: Suppose a Morse function f has six distinct critical values and a known zero-dimensional barcode consisting of three bars nested inside each other. How many different ways can f be embedded into $\mathbb{R}^3$ while preserving the given barcode? Here, let us ask a simpler, but equally intriguing question: Given a barcode that consists of one, two, or three nested bars as shown in Fig. 19, how many Morse functions (resp., Morse vector fields) on the sphere can we construct that give rise to the same barcode, assuming the largest bar is an extended persistence pair (that represents the connected component of a sphere)?

It turns out that there is only one valid configuration of Morse vector field that gives rise to one, two, or three bars within the above barcodes, respectively. With MSF Designer, we illustrate in Fig. 20 that we can easily construct these configurations using a few elementary moves. From the default configuration in Fig. 20, we can construct configuration Fig. 19 (middle) using an edge-min move, a persistence simplification, and geometric modification. Moving from configuration Fig. 19 (middle) to Fig. 19 (right) involves an edge-min move and geometric modification that do not affect the barcode. The immediate visualization of the barcode and function adjustment options give the user direct control to construct a Morse function with the desired barcode.

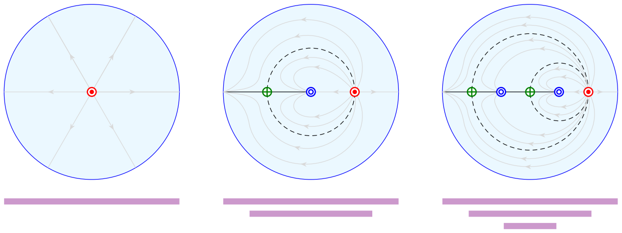

Fig. 19. Morse vector field configurations that give rise to one, two, and three bars in the barcode.

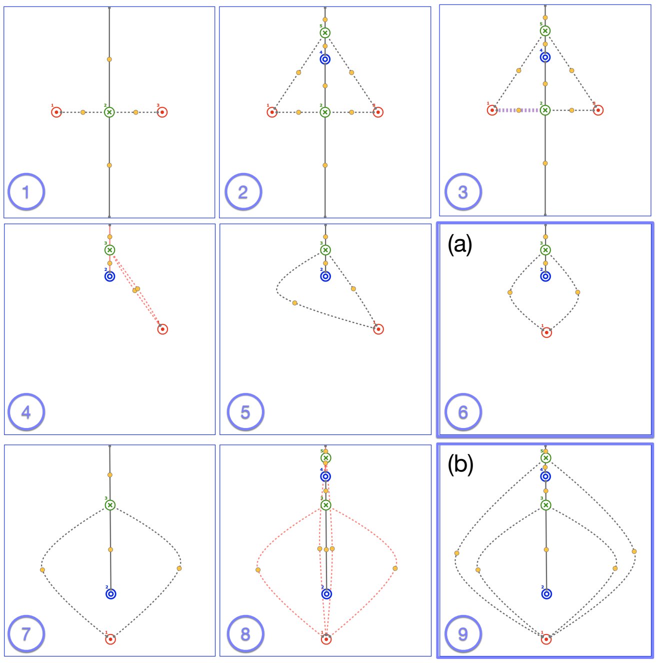

Fig. 20. Two Morse vector field configurations (a, b) that give rise to two or three nested bars in a given barcode, respectively.

6.5 Combinatorics of Vector Fields

Another inverse approach facilitated by MSF Designer is the following: given a fixed number and types of singularities, how many different Morse–Smale vector fields are there with this number and type of critical points? The option to disconnect and reconnect max-saddle and min-saddle edges to and from singularities allows for an interactive search for the desired vector fields.

Figure 21 shows the use of edge connecting and disconnecting, made easier by the "Undo" button, and ensured to be correct, rather than just abstract combinatorial objects, by the debugger. The option to save every configurations helps in more complex scenarios, as does the `json` format in which configurations are saved, which, in a future version of MSF Designer, would allow for easy passing to other tools to check for graph isomorphism.

7 Discussion

Inspired by topological data analysis, MSF Designer enables mathematicians to design and visualize Morse–Smale functions on the sphere. It also supports the exploration of their associated Morse–Smale complexes and gradient vector fields. Our work also applies to the more

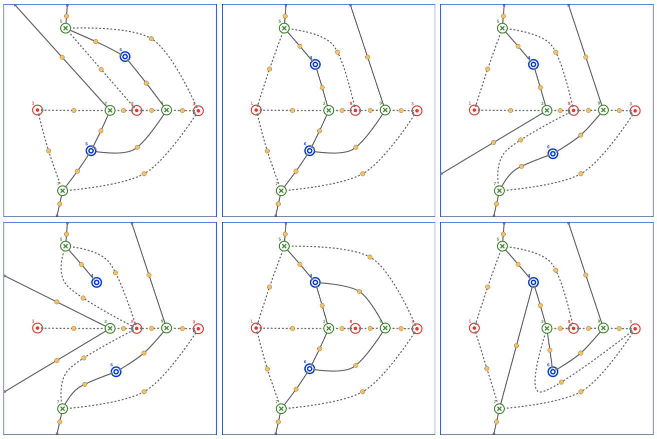

Fig. 21. Vector fields as combinatorial objects.

general setting of vector fields that are gradient-like for Morse–Smale functions [34, Theorem B]. A vector field is *gradient-like for f* if it is topologically trajectory equivalent to the gradient vector field ∇f of a function f and a Riemannian metric g_{ij} on $\mathbb{M}$. Morse vector fields are precisely the gradient-like vector fields without saddle-saddle connections (separatrices from a saddle to a saddle) [34].

Limitations. We limit our work to vector fields of Morse–Smale functions on the sphere, as functions on the sphere are well-studied [25] and the sphere is a simple, yet not trivial 2-manifold. Visualization of the surface of the sphere is also relatively easy, as the square boundary in the visuals created by MSF Designer represents a single point on the sphere. Due to the classification of 2-dimensional manifolds [19], any such manifold may be presented as a square, though with parts of the boundary identified, not necessarily contracted to a point. This would necessitate more work in ensuring a proper visualization for MSF Designer.

Extensions. A future mathematical application is to visualize the Wang, Fleitas, and Peixoto invariants (discussed in Sect. 6.1) directly alongside the barcode invariant. Such an application would allow mathematicians to see the effect of the elementary moves on these existing invariants, giving a unified language for working with different objects. A possible direction for broader impact is to develop an interface with other mathematical tools, such as those that check for graph isomorphism (see Sect. 6.5), for example, *SageMath* [1], *nauty and Traces* [21].

To enrich the visualization in the future, the graph of a Morse–Smale function could be visualized in real-time (e.g., Fig. 3). MSF Designer could be further extended to support the design of general Morse–Smale vector fields by allowing periodic trajectories. Such an extension is nontrivial as quadrangles alone are not sufficient to describe the domain decomposition with periodic trajectories, and we can no longer use the gradient of a function to approximate the vector fields.

Acknowledgments

This research was partially supported by NSF IIS-1910733 and DOE DE-SC0021015. This paper grew out of a discussion during the special workshop “Bridging Statistics and Sheaves” at the Institute for Mathematics and Applications (IMA) in May 2018. The authors would like to thank the organizers of the workshop and the IMA for hosting the event.

References

[1] SageMath. https://www.sagemath.org/.

[2] V. I. Arnold. Topological classification of Morse functions and generalisations of Hilbert’s 16-th problem. *Mathematical Physics, Analysis and Geometry*, 10(3):227–236, 2007.

[3] M. J. Catanzaro, J. M. Curry, B. T. Fasy, J. Lazovskis, G. Malen, H. Riess, B. Wang, and M. Zabka. Moduli spaces of Morse functions for persistence. *Journal of Applied and Computational Topology*, 4:353–385, 2020.

[4] T. K. Dey and R. Slechta. Edge contraction in persistence-generated discrete morse vector fields. *Computers & Graphics*, 74:33–43, 2018.

[5] H. Edelsbrunner and J. Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

[6] H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Morse-Smale complexes for piece-wise linear 3-manifolds. *ACM Symposium on Computational Geometry*, pages 361–370, 2003.

[7] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. *Discrete & Computational Geometry*, 30(1):87–107, 2003.

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

[9] H. Edelsbrunner, D. Morozov, and V. Pascucci. Persistence-sensitive simplification functions on 2-manifolds. In *Proceedings of the 22nd Annual Symposium on Computational Geometry*, pages 127–134, 2006.

[10] G. Fleitas. Classification of gradient-like flows on dimensions two and three. *Boletim da Sociedade Brasileira de Matemática - Bulletin/Brazilian Mathematical Society*, 6(2):155–183, 1975.

[11] R. Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

[12] A. Gyulassy, D. Günther, J. A. Levine, J. Tierny, and V. Pascucci. Conforming Morse-Smale complexes. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2595, 2014.

[13] A. Gyulassy, A. Knoll, K. C. Lau, B. Wang, P.-T. Bremer, M. E. Papka, L. A. Curtiss, and V. Pascucci. Interstitial and interlayer ion diffusion geometry extraction in graphitic nanosphere battery materials. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 22(1):916–925, 2015.

[14] A. Gyulassy, A. Knoll, K. C. Lau, B. Wang, P.-T. Bremer, M. E. Papka, L. A. Curtiss, and V. Pascucci. Morse-Smale analysis of ion diffusion for dft battery materials simulations. *Topology-Based Methods in Visualization (TopoInVis)*, 2015.

[15] A. Gyulassy, N. Kotava, M. Kim, C. Hansen, H. Hagen, and V. Pascucci. Direct feature visualization using Morse-Smale complexes. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1549–1562, 2012.

[16] J. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *IEEE Computer Architecture Letters*, 22(08):27–36, 1989.

[17] A. Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pages 453–460, 1998.

[18] A. Hertzmann and D. Zorin. Illustrating smooth surfaces. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 517–526, 2000.

[19] J. M. Lee. *Introduction to Topological Manifolds*. Springer New York, NY, 2011.

[20] Y. Matsumoto. *An Introduction to Morse Theory*. American Mathematical Society, 1997.

[21] B. McKay and A. Piperno. nauty and Traces: Graph canonical labeling and automorphism group computation. https://pallini.di.uniroma1.it/.

[22] J. Milnor. *Morse Theory*. Princeton University Press, New Jersey, 1963.

[23] K. Mischaikow and V. Nanda. Morse theory for filtrations and efficient computation of persistent homology. *Discrete & Computational Geometry*, 50(2):330–353, 2013.

[24] X. Ni, M. Garland, and J. C. Hart. Fair Morse functions for extracting the topological structure of a surface mesh. *ACM Transactions on Graphics (TOG)*, 23(3):613–622, 2004.

[25] L. I. Nicolaescu. Counting Morse functions on the 2-sphere. *Compositio Mathematica*, 144(5):1081–1106, 2008.

[26] A. A. Oshemkov. Morse functions on two-dimensional surfaces. encoding of singularities. *Trudy Matematicheskogo Instituta imeni VA Steklova*, 205:131–140, 1994.

[27] A. A. Oshemkov and V. V. Sharko. Classification of morse-smale flows on two-dimensional manifolds. *Matematicheskiĭ Sbornik*, 189(8):93–140, 1998.

[28] J. Palis and W. de Melo. *Geometric theory of dynamical systems: An introduction*. Springer-Verlag, 1982.

[29] M. M. Peixoto. On the classification of flows on 2-manifolds. In M. M. Peixoto, editor, *Dynamical Systems*, pages 389–419. Academic Press, 1973.

[30] K. Polthier and E. Preuss. Identifying vector fields singularities using a discrete hodge decomposition. In *Visualization and Mathematics III*, pages 112–134. Springer, 2003.

[31] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 465–470, 2000.

[32] A. Rockwood and S. Binderwala. A toy vector field based on geometric algebra. *Proceedings of the Application of Geometric Algebra in Computer Science and Engineering Conference*, pages 179–185, 2001.

[33] V. Sharko. Smooth and topological equivalence of functions on surfaces. *Ukrainian Mathematical Journal*, 55(5):832–846, 2003.

[34] S. Smale. On gradient dynamical systems. *Annals of Mathematics*, 74(1):199–206, 1961.

[35] J. Stam. Flows on surfaces of arbitrary topology. *ACM Transactions on Graphics*, 22(3):724–731, 2003.

[36] H. Theisel. Designing 2D vector fields of arbitrary topology. *Computer Graphics Forum*, 21(3):595–604, 2002.

[37] Y. Tong, S. Lombeyda, A. N. Hirani, and M. Desbrun. Discrete multiscale vector field decomposition. *ACM Transactions on Graphics*, 22:445–452, 2003.

[38] X. Tricoche, G. Scheuermann, and H. Hagen. Continous topology simplification of planar vector fields. *Proceedings of the conference on Visualization*, pages 159–166, 2001.

[39] G. Turk. Texture synthesis on surfaces. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 347–354, 2001.

[40] J. J. van Wijk. Image based flow visualization. *ACM Transactions on Graphics*, 21(3):745–754, 2002.

[41] J. J. van Wijk. Image based flow visualization for curved surfaces. *IEEE Visualization*, pages 123–130, 2003.

[42] X. Wang. The C*-algebras of Morse-Smale flows on two-manifolds. *Ergodic Theory and Dynamical Systems*, 10(3):565–597, 1990.

[43] L.-Y. Wei and M. Levoy. Texture synthesis over arbitrary manifold surfaces. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 355–360, 2001.

[44] J. Wejchert and D. Haumann. Animation aerodynamics. *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 19–22, 1991.

[45] R. Westermann, C. Johnson, and T. Ertl. A level-set method for flow visualization. *IEEE Visualization*, pages 147–154, 2000.

[46] L. Yan, T. B. Masood, R. Sridharamurthy, F. Rasheed, V. Natarajan, I. Hotz, and B. Wang. Scalar field comparison with topological descriptors: Properties and applications for scientific visualization. *Computer Graphics Forum*, 40(3):599–633, 2021.

[47] E. Zhang, K. Mischaikow, and G. Turk. Vector field design on surfaces. *ACM Transactions on Graphics*, 25(4):1294–1326, 2006.