TOPOLOGICAL ANALYSIS OF SCALAR FUNCTIONS FOR SCIENTIFIC DATA VISUALIZATION

by

Vijay Natarajan

Department of Computer Science
Duke University

Date: _____

Approved:

_____
Prof. Herbert Edelsbrunner, Supervisor

_____
Prof. Lars Arge

_____
Prof. John Harer

_____
Prof. Xiaobai Sun

_____

Dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Computer Science
in the Graduate School of
Duke University

2004

ABSTRACT

TOPOLOGICAL ANALYSIS OF SCALAR FUNCTIONS FOR SCIENTIFIC DATA VISUALIZATION

by

Vijay Natarajan

Department of Computer Science
Duke University

Date: _____

Approved:

_____
Prof. Herbert Edelsbrunner, Supervisor

_____
Prof. Lars Arge

_____
Prof. John Harer

_____
Prof. Xiaobai Sun

_____

An abstract of a dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Computer Science
in the Graduate School of
Duke University

2004

Abstract

Scientists attempt to understand physical phenomena by studying various quantities measured over the region of interest. A majority of these quantities are scalar (real-valued) functions. These functions are typically studied using traditional visualization techniques like isosurface extraction, volume rendering etc. As the data grows in size and becomes increasingly complex, these techniques are no longer effective. State of the art visualization methods attempt to automatically extract features and annotate a display of the data with a visualization of its features. In this thesis, we study and extract the topological features of the data and use them for visualization. We have three results:

- An algorithm that simplifies a scalar function defined over a tetrahedral mesh. In addition to minimizing the error introduced by the approximation of the function, the algorithm improves the mesh quality and preserves the topology of the domain. We perform an extensive set of experiments to study the effect of requiring better mesh quality on the approximation error and the level of simplification possible. We also study the effect of simplification on the topological features of the data.

- An extension of three-dimensional Morse-Smale complexes to piecewise linear 3-manifolds and an efficient algorithm to compute its combinatorial analog. Morse-Smale complexes partition the domain into regions with similar gradient flows. Letting n be the number of vertices in the input mesh, the running time of the algorithm is proportional to $n \log(n)$ plus the total size of the input mesh plus the total size of the output. We develop a visualization tool that displays different substructures of the Morse-Smale complex.

- A new comparison measure between k functions defined on a common d-manifold. For the case $d = k = 2$, we give alternative formulations of the definition based on a Morse theoretic point of view. We also develop visualization software that performs local comparison between pairs of functions in datasets containing multiple and sometimes time-varying functions.

We apply our methods to data from medical imaging, electron microscopy, and x-ray crystallography. The results of these experiments provide evidence of the usability of our methods.

To my parents

Acknowledgements

I owe whatever I have achieved to my parents. They have been there for me always, supporting my decisions and believing in me. I have learned a lot from them, most importantly my sense of values. I will treasure this for the rest of my life.

My advisor, Prof. Herbert Edelsbrunner, has been a major influence in my research career. I got introduced to the field of computational topology when I was still an undergrad and Prof. Subir Ghosh at T.I.F.R. handed me the introductory paper that Herbert had written on the subject. I had recently taken courses in topology and algebra and was excited that these subjects had practical relevance. This excitement has grown by leaps and bounds while working with Herbert. Discussion sessions with him helped me refine my reasoning. His meticulous attention to detail and constructive feedback while working on manuscripts helped me improve my writing skills. I will be forever grateful to him for his support and guidance during the past five years.

I have had the pleasure of working with two wonderful people: Prof. John Harer and Dr. Valerio Pascucci. I learned many mathematical ideas from John. He explains them in a way that makes it all seem so simple. I can only hope that other mathematicians that I collaborate with in future are like him. Valerio introduced me to the wonderful area of visualization and shared the insights that he had gained from his experiences while doing interdisciplinary research. These have been very valuable to me due to the nature of my work where I need to talk with scientists from other disciplines. He has also played the role of a mentor, helping me make my career decisions.

I was lucky to have great committee members in Prof. Lars Arge and Prof. Xiaobai Sun. I want to thank them for giving valuable comments on my thesis and for being so flexible in setting up a date for my final presentation. I will always be grateful to

Prof. Pankaj Agarwal for giving me valuable advice on many academic matters and career options. He has often gone out of the way to offer his help or give a word of encouragement. The administrative and lab staff members in the department have always been very helpful. I would like to particularly thank Diane Riggs: she has been of great assistance throughout the course of my stay in the department. I want to thank Celeste Hodges for her kind words of encouragement. Various students and postdocs at Duke and UNC gave much needed professional support, attending my practice talks and giving feedback, discussing ideas, answering questions etc. Besides being colleagues, they are my good friends as well. I want to thank Sathish Govindarajan, Yusu Wang, Ajith Mascarenhas, Gopi Meenakshisundaram, David Cohen-Steiner, Lipyeow Lim, Andrew Ban, and Vicky Choi and wish them a bright future. Going for coffee to the Bryan center with many of them, mostly with Yusu, served as a nice break in the afternoon.

During my stay at Duke, I have enjoyed the friendship of many people. I want to thank all of them for making my graduate life memorable. I could not have asked for better housemates than Hari and Mohan. Knowing Vijay Srinivasan has been a rewarding and humbling experience. Friday dinners with Jaidev, Lavanya and us "usual suspects" were awesome. Late night discussions on spirituality, with Vamsee and Pramod arguing endlessly, were thought provoking. Having Srikanth as an officemate in my first year was enlightening because I learned a lot about doing research. With Dmitriy as my officemate in the past few months, there was never a dull moment. I have been fortunate to be able to stay in touch with great friends from college especially through our mailing list *diljale*. These are some of the most talented and versatile people that I have met. They have and continue to inspire me in many ways.

My brothers and sisters-in-law have always encouraged me to pursue my dreams

and I want to thank all of them for their support: Jaishankar, Uma, Sunder and Subha. Finally, I want to thank my fiancée Mala for her patience and unconditional support.

This work was supported by NSF under grant NSF-CCR-0086013 and by Lawrence Livermore National Laboratory under sub-contracts with Duke.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis explores the use of topological analysis for studying features in scientific data. We restrict our study to scalar functions measured on a two- or three-dimensional space. We develop methods to compute topological properties of scalar functions, compare two functions based on their topological features, and preserve these properties while simplifying the representation of the data. In this chapter, we first motivate the need for a topological approach in order to obtain new and effective methods for analysis and visualization purposes. Next, we summarize the contributions of this thesis and introduce the underlying techniques used in our work. Finally, we outline the layout of the material in this dissertation.

1.1 Scalar Functions

A function uniquely maps members of one set to members of another set. We are interested in functions that map points from a 2- or 3-dimensional space to real values. These functions are called scalar functions. Graphs of familiar functions from elementary calculus are shown in Figure 1.1. Scientific datasets are often functions but typically do no have an analytic description of the function. In fact, the value of a function is measured at discrete points in the domain by physical means or computed using procedural methods. Different interpolation techniques are then used to derive a continuous and, if necessary, smooth representation of the function. We list below some of the sources of such functions along with a description of the properties studied in each case. This list is representative of some of the application areas that we think our methods can be applied to and is by no means exhaustive.

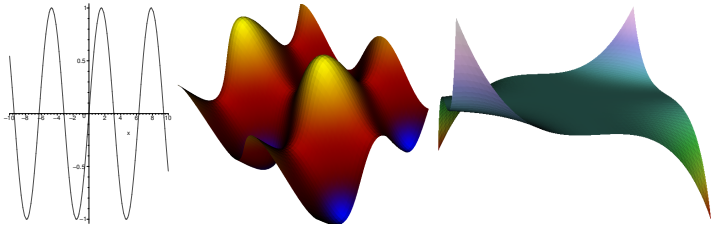

Figure 1.1: Graphs of some analytic functions. From left to right: $y = \sin(x)$, $z = \sin(x) + \cos(y)$, and $z = x^3y^3$.

Source 1 (X-ray Crystallography) X-ray crystallographers compute the electron density at various points of a molecular crystal using diffraction measurements from x-rays bouncing off the crystal. It is essential to know the electron density to perform structure related studies of the molecule. The electron density is a scalar function typically defined on a subset of the three-dimensional Euclidean space, $\mathbb{R}^3$.

Source 2 (Medical Imaging) Magnetic Resonance Imaging (MRI) is a popular technique used to take pictures of different slices of the human body. The atom density over the slice is mapped to a gray-scale image and studied by radiologists to detect tumors. The scalar function in this case is the density defined on a set of two-dimensional planes stacked together and can be viewed as a function on a subset of $\mathbb{R}^3$.

Source 3 (Computational Fluid Dynamics) The field of computational fluid dynamics studies efficient methods to compute fluid flow properties. Some physical quantities that play an important role in this context include pressure, temperature, and fluid density. The scalar function here is one of the above quantities defined either on the surface of the fluid or on the volume occupied by the fluid.

Source 4 (Electron Microscopy) Electron Microscopy is a technique used to obtain images of macromolecules by placing them in vitreous ice, taking pictures of

slices of the object and finally deriving the three-dimensional structure using computed tomography. The scalar function here is a density function defined on a subset of $\mathbb{R}^3$.

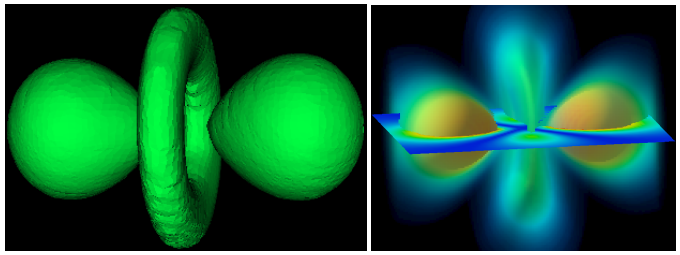

Figure 1.2: The figure on the left shows an isosurface extracted from an electron density of a hydrogen molecule. The figure on the right is a visualization of the same dataset using volume rendering where different sections of the volume are mapped to colors that correspond to the function value but with varying transparency so that the core is also visible.

Humans are able to easily interpret and comprehend visual information. The field of data visualization capitalizes on this ability and aims to give the user a deeper understanding of the data and the underlying laws governing it. This is achieved by providing a comprehensive display of the data along with annotations. A simple visualization method interprets the scalar value at each point as an additional coordinate and plots a graph, as shown in Figure 1.1. Obviously, this technique is useful only if the domain is one- or two-dimensional. Other techniques are available to handle higher-dimensional domains while being applicable to lower dimensions too. Figure 1.2 shows two of these techniques applied to visualize electron density data. Again, these techniques do have limitations and it is imperative to make a good choice for an effective and efficient visualization. A severe limitation of many of the existing techniques is the inability to handle huge datasets, which are becoming increasingly common. One approach towards solving this problem is to compress the data using

geometric techniques. Another approach is to automatically extract features from the dataset and present them to the user. Computing the topological properties of the domain is a step in the latter direction and has been studied, under different names, for both scalar functions [7, 34, 112] and for vector functions [50, 59, 99].

1.2 Topological Analysis

We now discuss what topological analysis means and how it can be used for extracting features from scientific datasets. For simplicity, let us consider a smooth scalar function defined on a subset of the plane, $\mathbb{R}^2$. The function value can be interpreted as a third coordinate and a surface plot or *terrain* can be drawn as shown for the function $\sin(x) + \cos(y)$ in Figure 1.1. Distinct features in this terrain include the mountain peaks, valleys, and mountain passes between the peaks. Each one of these features correspond to a point in the plane where the gradient of the function is zero. These points are called the *critical points* or *singularities* of the function. A topological analysis of the function refers to the computation and classification of structures over the critical points.

All our methods have their mathematical footing in Morse theory. An important result in this theory states that any generic smooth function has a discrete set of non-degenerate critical points. It further says that in the neighborhood of each non-degenerate critical point, the function has a quadratic behavior. In particular, in the neighborhood of the corresponding critical point, the surface looks like one of those shown in Figure 1.3. This quadratic behavior is exhibited in higher dimensions too. We transport this result to piecewise linear domains in order to classify critical points. Morse theory also shows the invariance of certain properties of critical points over all smooth functions defined on a given domain. This suggests that the critical point behavior is global in nature although the classification can be done locally.

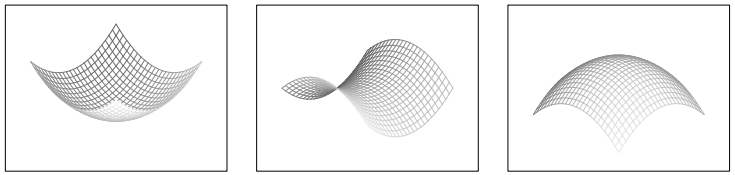

Figure 1.3: Surface in the neighborhood of non-degenerate critical points of a smooth function on the plane: (left to right) a minimum, saddle, and maximum.

Note that the number of critical points is usually smaller than the input size. Therefore, a structure built on top of them will be easy to comprehend. Moreover, a display of this structure will give a global view of the function behavior making it easy to identify regions of interest and perhaps obtain a magnified view in later iterations. In cases where the number of critical points is too large to handle we can apply simplification techniques to reduce their number. We do not address this issue in this thesis and mention it as future work. Having given the general context, we now describe the contributions of this thesis.

1.3 Contributions

An important aspect of the techniques we develop is the separation of combinatorial and numerical components of algorithms. This separation leads to implementations that are efficient because combinatorial algorithms are typically simple and fast. We also simulate different properties of smooth functions like genericity, differentiability etc. and hence transport intuition from the smooth setting to produce consistent results in the piecewise linear domain. We have three collections of results: density map simplification [72], 3D Morse-Smale complexes [28] and scalar function comparison [29].

Density map simplification. We describe an algorithm for simplifying a scalar

function represented by a tetrahedral mesh. We refer to the function as a *density* function/map due to historical reasons: density is a frequently measured quantity in the scientific community. The algorithm consists of two main ingredients:

1. A topology preserving edge contraction operation that ensures the simplified models are homeomorphic to the original.

2. A quadratic cost function that determines the order of edge contractions. This cost function attempts to both preserve the density map as well as improve the quality of the tetrahedra in the mesh.

We perform extensive experiments to study the effect of increasing the relative weight of the mesh quality improvement on the density map preservation as well as the size of the coarsest mesh reachable by simplification. We also study statistics of the critical point count during the simplification process in order to compare the geometric and topological simplification.

3D Morse-Smale Complexes. A Morse-Smale complex partitions the domain of the scalar function into regions with uniform gradient flow behavior. We consider the case where the domain is three-dimensional. Morse-Smale complexes were defined more than fifty years ago for smooth functions [98]. We extend these definitions to the piecewise linear domain. Further, we give algorithms for constructing a complex that is combinatorially indistinguishable from the Morse-Smale complex. We develop a visualization tool that combines standard visualization techniques with the display of sub-structures of the Morse-Smale complex. Working with material scientists, we study dislocations in a copper crystal using our visualizations. We also study the shape of biological macromolecules using one-dimensional substructures of the Morse-Smale complex.

Scalar function comparison. We define a new comparison measure between k

functions defined on a common d-manifold. For the case when $k = d = 2$, we give alternative formulations for the measure that relates it to the Jacobi set, which consists of critical points of one function restricted to level sets of the other. We develop visualization software that enables the user to compare multiple scalar functions. We extend the software to time-varying functions and use it to study two functions measured during a combustion process.

1.4 Layout of Material

We begin the dissertation with this chapter motivating the study of topology for the development of effective techniques for scientific data visualization. The next three chapters discuss the three contributions of this thesis: density map simplification, 3D Morse-Smale complexes, and scalar function comparison. We give our conclusions and mention future work in Chapter 5. There are various terms used in this dissertation that may not be new to a reader who is familiar with the areas of topology and Morse theory. However, other readers may require an introduction to some of the basic terminology and feel a need to refer back to the definitions while reading the dissertation. We have collected these terms together with brief descriptions and placed them in Appendix A, which is organized into four topics: manifolds, simplicial complexes, homology, and Morse theory.

Chapter 2

Density Map Simplification

We consider scientific datasets that describe density (scalar) functions over a three-dimensional domain and study the relationship between topological and geometric simplification. In this chapter, we describe a simplification algorithm that constructs the coarser representation with a three-fold objective:

- get a good approximation of the scalar function;
- preserve the topological type of the mesh and;
- improve the quality of the tetrahedral mesh elements.

We evaluate our results by computing the approximation error, visualizing the function using isosurface extraction, and validating the topological correctness of the mesh. We assess the change in topological features by tracking the number of critical points during the simplification process.

2.1 Introduction

2.1.1 Motivation

Scientific datasets are rapidly growing in size making it difficult to visualize them using traditional techniques. One popular approach to overcome this problem is to reduce the size of the data and work with the resulting coarser representation. The smaller size results in fast visualization and analysis of the data. The techniques used to reduce the size of the data are typically geometric in nature *i.e.* they work on the elements of the underlying mesh. Changes to the geometry might affect the

topology of the mesh, which in turn could affect the analysis drastically. Therefore, it is important to control the change in topology during the geometry simplification.

2.1.2 Prior Work

Many of the techniques used for tetrahedral mesh simplification are extensions of the ones used for surface simplification. Surface mesh simplification has been studied in a wide range of communities including cartography, computational geometry, computer graphics, computer vision, finite element analysis, and scientific visualization. Heckbert and Garland [47] survey the surface simplification algorithms from the above mentioned fields and classify them based on the type of technical problem they solve. Decimation methods form the class of simplification algorithms that start with a polygonal representation and simplify it until the desired level of approximation is achieved. A popular approach to surface decimation uses edge contractions [41, 42, 51, 53]. An edge contraction deletes an edge by merging its endpoints into one vertex. The sequence of edge contractions is critical for its success and different criteria have been studied to prioritize the edges. Garland and Heckbert [35] use a quadric error metric to determine the order of edge contractions and extend this error measure to surfaces with attributes [36]. Hoppe [52] proposed an extension of the quadric error metric for storing errors in surface attributes, which scales better with the number of attributes. Cignoni et al. [19] give a comparison of some of the above mentioned algorithms. Various other approaches to surface decimation, like vertex clustering [81], vertex removal [84], and triangle contraction [38], have also been proposed.

Renze and Oliver [80] extend the vertex removal approach to perform volume decimation. Trotts et al. [101] extend the triangle contraction operation described by Gieng et al. [38] to a tetrahedral contraction operation and construct multi-

ple levels of tetrahedral meshes approximating a density function. Popovic and Hoppe [77] extended the edge contraction operation used to generate progressive surfaces meshes [51] in order to handle triangulations in arbitrary dimensions. Staadt and Gross [93] apply this extension to the special case of 3D and give a robust implementation for tetrahedral meshes. They also address the definition of appropriate cost functions for specific applications, like the finite element method. Van Gelder et al. [102] compare a mass-based and a density-based metric for use in rapidly decimating a tetrahedral mesh. Cignoni et al. [17] compare different cost functions that have been used to prioritize edge contractions. Software performing tetrahedral mesh simplification is available now both as freeware visualization packages [18, 104] and as part of commercial visualization packages like Amira [97], VolView [56], and Ansys ICEM CFD [4].

Topological simplification of scalar fields has also been studied, primarily in the graphics and visualization community, albeit using different notions of what constitutes a topological feature. He et al. [49] reduce the number of holes, tunnels, and cavities in a surface by representing the surface as a zero set of a function in $\mathbb{R}^3$ and then extracting isosurfaces from smoothed versions of the function. Guskov and Wood [45] order the tunnels in a surface and remove the smaller ones using local remeshing techniques. Bremer et al. [11] use a hierarchy of Morse-Smale complexes to generate simplified scalar fields. Some recent work addresses the related issues of preserving the topology of isosurfaces [16] or controlling the topology simplification of isosurfaces [37] during the volume simplification.

2.1.3 Approach and Results

We simplify a tetrahedral mesh representing a three-dimensional density function by a sequence of edge contractions. We modify the quadric error measure described by

Garland and Heckbert [35] to combine three goals in prioritizing the contractions:

- the accurate approximation of the density function;
- the faithful preservation of global topological type of the mesh;
- the improvement of the mesh quality defined in terms of angles.

To pursue the first goal, we extend well established ideas from $\mathbb{R}^2$ to $\mathbb{R}^3$. The second goal needs no new results and is based on applying tests described in [22]. To pursue the third goal, we develop a new idea, namely the addition of particular hyperplanes to the quadrics that have a positive influence on the mesh quality. We perform various experiments which show that a small relative weight of the third goal furnishes dramatic improvements in mesh quality. Further increasing that weight adversely affects the approximation of the density function. Lindstrom and Turk [62] attempt to improve the mesh quality by introducing an additional constraint while determining the location of the vertex that replaces an edge upon its contraction. The effect of their method is limited because the new constraint applies only if the other constraints lead to an ambiguous solution, which happens typically at the mesh boundary. In contrast, our method influences the placement of every vertex and also affects the sequence in which the edges are contracted. By adding extra hyperplanes into the quadric error metric, we pro-actively influence the shape of the tetrahedra created after an edge contraction. This is stronger than previous techniques, like that described in [17], which merely do not perform an edge contraction if it results in badly shaped tetrahedra. We observe an interesting side-effect: a small but non-zero weight on the mesh quality improvement results in better approximations of the density map.

Another surprising finding concerns the relationship between geometric and topological simplification, the latter aiming at preserving the critical point structure of the function. While geometric simplification preserves the overall shape of the func-

tion, it sometimes introduces a large number of spurious critical points that confuse the topological picture. However, we show that these newly created critical points can be converted to regular points by a small change in function value and hence that they are of less importance. In other words, geometric simplification is compatible with but not a substitute for topological simplification.

2.2 Edge contraction

The input to our simplification algorithm is a triangulation K of a 3-manifold with boundary. We extend K to the triangulation of a 3-manifold without boundary by connecting a dummy vertex to each boundary component. The dummy vertex can be connected to simplices on the boundary using the cone operation. The *cone* from a vertex x to a k-simplex σ is the convex hull of x and σ, which is the $(k+1)$-simplex $x\sigma$. The operation is defined only if x is not an affine combination of the vertices of σ. If $\mathrm{Bd}\, K$ is connected we just add one dummy vertex, ω, and denote the extended triangulation by $\mathrm{K}_\omega = K \cup \omega \mathrm{Bd}\, K$. The link of a simplex $\sigma \in \mathrm{K}_\omega$ is denoted by $\mathrm{Lk}_\omega \sigma$. For a simplex $\sigma \in \mathrm{Bd}\, K$, the link of σ within the boundary is denoted by $\mathrm{Lk}_{\mathrm{Bd}}\, \sigma$. In a 3-manifold without boundary, the stars and the links are particularly simple: the link of a vertex is a sphere, that of an edge is a circle, and that of a triangle is a pair of vertices. Similarly in a 2-manifold, the link of a vertex is a circle and that of an edge is a pair or vertices. Note that we can determine whether a given simplicial complex is a triangulation of a 3-manifold by checking if the link at every vertex is homeomorphic to the 2-dimensional sphere $\mathbb{S}^2$. In this section, we discuss the basic operation in our algorithm that contracts an edge to a vertex. We describe the conditions used to determine if an edge contraction changes the topological type and the cost, associated with each edge, that is used to order the edges.

Upon contraction of an edge ab, we replace it with a vertex c. This changes the

triangulation only in the neighborhood of a and b. In particular, the cofaces of a and b are deleted and simplices connecting c to the boundary of the created hole are added. Formally, these simplices are the cones from c to simplices in the link of the set of simplices $L = \{ab, a, b\}$. Figure 2.1 illustrates an edge contraction but shows only a subset of the simplices that are removed and added.

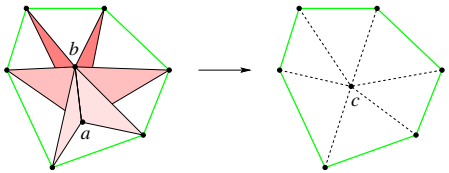

Figure 2.1: Edge contraction: the edge ab is contracted to the vertex c. Only the changes in the star of the edge ab are shown.

2.2.1 Preserving Topology

Our simplification algorithm performs a sequence of edge contractions on the tetrahedral mesh. Each edge contraction preserves the topological type of the mesh. The algorithm recognizes the edges that can be contracted without changing the topological type by looking at their neighborhoods. The ability to make this judgment based on local computations is crucial for the efficiency of our algorithm.

A 3-*complex* is a simplicial complex consisting of tetrahedra, triangles, edges and vertices. Dey et al. [22] derive local criteria, called link conditions, for recognizing when an edge contraction in a 3-complex preserves the topological type. The link conditions compare the link of the edge ab that is to be contracted with the links of its endpoints. Figure 2.2 shows a situation in a 3-manifold where a contraction would change the topology. In the case of a 3-manifold with boundary, N, Dey et al. [22] show that the contraction of an edge ab preserves the topological type if the intersection of the links of the two vertices equals the link of the edge, and this is

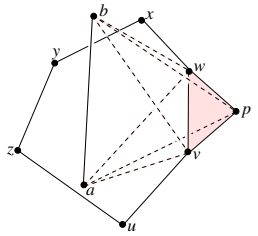

Figure 2.2: Triangle vwp lies in the link of both a and b. After contracting ab to a new vertex c, the triangle vwp belongs to only one tetrahedron, namely $vwpc$. The neighborhood of a point in vwp is thus no longer homeomorphic to $\mathbb{R}^3$. The ring of edges in $\operatorname{Lk} ab$ is $uvwxyzu$.

true both in the extended 3-complex and in the boundary of the 3-complex:

$$\operatorname{Lk}_\omega a \cap \operatorname{Lk}_\omega b = \operatorname{Lk}_\omega ab; \quad \text{and} \tag{2.1}$$

$$\operatorname{Lk}_{\mathrm{Bd}} a \cap \operatorname{Lk}_{\mathrm{Bd}} b = \operatorname{Lk}_{\mathrm{Bd}} ab. \tag{2.2}$$

2.2.2 Specialized Link Conditions

In order to implement the above conditions, we would have to consider the cofaces of ω as special cases because these are not explicitly stored. To simplify the implementation, we eliminate ω from the condition. We have three cases, depending on whether ab, a, and b belong to the boundary or the interior.

`Case` 1: $a, b \in \operatorname{Bd} N$ and $ab \notin \operatorname{Bd} N$. The contraction of ab would change the topological type of N by pinching. It is therefore prohibited.

`Case` 2: At least one of a or $b \notin \operatorname{Bd} N$. Without loss of generality, assume that b is not on the boundary. Since b and hence ab are not on the boundary, $\operatorname{Lk}_{\mathrm{Bd}} b$ and $\operatorname{Lk}_{\mathrm{Bd}} ab$ are not defined and Condition (2) does not apply. The only vertices and edges whose links contain ω or any of its cofaces are the ones on the boundary. This implies $\operatorname{Lk}_\omega b = \operatorname{Lk} b$ and $\operatorname{Lk}_\omega ab = \operatorname{Lk} ab$. Condition (1) now simplifies to $\operatorname{Lk} a \cap \operatorname{Lk} b = \operatorname{Lk} ab$.

`Case` 3: $ab \in \operatorname{Bd} N$. We necessarily also have a and b on the boundary. We partition $\operatorname{Lk}_\omega a$ into $\operatorname{Lk} a$ and the set $\omega\operatorname{Lk}_{\operatorname{Bd}} a$ that contains the simplices that are cofaces of ω. Similarly, we partition $\operatorname{Lk}_\omega b$ and $\operatorname{Lk}_\omega ab$ and obtain

$$(\operatorname{Lk} a \cap \operatorname{Lk} b) \mathbin{\dot\cup} (\omega\operatorname{Lk}_{\operatorname{Bd}} a \cap \omega\operatorname{Lk}_{\operatorname{Bd}} b) = \operatorname{Lk} ab \mathbin{\dot\cup} \omega\operatorname{Lk}_{\operatorname{Bd}} ab,$$

which is equivalent to Condition (1). Three of the terms contain no simplex in the star of ω and the other three contain only simplices in the star of ω. We can therefore express the condition as a conjunction of two conditions. We further simplify by removing ω from the second set of three terms and get

$$\begin{aligned} \operatorname{Lk} a \;\cap\; \operatorname{Lk} b &= \operatorname{Lk} ab; \quad \text{and} \\ \operatorname{Lk}_{\operatorname{Bd}} a \;\cap\; \operatorname{Lk}_{\operatorname{Bd}} b &= \operatorname{Lk}_{\operatorname{Bd}} ab. \end{aligned}$$

The following lemma summarizes the results of the case analysis by stating a modified link condition for a 3-manifold with boundary.

LEMMA 1 If N is a 3-manifold with boundary then the contraction of an edge $ab \in N$ preserves the topological type if one of the following is true:

(1) at least one of a and b does not belong to $\operatorname{Bd} N$ and $\operatorname{Lk} a \cap \operatorname{Lk} b = \operatorname{Lk} ab$;

(2) ab belongs to $\operatorname{Bd} N$, $\operatorname{Lk} a \cap \operatorname{Lk} b = \operatorname{Lk} ab$, and $\operatorname{Lk}_{\operatorname{Bd}} a \cap \operatorname{Lk}_{\operatorname{Bd}} b = \operatorname{Lk}_{\operatorname{Bd}} ab$.

Lemma 1 applies to our data, which in all cases consists of a tetrahedral mesh of a cube in $\mathbb{R}^3$. In Section 2.3.3, we describe the procedure that checks the link conditions and explain how to make it more efficient than the direct implementation of the formulas.

2.2.3 Cost of Contraction

We use a cost associated with each edge to determine the order of contractions. A vertex is a point in $\mathbb{R}^4$, with three spatial coordinates and the fourth giving the

function value. Each vertex and edge is associated with a finite set of hyperplanes in $\mathbb{R}^4$. The cost of an edge is the minimum, over all points of $\mathbb{R}^4$, of the sum of square distances between the point and the hyperplanes associated with the edge and its endpoints. This cost can be computed from the hyperplanes using an extension of the quadric error measure proposed by Garland and Heckbert [35]. Hoppe [52] extends the quadric error metric for surface attributes by first performing a projection to $\mathbb{R}^3$ and then computing geometric and attribute errors. This approach is particularly efficient when the number of attributes is large unlike our setting with only one attribute. We chose to use the simple and more direct extension of Garland and Heckbert's quadric error metric, namely performing a projection in $\mathbb{R}^4$ for computing the error.

The purpose of the hyperplanes associated with a vertex is to locally preserve the density function. We use hyperplanes spanned by tetrahedra of the mesh. Initially, each vertex is associated with the set of hyperplanes spanned by the tetrahedra in its star. When we create a new vertex c by contracting the edge ab we associate the union of the sets of a and b to c. The purpose of the hyperplanes associated with an edge is to locally improve the quality of the mesh. We use perpendicular bisectors of edges. Specifically, the hyperplanes associated with ab are the bisectors of the edges in the link of $L = \{ab, a, b\}$. For each edge in this link, we take the bisecting plane in $\mathbb{R}^3$ and extend it vertically to a hyperplane in $\mathbb{R}^4$. Figure 2.3 illustrates this idea one dimension lower, where the link of a contractible closed edge is a circle. The rationale for this choice of hyperplanes is to encourage almost spherical links of new vertices. As a consequence, the new tetrahedra are almost isosceles, with three almost equally long edges. Since there are no preferred vertices, we really encourage regular tetrahedra. The contraction of an edge ab causes a change in the link of all vertices in $\operatorname{Lk} a \cup \operatorname{Lk} b$ and thus requires an update in the sets of hyperplanes associated with

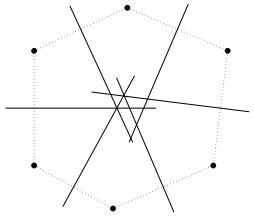

Figure 2.3: The dotted link of a closed edge and the solid bisectors of its edges.

the edges incident to these vertices.

Let H be a set of hyperplanes and $x = (x_1, x_2, x_3, x_4)^T$ a point in $\mathbb{R}^4$. Let the unit normal of a hyperplane $h \in H$ be $v_h = (v_1, v_2, v_3, v_4)^T$ and the offset $\delta_h = -p^T \cdot v_h$, for any point $p \in h$. The square of the distance between h and x is given by:

$$\begin{aligned} D_h &= ((x-p)^T \cdot v_h)^2 = (x^T \cdot v_h - p^T \cdot v_h)^2 \\ &= (x^T \cdot v_h + \delta_h)^2 = (\mathbf{x}^T \cdot \mathbf{v}_h)^2 \\ &= (\mathbf{x}^T \cdot \mathbf{v}_h)(\mathbf{v}_h^T \cdot \mathbf{x}) = \mathbf{x}^T(\mathbf{v}_h \cdot \mathbf{v}_h^T)\mathbf{x}, \end{aligned}$$

where $\mathbf{x} = (x_1, x_2, x_3, x_4, 1)^T$ and $\mathbf{v}_h = (v_1, v_2, v_3, v_4, \delta_h)^T$. The sum over all $h \in H$ is

$$D = \sum_{h \in H} D_h = \mathbf{x}^T \left(\sum_{h \in H} \mathbf{v}_h \cdot \mathbf{v}_h^T \right) \mathbf{x}.$$

The 5-by-5 matrix $\mathbf{Q} = \sum_{h \in H} \mathbf{v}_h \cdot \mathbf{v}_h^T$ is symmetric and positive semi-definite and is called the *fundamental quadric* of H. Instead of storing sets of hyperplanes, we store their fundamental quadrics. This representation requires two types of updates whenever we contract an edge ab. First the quadric of the new vertex c is computed as the sum of the quadrics of a and b. This new quadric really represents a multi-set of hyperplanes because a hyperplane associated with both endpoints is now counted twice. The difference to the quadric of the set (without double-counting) is however small since a single hyperplane cannot be counted more than four times. Second, the

contraction changes all edges that have a or b as endpoint. We update the quadrics stored at edges associated with the changed bisectors by subtracting the contributions of the old and adding the contributions of the new bisectors.

Although the edge contraction operation preserves the continuity of the model, it does not handle the boundary very well. Following the solution proposed by Garland and Heckbert [35], we rectify this by adding boundary constraints. For each boundary triangle, we include the hyperplane passing through the triangle and perpendicular to the hyperplane spanned by the tetrahedron that contains the triangle as a face. Further, these new hyperplanes are weighted with a large penalty value preventing vertices from moving too far from the boundary. Weighted hyperplanes can be easily incorporated into the current setting. The square distance between a vertex x and a hyperplane h with weight w_h is now $D_h = w_h \cdot \mathbf{x}^T(\mathbf{v}_h \mathbf{v}_h^T)\mathbf{x}$. The sum of square distances to all hyperplanes can be derived as before. We compute new quadrics by adding old ones, same as before. The only change is in the step where we compute the initial quadrics.

Another place where we use weights is in controlling the influence of the mesh quality improving hyperplanes on the simplification process. Each such hyperplane is weighed by a constant $\varphi \geq 0$, which we refer to as the *mesh quality factor*. For example, $\varphi = 0$ corresponds to no influence from these hyperplanes and $\varphi = 1$ corresponds to equal influence of both types of hyperplanes.

2.2.4 Optimal Vertex Placement

The cost of an edge ab depends also on the location of the new vertex. In the generic case, there is a unique location $c \in \mathbb{R}^4$ that minimizes the error. This minimum is

given by setting the partial derivatives to zero, for $i = 1, 2, 3, 4$:

$$\begin{aligned}\frac{\partial D(x)}{\partial x_i} &= \frac{\partial \mathbf{x}^T}{\partial x_i} \cdot \mathbf{Q} \cdot \mathbf{x} + \mathbf{x}^T \cdot \mathbf{Q} \cdot \frac{\partial \mathbf{x}}{\partial x_i} \\ &= \mathbf{q}_i^T \cdot \mathbf{x} + \mathbf{x}^T \cdot \mathbf{q}_i \\ &= 0,\end{aligned}$$

where $\mathbf{q}_i^T$ is the i-th row and $\mathbf{q}_i$ the i-th column of $\mathbf{Q}$. The solution is given by $c = -Q^{-1} \cdot q$, where Q is the upper left 4-by-4 submatrix of $\mathbf{Q}$ and q is the 4-vector consisting of the upper four entries in the fifth column of $\mathbf{Q}$.

In the non-degenerate case, Q has rank four. We detect degeneracies by estimating the rank of Q before computing c. Ranks three, two and one correspond to a line, plane and hyperplane of minima in $\mathbb{R}^4$, respectively. Note that Q has at least one non-zero diagonal element and hence the rank is never zero. In each of the three degenerate cases, we add the contributions of additional hyperplanes to increase the rank of the matrix. We estimate the rank by comparing the coefficients of the characteristic polynomial given by

$$\det(Q - \lambda I) = \sigma_4 - \lambda\sigma_3 + \lambda^2\sigma_2 - \lambda^3\sigma_1 + \lambda^4.$$

Here, σ_4 is the determinant and σ_1 is the trace of Q. The other two coefficients are sums of 3-by-3 and 2-by-2 minors of Q. We note that the coefficients are cheaper to compute than the eigenvalues and may be substituted for the latter in estimating the rank of the matrix. Specifically, we consider Q to have rank three, two and one, respectively, if the absolute value of $256\sigma_4/\sigma_1^4$, $16\sigma_3/\sigma_1^3$ and $8\sigma_2/\sigma_1^2$ is small. Restricting the above ratios to small values is equivalent to bounding the number of eigenvalues of Q that are close to zero.

Let ab be the edge being contracted. In the case of a rank three matrix, we add the contribution of the hyperplane normal to the line of minima that passes through

the midpoint of ab to the fundamental quadric, hoping that the rank increases to four. If it does not, then our fall back strategy is to select one of the endpoints of ab or its midpoint as the new vertex location. In the case of a rank two matrix, we have a plane of minima. We compute two hyperplanes that are orthogonal to each other and the plane and that pass through the midpoints of ab and add their contributions to the quadric. We compute the first hyperplane by taking the cross product of the two independent rows, and the second by taking the wedge product of the now three independent rows. Similarly, in the case of a rank one matrix, we get three new hyperplanes and add their contributions to the quadric.

2.3 Algorithm

In this section, we describe our implementation of the edge contraction operation. Various choices were made in determining the order of contractions, recognizing topology preserving contractions, and updating all data structures. We begin with the data structures and the outline of the simplification procedure.

2.3.1 Data Structure

We use a heap to implement the priority queue for the edges of the mesh. Along with the edges, we store the cost of contraction and the optimal vertex location. We store the mesh in a triangle-edge data structure [69], which is a version of the more general edge-facet data structure by Dobkin and Laszlo [23]. It is made up of triangles, each represented by the six possible orderings of its three vertices. As illustrated in Figure 2.4, each ordering maintains a pointer to the next triangle reached by a rotation about the edge of its first two vertices. The list of triangles is stored in an array. The coordinates of the vertices are stored in another array, and the indices of the vertices in this array are used as vertex names.

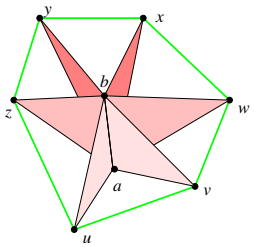

Figure 2.4: Each ordering abv stores a pointer to the next triangle: $abv.\texttt{next} = abw$. By following these pointers, we traverse the ring of triangles in the star of ab. After contracting ab to c, the ring of triangles becomes a ring of edges around c.

2.3.2 Algorithm Overview

Function SIMPLIFY performs a sequence of edge contractions to simplify the tetrahedral mesh K. It continues until the mesh reaches a user-specified number of at most v_0 vertices or no edge can be contracted without changing the topological type, whichever occurs first.

```
Mesh SIMPLIFY (Mesh K)
    initialize priority queue PQ with set of edges in K;
    while #vertices in K exceeds v_0 do
        pop the minimum cost edge from PQ and call it ab;
        if PRESTOP (ab) then
            delete edges in St a and St b from PQ;
            update costs of edges in St x, x ∈ Lk a ∪ Lk b;
            K = CONTRACT (K, ab, c);
            insert edges in St c into PQ;
        endif
    endwhile;
    return K.
```


The remainder of this section explains the two main functions used in this algorithm.

2.3.3 Checking Link Conditions

Function PRESTOP uses the two conditions in Lemma 1 to determine whether or not the contraction of ab preserves the topological type of K. We prevent redundant tests by first checking which part of the edge is on the boundary and then test zero, one or two conditions.

```
boolean PRESTOP (Edge ab)
  if a, b ∈ Bd K and ab ∉ Bd K then return FALSE endif;
  if ab ∉ Bd K then return LINKCOND1 (ab) endif;
  if ab ∈ Bd K then return (LINKCOND1 (ab) and LINKCOND2 (ab)) endif.
```

Functions LINKCOND1 implementing Condition (1) and LINKCOND2 implementing Condition (2) use enumerations of the simplices in the link of a vertex or edge. In Function LINKCOND2, we also need the restrictions of these links to the boundary of K, and to facilitate their computation, we label each triangle in $\mathrm{Bd}\,K$. Each link is computed by a local search procedure that starts at an ordered triangle provided by the vertex or edge for which we compute the link. Next, we describe the implementation of Function LINKCOND1, which determines whether or not the intersection of the links of a and b contains simplices that do not belong to the link of ab. We use a marking mechanism to keep track of the processed vertices.

```
boolean LINKCOND1 (Edge ab)
  foreach v ∈ Lk a do MARK (v) endfor;
  foreach v ∈ Lk ab do UNMARK (v) endfor;
  foreach v ∈ Lk b do if ISMARKED (v) then return FALSE endif endfor;
  return TRUE.
```

After testing the three links, we unmark all vertices again. We repeat the same test for edges in the intersection of the links restricting it to edges that connect two vertices in $\mathrm{Lk}\, ab$. It is not necessary to test triangles. Condition (2) is tested in a similar manner by Function LinkCond2, which traverses the links of a, b and ab on the boundary of the mesh.

2.3.4 Contracting Edges

Function Contract updates the mesh K by contracting an edge ab as follows:

```
Mesh Contract (Mesh K, Edge ab, Vertex c)
  foreach triangle axy ∈ St a do
    if x, y ≠ b then add cxy to K endif
  endfor;
  foreach triangle bxy ∈ St b do
    if x, y ≠ a and cxy ∉ K then add cxy to K endif
  endfor;
  delete all triangles in St a and St b from K;
  return K.
```

2.3.5 Reentering Edges

Note that the contraction of ab may change the status of other edges in the mesh. We are interested in edges xy that violate the conditions of Lemma 1 before the contraction of ab and that satisfy these conditions after the contraction of ab. We say these edges *turn contractible*. We detect the edges that have the potential for turning contractible and add them to the priority queue, using labels to avoid duplicate entries. We prove below that only edges in a relatively small subset of the link of c can turn contractible. This result is essential for an efficient detection of these edges.

Lemma 2 If the contraction of ab causes another edge xy to turn contractible x and y are contained in $\mathrm{Lk}\, ab$.

Proof. We have $\mathrm{Lk}\, xy \subseteq \mathrm{Lk}\, x \cap \mathrm{Lk}\, y$ for every edge xy, and we have equality if xy does not violate the conditions in Lemma 1. Suppose $\mathrm{Lk}\, x \cap \mathrm{Lk}\, y$ consists of the cycle $\mathrm{Lk}\, xy$ plus some additional simplices. The only way the contraction of ab can cause xy to turn contractible is by removing these extra simplices. Now, for $\mathrm{Lk}\, x \cap \mathrm{Lk}\, y$ to shrink, we need a and b in both links. Since $a \in \mathrm{Lk}\, x$ iff $x \in \mathrm{Lk}\, a$, this is equivalent to $x, y \in \mathrm{Lk}\, a \cap \mathrm{Lk}\, b$. Lemma 2 follows because ab satisfies both link conditions, particularly $\mathrm{Lk}\, a \cap \mathrm{Lk}\, b = \mathrm{Lk}\, ab$. □

2.3.6 Accumulating Edge Bisectors

We conclude this section with a brief comparison between two different implementations of the mesh quality improvement using bisecting hyperplanes. The algorithm, as explained above, uses a *memoryless* implementation in which the bisectors at each step are taken for edges in the current mesh. Alternatively, we could accumulate the bisectors of edges in the original mesh, similar to the way we accumulate hyperplanes spanned by tetrahedra in the original mesh. Initially, each vertex stores the quadric defined by the hyperplanes bisecting the edges in the link. Note that in the case of a vertex, the link is precisely the boundary of the star. Later in the process the vertex represents a collection of vertices in the original mesh and stores the quadric defined by the hyperplanes bisecting edges in the boundary of union of vertex stars. The link condition guarantees that this is a topological ball and its boundary is a topological sphere. The quadric of this set of hyperplanes can be computed by accumulation, but there is a complication caused by the need to remove hyperplanes that bisect edges in the interior of the ball. We cope with this complication by inclusion-exclusion:

- initialize a quadric for each simplex of the original mesh defined by the bisectors of the edges in the boundary of the star of the simplex;
- upon contracting the edge ab to the vertex c, compute the quadric as $\mathbf{Q}(c) = \mathbf{Q}(a) + \mathbf{Q}(b) - \mathbf{Q}(ab)$.

Similarly, we use inclusion-exclusion to compute the quadrics of newly formed edges and triangles. We note that inclusion-exclusion is also the preferred way to accumulate the quadrics of shape preserving hyperplanes, except that the simpler method of just adding quadrics commits only the negligible error of double-counting certain hyperplanes.

2.4 Experiments

There are two parameters that affect the performance of our algorithm: the target vertex count, v_0, and the relative weight of the hyperplanes that were added into the quadric to improve the mesh quality, φ. We perform various experiments in order to determine an good values for φ. We evaluate the results by computing approximation errors, visualizing the simplified mesh through isosurfaces and looking at critical point statistics. We compute the approximation error of a simplified mesh as the root mean square and maximum of the error at each vertex of the simplified mesh. The error at a vertex is the difference between its density value and density value of the corresponding point in the original mesh. All results reported below are for the memoryless implementation unless explicitly specified.

2.4.1 Datasets

We apply our simplification algorithm to four datasets. Table 2.1 lists the size of the datasets. Isosurfaces of the original and several simplified versions of the data can

be seen in Figures 2.10- 2.14. The first two are MRI scan data from the Chapel Hill

dataset	#vertices	#tetrahedra
`brain`	51,772	285,768
`head`	70,262	391,608
`turbine`	126,976	714,420
`hydrogen`	32,768	178,746
`ribosome`	512,000	2,958,234

Table 2.1: List of datasets used for evaluation.

volume rendering test data set, volume I. The third is density data of a turbine blade. The fourth contains electron density data for a hydrogen molecule and the fifth is cryo electron microscopy data of a ribosome. Each of the datasets is available to us for input as a tetrahedral mesh with function values specified at the mesh vertices and linearly interpolated within the mesh elements. We eliminate the effects of large scale differences between the spatial coordinates and the function values by normalizing the data within the unit hypercube in $\mathbb{R}^4$.

2.4.2 Tetrahedral Shape Improvement

The objective of the first experiment is to determine the effect of varying φ on the mesh quality. We do this by applying the simplification using various values of φ and computing the dihedral, solid, and face angles of the tetrahedra in the simplified meshes. The average values of the three types of angles remain almost constant at around $1.20, 0.49$, and 1.05 radians. We see in Figure 2.5 that the introduction of a positive value for φ sharpens the distribution of angles around their respective averages. Figure 2.6 provides visual evidence of the improvement in the tetrahedral shape quality. These results are for experiments run on the `hydrogen` dataset. The other datasets behave similarly.

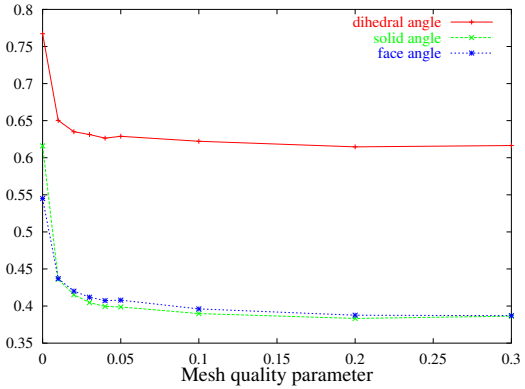

Figure 2.5: Graph of the standard deviation for the dihedral, solid and face angle measurements for various values of φ. Note the initial dip followed by almost no change.

2.4.3 Approximation Error vs Tetrahedral Shape

As mentioned earlier, we add weighted hyperplanes into the error quadric in the hope of improving the shape of tetrahedra in the mesh. However, adding these hyperplanes reduces the weight on the density map error that should be minimized for a good approximation. The objective of our second experiment is to study the effect of varying φ on the approximation error. Figure 2.7 shows the root mean square and max error for various values of φ, both of which increase for increasing φ, as expected.

2.4.4 Topology Preservation vs Tetrahedral Shape

In this experiment, we study the effect of varying φ on the smallest achievable vertex count. Violation of the link condition seems to require badly shaped tetrahedra, so we expect that we can reach smaller sizes if we increase φ. Figure 2.8 is a graph of the number of vertices in the smallest mesh reachable by the simplification algorithm

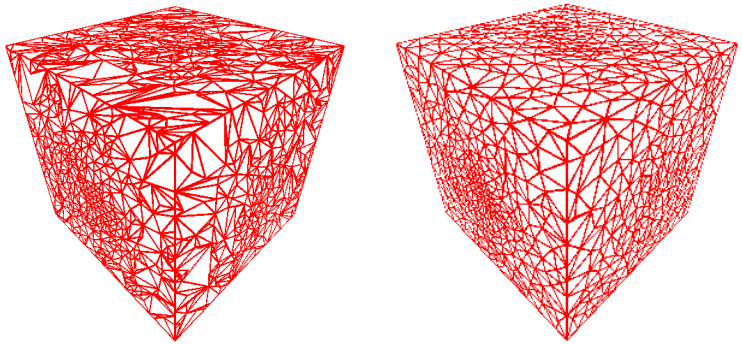

Figure 2.6: The simplified meshes of the `hydrogen` dataset obtained for $\varphi = 0.0$ (left) and $\varphi = 0.01$ (right). Note the dramatic improvement in the shape of the elements for a non-zero shape quality factor.

for various values of the parameter φ. Note the expected dip followed by an almost horizontal section. This is consistent with the earlier observation of a dramatic improvement in the shape quality of the mesh tetrahedra even with small values of φ followed by no significant change on further increasing φ.

2.4.5 Density Map Preservation

The results of the above experiments suggests we choose a mesh quality factor in the range where the graphs show significant improvement in the shape quality of mesh tetrahedra. We set $\varphi = 0.02$ and run the simplification algorithm on the five datasets. Figures 2.10- 2.14 displays a small sample of isosurface to provide a feeling for the effect the simplification has on the datasets. Significant artifacts begin to appear when the target vertex count drops to 10% or below. This is reflected in the root mean square and max errors shown in Tables 2.2 - 2.4. There is a sharp increase in error when the target vertex count drops below 10% for the `brain`, `head` and `turbine` datasets, whereas the `hydrogen` and `ribosome` datasets start degrading only below

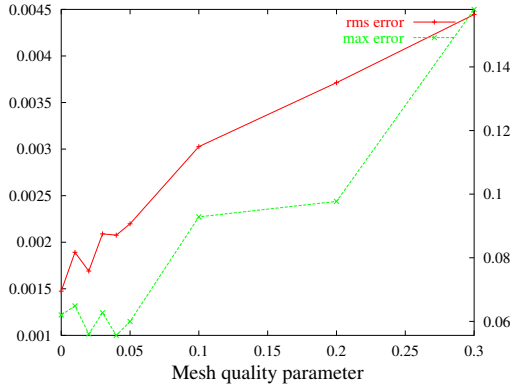

Figure 2.7: Graphs of the root mean square error (with values shown on the left) and maximum error (with values shown on the right) of the simplified meshes for various values of φ. After some erratic fluctuation the errors increase monotonically.

2%. The tables also show the time taken to perform the simplification.

`brain`

%	#vert	rms	max	time
100	51,772			
50	25,886	0.001	0.051	224
30	15,531	0.003	0.123	310
20	10,354	0.007	0.145	356
10	5,177	0.015	0.261	406
5	2,588	0.022	0.243	435
3	1,553	0.034	0.492	447
2	1,035	0.036	0.306	453
1	517	0.051	0.344	460

`head`

%	#vert	rms	max	time
100	70,262			
50	35,131	0.001	0.045	315
30	21,078	0.004	0.074	438
20	14,052	0.008	0.209	504
10	7,026	0.018	0.399	575
5	3,513	0.026	0.332	616
3	2,107	0.032	0.285	633
2	1,405	0.043	0.463	643
1	702	0.066	0.463	653

Table 2.2: The root mean square and max errors associated with each of the simplified meshes for the `brain` and `head` MRI datasets and the running time in seconds.

2.4.6 Critical Point Statistics

The topography of a density map is often expressed in terms of its critical points, which in the generic case are of one of four types: minima, 1-saddles, 2-saddles and

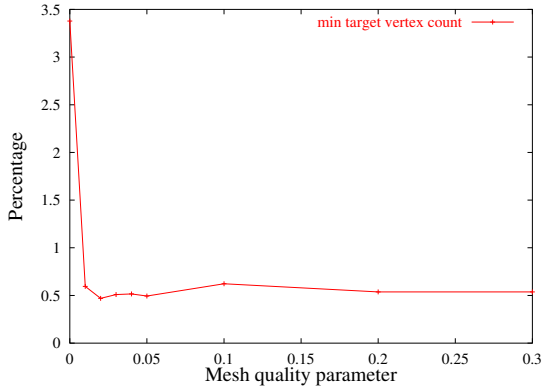

Figure 2.8: Graph of the number of vertices in the smallest mesh reachable by the algorithm. Similar to the mesh quality we observe a dramatic initial improvement followed by almost no change.

maxima. As defined in [28], the *lower link* of a vertex u is the subcomplex of the link induced by the vertices with smaller function value than u. Using reduced Betti numbers for measuring the connectivity of the lower link, we classify u as regular or critical. In the piecewise linear case, a critical vertex can have non-trivial multiplicity even in the generic case, which is reflected in our statistics shown in Figure 2.9. Contrary to our initial expectations, the simplification first increases the number of critical points before decreasing them. We explain this phenomenon by the temporary creation of spurious critical points in relatively flat regions of the distribution. The criticality of these vertices is based on small fluctuations of the density function. We substantiate this rationalization by measuring the importance of a critical point as the amount of change in function value necessary to turn it into a regular point. Formally, we compute the persistence of the critical vertices as defined in [31]. In the graphs of Figure 2.9, we reflect this information by ignoring critical points whose persistence is less than a threshold that increases from back to front. We see that a

`turbine`

%	#vert	rms	max	time
100	126,979			
50	63,488	0.000	0.008	606
30	38,092	0.001	0.053	825
20	25,395	0.002	0.087	938
10	12,698	0.010	0.275	1,058
5	6,348	0.022	0.364	1,126
3	3,809	0.032	0.602	1,155
2	2,539	0.035	0.602	1,170
1	1,269	0.069	0.692	1,187

`hydrogen`

%	#vert	rms	max	time
100	32,768			
50	16,384	0.000	0.010	135
30	9,830	0.001	0.059	187
20	6,553	0.002	0.069	213
10	3,276	0.005	0.092	242
5	1,638	0.009	0.132	257
3	983	0.013	0.132	264
2	655	0.018	0.189	267
1	327	0.026	0.174	271

Table 2.3: The root mean square and max errors associated with each of the simplified meshes for the `turbine` and `hydrogen` datasets and the running time in seconds.

`ribosome`

%	#vert	rms	max	time
100	512,000			
50	256,000	0.000	0.016	3,009
30	153,600	0.001	0.223	4,178
20	102,400	0.002	0.684	4,771
10	51,200	0.003	0.392	5,403
5	25,600	0.006	0.140	5,760
3	15,360	0.007	0.205	5,914
2	10,240	0.008	0.218	5,997
1	5,120	0.011	0.310	6,087

Table 2.4: The root mean square and max errors associated with each of the simplified meshes for the `ribosome` dataset and the running time in seconds.

very small threshold suffices to erode the gain in critical points caused in the initial simplification phase.

2.4.7 Sanity Checks

To ensure that the implementation does not have subtle flaws that create biases or other artifacts is always a challenge when working with non-trivial datasets. Typically, one looks for unusual behavior while testing the code against special data. In addition, we check the code by collecting evidence that the output is structurally correct. We perform low and high level structural checks of the mesh. At the lowest

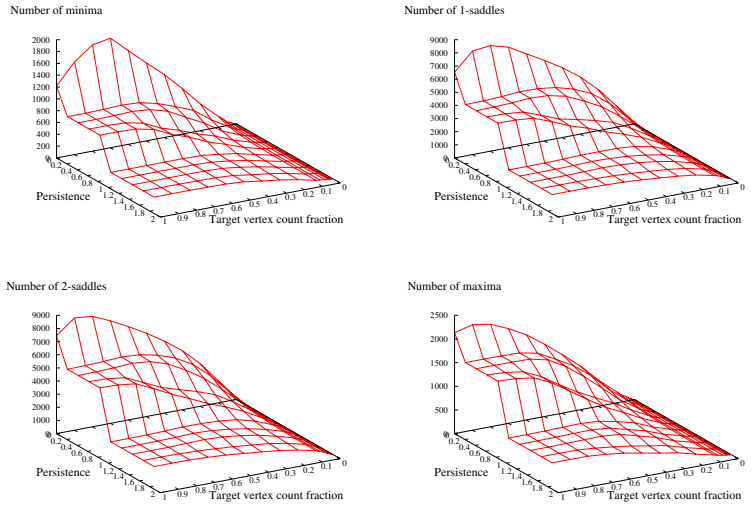

Figure 2.9: Graphs of the numbers of critical points in the `brain` data: The numbers change from left to right as we simplify the density and from back to front as we eliminate critical points of low persistence. The graphs for remaining datasets are similar.

level, we test whether the triangles in K are connected the right way. Details of such tests can be found in Mücke [69]. At a higher level, we compute the Euler characteristic: $\chi = s_0 - s_1 + s_2 - s_3$, where s_i is the number of i-simplices in K. The Euler characteristic of a 3-ball is 1, and since our algorithm maintains the topological type, it must be 1 throughout the process. There is a relation between the Euler characteristic and the critical points, namely $\chi = c_0 - c_1 + c_2 - c_3$, where c_0, c_1, c_2 and c_3 count the minima, 1-saddles, 2-saddles and maxima, respectively. The compactification that changes the 3-ball into the 3-sphere changes the Euler characteristic to 0. The alternating sum of critical points thus furnishes another test for the correctness of the simplified mesh.

2.4.8 Inclusion-Exclusion

We repeated all of the above experiments using the inclusion-exclusion method for accumulation of bisecting hyperplanes and compared the results with the memoryless method, finding possibly predictable differences in performance:

(i) the inclusion-exclusion method is faster than the memoryless method by a factor of about three;

(ii) the approximation error for the inclusion-exclusion method is marginally higher than that of the memoryless method;

(iii) the reachable limit of maximum simplification increases from about 0.5% for the memoryless method to about 3% for the inclusion-exclusion method.

In summary, the quality of the simplification is marginally worse for the inclusion-exclusion method, but the running time is somewhat better. We place more weight on the quality of the computed result than on speed and thus decided to present detailed experimental results only of the memoryless method of mesh quality improvement.

2.5 Discussion

We described an algorithm for simplifying a density function represented by a tetrahedral mesh of a three-dimensional geometric domain. The main ingredients of the algorithm are topology preserving edge contractions and quadratic cost functions that attempt to preserve the density map as well as improve the mesh quality. We performed various computational experiments to determine relationships between the parameters that control the algorithm. We ran our algorithm on five datasets and evaluated the results by computing the approximation error, some isosurfaces, and the number of critical points, all as variables depending on the amount of simplification. We conclude this chapter by mentioning a couple of future projects.

(1) Use the hierarchy of critical points to get a comparison between two similar density functions that is more qualitative than the approximation error computed in this work. Study the behavior of this comparison as the functions become progressively less similar.

(2) Compare the numerical method that maintains the mesh boundary using extra hyperplane constraints with a combinatorial method based on the general link condition. The latter method would preserve the face and edge structure of the mesh boundary and treat boundary vertices with higher priority.

The natural next step in simplifying a density function is a synthesis of geometric and topological methods, similar to the work of Bremer et al. [11] and Edelsbrunner et al. [30] for two-dimensional functions. This amounts to constructing the three-dimensional Morse-Smale complex [28] and simplifying it in a sequence of cancellations ordered by persistence [31]. We describe the construction of the Morse-Smale complex in three-dimensions in the next chapter and discuss the technical challenges involved in implementing a hierarchical version. An alternative would be to simplify the density function by re-prioritizing the edge contractions in our current algorithm to include topological information about the critical points. An advantage of the latter idea is that it is potentially easy to implement by extending the existing implementation.

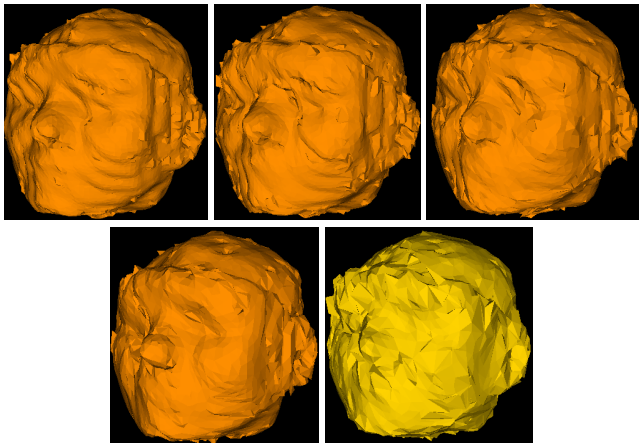

Figure 2.10: Top-left to bottom-right: isosurfaces extracted from the original `brain` dataset and after simplifying the mesh to 50%, 30%, 20% and 10% of its original size.

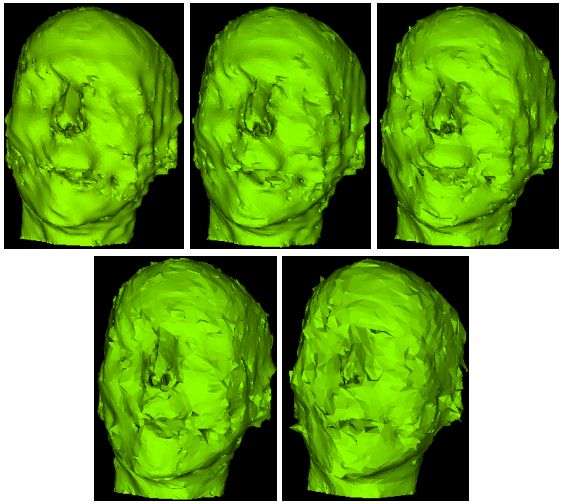

Figure 2.11: Top-left to bottom-right: isosurfaces extracted from the original `head` dataset and after simplifying the mesh to 50%, 30%, 20% and 10% of its original size.

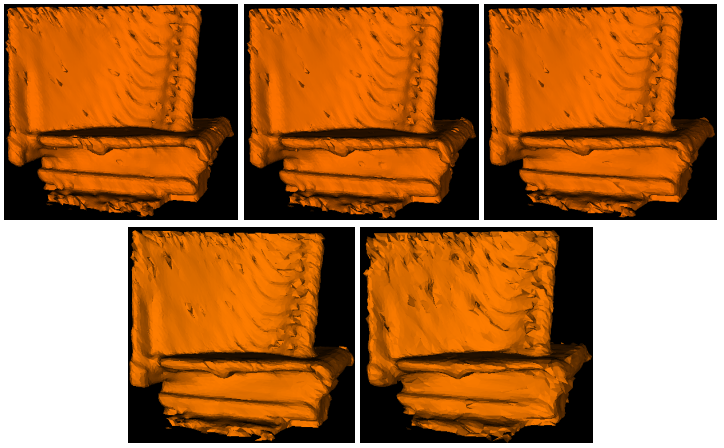

Figure 2.12: Top-left to bottom-right: isosurfaces extracted from the original `turbine` dataset and after simplifying the mesh to 50%, 30%, 20% and 10% of its original size.

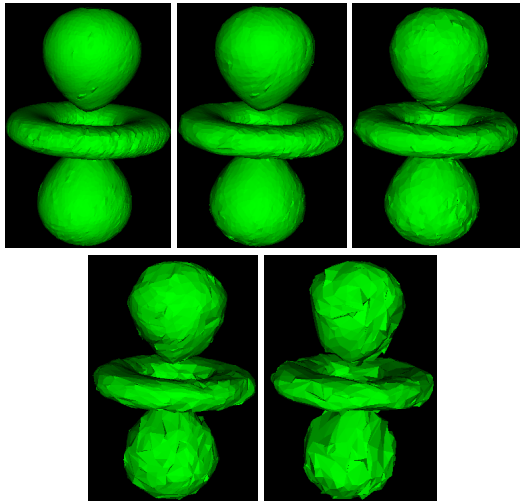

Figure 2.13: Top-left to bottom-right: isosurfaces extracted from the original `hydrogen` dataset and after simplifying the mesh to 50%, 30%, 10% and 5% of its original size.

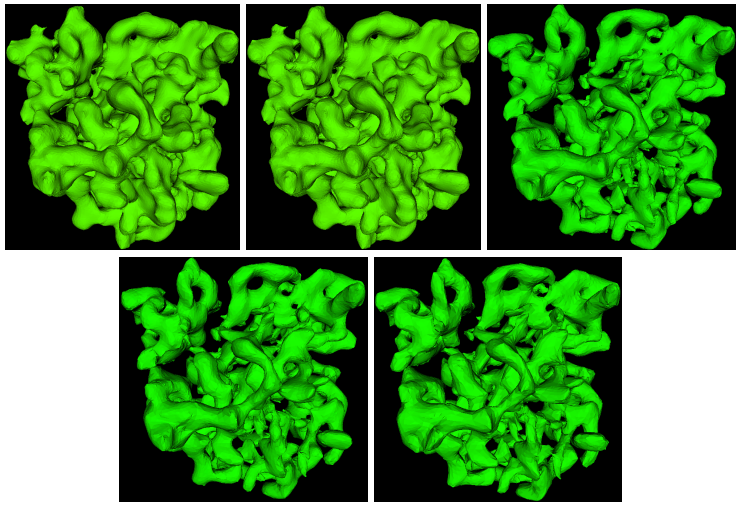

Figure 2.14: Top-left to bottom-right: isosurfaces extracted from the original `ribosome` dataset and after simplifying the mesh to 50%, 30%, 20% and 10% of its original size.

Chapter 3

3D Morse-Smale Complexes

We continue to study three-dimensional scalar functions in this chapter. However, we now focus on the extraction of topological features to present a global view of the function. In order to extract the topological features, we use results from Morse theory, which has been traditionally used for studying the topology of manifolds. Appendix A describes the basic terminology from Morse theory that we require for the discussion in this chapter. For a more comprehensive view, we refer to the books on this subject by Matsumoto [65] and Milnor [68]. The Morse-Smale complex of a Morse function partitions its domain into regions with uniform gradient flow. Such partitions were developed to study the behavior of dynamical systems. We extend Morse-Smale complexes to piecewise linear 3-manifolds and describe a combinatorial algorithm to compute them.

3.1 Introduction

3.1.1 Motivation

There is an abundance of natural phenomena that can be modeled by three-dimensional Morse functions. In oceanography, we study the distribution of temperature and other measurements over the Earth's oceans. In medical imaging, we reconstruct the inside of a living body from density distributions measured by MRI and other sensing technology. In x-ray crystallography, we determine the conformations of proteins and other molecules from electron densities derived from x-ray diffractions. In each case, essential information is obtained from variations of the density over the

space. Morse theory offers the basic mathematical language to reason qualitatively and quantitatively about this variation. In oceanography, we might be interested in the temperature extrema and how they change over time. In medical imaging, we use sharp changes in density to segment the body into bone, tissue and other constituents. In x-ray crystallography, we reconstruct geometric structure by following ridges connecting maxima in the electron density. Clearly, a systematic study of the variation in density (*i.e.* gradient of the density function) will be of significant use in understanding data obtained from the above mentioned scientific domains. Indeed, such studies have been conducted in the past.

3.1.2 Related work

The Morse-Smale complex captures the gradient flow characteristics of the scalar field by partitioning the space into regions of uniform flow. Thom is probably the first to formally develop a way of partitioning space using gradient flows [98]. One of the earliest work on such partitions that used Morse theory is that of Smale in the context of dynamical systems [88, 90]. He uses results from the study of these partitions to construct an elegant proof of the higher dimensional Poincaré conjecture [89, 91]. Besides the work done by Smale, there has been extensive study of such partitions in the smooth category [85]. All work on piecewise-linear manifolds have, however, been restricted to the two-dimensional case. Edelsbrunner et al. [30] define the Morse-Smale complex for piecewise-linear 2-manifolds by considering the PL function as the limit of a series of smooth functions and using the intuition to transport ideas from the smooth case. We follow the same approach in 3D. There are newer types of criticalities and hence new types of cells in the Morse-Smale complex when we move from 2D to 3D. This makes the extensions of the results from 2D a non-trivial task.

The two-dimensional piecewise-linear case has been studied extensively in different

fields, under different names, motivated by the need for an efficient data structure to store surface features. Cayley [15] and Maxwell [67] propose a subdivision of surfaces using peaks, pits, and saddles along with curves between them. Warntz uses such partitions to study surfaces arising in social sciences [105]. Pfaltz [76] proposed a graph based representation of the partition, called a *surface network*. Applications and extensions of surface networks have been studied within various fields including cartography [109], computer vision [57], and crystallography [54]. The development of different data structures for representing topographical features is discussed in a collection of expositions edited by Rana [78].

Three-dimensional density functions are commonly visualized by drawing one or several level sets. In three-dimensional Euclidean space, such a set is generically a 2-manifold, often referred to as an isosurface, which divides the space into inside and outside. The 1-parameter family of isosurfaces sweeps out each cell in the Morse-Smale complex in a predictable manner, starting at the minimum and proceeding towards the opposite maximum while crossing the boundary everywhere at a right angle. The most popular method for computing an isosurface is the marching cube algorithm, which assumes the density is given by its values at the vertices of a regular cubic grid [64]. Extensions and improvements of this algorithm can be found in [43, 55, 66, 71, 73, 103, 107, 111].

The marching cube algorithm visits the entire grid, which implies a running time proportional to the number of grid cells. A significant improvement in performance can be achieved by limiting the traversal to those cells that have a non-empty intersection with the constructed isosurface. Starting at a 'seed edge', the algorithm traverses the cells following the component of the isosurface as it is uncovered [6]. A minimal collection of seed edges that touches each component of every level set is provided by a minimal covering of the Reeb graph [79], stored for quick access in a

hierarchical data structure referred to as the contour tree [58]. The Reeb graph is a compressed representation of the components, but it has no geometric information related to the gradient flow as expressed by the Morse-Smale complex. Extensions and improvements of the original algorithm for constructing contour trees can be found in [14, 75, 96].

Another concept related to Morse-Smale complexes is the medial axis of a shape in three-dimensional Euclidean space. As introduced by Blum [10], it is the set of centers of spheres that touch the boundary of the shape in at least two points without crossing it. Medial axes are used in a wide variety of applications, including shape representation [21, 86], mesh generation [87], geometric modeling [94], motion planning [44], image processing [74] and computer vision [110]. If the boundary is an orientable 2-manifold embedded in three-dimensional Euclidean space, we may define the signed distance as a function over the space. The medial axis then consists of arcs and quadrangles in the Morse-Smale complex.

Partitions similar to the Morse-Smale complex have been computed earlier for vector fields as well but most of them employ numerical methods. Helman and Hesselink [50] classify the zeros of a vector field and perform particle tracing to compute the topology of the vector fields in two and three dimensions. Globus et al. [39] describe a software for visualizing the topology of three dimensional vector fields that also uses numerical methods to trace the flow lines.

3.1.3 Approach and Results

A fundamental difficulty in applying Morse theoretic ideas to scientific problems is the lack of smoothness in real data. We take a combinatorial approach to this problem by simulating smoothness to the extent necessary and then carry on the intuition from the smooth setting. One advantage of our approach as opposed to a numerical one is

the guarantees that we can provide about the consistency of the computed structures. We extend the definition of Morse-Smale complexes to the piecewise linear domain and describe it as an overlay of the descending and ascending manifolds. We also give a combinatorial algorithm for constructing it with guaranteed structural correctness. Finally, we describe our visualization tool for displaying sub-structures of the Morse-Smale complex. We use the visualizations to study dislocations in a copper crystal and the shape of biological macromolecules electron microscopy data.

3.2 Definition

The datasets that we work with are piecewise linear. So, we need to transport Morse theoretic ideas that were originally developed in the smooth setting into the piecewise linear domain. After giving definitions for the Morse-Smale complex for smooth 3-manifolds, we introduce quasi Morse-Smale complexes and discuss the artifacts that arise upon moving to the piecewise linear domain.

3.2.1 Smooth 3-Manifolds

Integral Lines. Given a Riemannian metric on $\mathbb{M}$ and a local coordinate system with orthonormal tangent vectors $\frac{\partial}{\partial x_i}(p)$, the *gradient* of f at p is

$$\nabla f(p) \;=\; \left[\frac{\partial f}{\partial x_1}(p), \frac{\partial f}{\partial x_2}(p), \frac{\partial f}{\partial x_3}(p)\right]^T.$$

It is the zero vector iff p is critical. An *integral line* $\gamma : \mathbb{R} \to \mathbb{M}$ is a maximal path whose velocity vectors agree with the gradient: $\frac{\partial \gamma}{\partial s}(s) = \nabla f(\gamma(s))$ for all $s \in \mathbb{R}$. Each integral line is open at both ends, and we call $\operatorname{org}\gamma = \lim_{s\to-\infty}\gamma(s)$ the *origin* and $\operatorname{dest}\gamma = \lim_{s\to\infty}\gamma(s)$ the *destination* of γ. Both are necessarily critical points of f. Integral lines are pairwise disjoint. We consider each critical point as an integral line

by itself, and with this stipulation the integral lines partition $\mathbb{M}$. We use them to decompose $\mathbb{M}$ into regions of similar flow patterns.

Ascending and Descending Manifolds. The *descending* and *ascending manifolds* of a critical point p are

$$\begin{aligned} D(p) &= \{p\} \cup \{x \in \mathbb{M} \mid x \in \operatorname{im}\gamma, \operatorname{dest}\gamma = p\}, \\ A(p) &= \{p\} \cup \{x \in \mathbb{M} \mid x \in \operatorname{im}\gamma, \operatorname{org}\gamma = p\}, \end{aligned}$$

where $\operatorname{im}\gamma$ is the image of the path γ on $\mathbb{M}$. If x and y are points different from p that belong to the descending and the ascending manifolds of p then $f(x) < f(p) < f(y)$. This implies that $D(p) \cap A(p) = p$. The descending manifolds of f are the ascending manifolds of $-f$ and, symmetrically, the ascending manifolds of f are the descending manifolds of $-f$. This implies that the two types of manifolds have the same structural properties. Specifically, the descending manifold of a critical point p of index i is an open cell of dimension $\dim D(p) = i$. Since the integral lines partition $\mathbb{M}$, so do the descending manifolds. Moreover, they form a complex as the boundary of every cell is the union of lower-dimensional cells that are its *faces*. The ascending manifolds form a dual complex: for critical points p and q of f, $\dim D(p) = 3 - \dim A(p)$, and $D(p)$ is a face of $D(q)$ iff $A(q)$ is a face of $A(p)$.

Morse-Smale Complex. A Morse function f is *Morse-Smale* if the descending and ascending manifolds intersect only transversally. Suppose $D(p)$ and $A(q)$ have non-empty common intersection. If $\dim D(p) = 2$ and $\dim A(q) = 1$ then the transversality assumption implies $D(p) \cap A(q) = p = q$. In the more interesting case in which both are 2-manifolds, $A(p)$ and $D(q)$ are faces of $A(q)$ and $D(p)$ and, as illustrated in Figure 3.1, the common intersection is a simple path connecting the two critical points. Following [30], we define the cells of the *Morse-Smale complex* as the components of the sets $D(p) \cap A(q)$, over all critical points p and q of f. By definition,

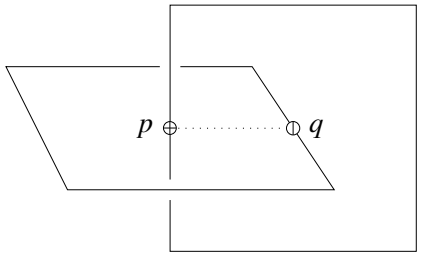

Figure 3.1: The dotted line is the common intersection of the descending 2-manifold of p and the ascending 2-manifold of q.

each cell of the Morse-Smale complex is a union of integral lines that all share the same origin q and the same destination p. The dimension of the cell is then the difference between the two indices. We call the cells of dimension 0 to 3 *nodes*, *arcs*, *quadrangles*, and *crystals*. Each two-dimensional cell is indeed a quadrangle, but its boundary may be glued to itself. The prototypical case of a crystal is a cube, which we imagine standing on its tip, but more interesting cases are possible as shown in Figure 3.2.

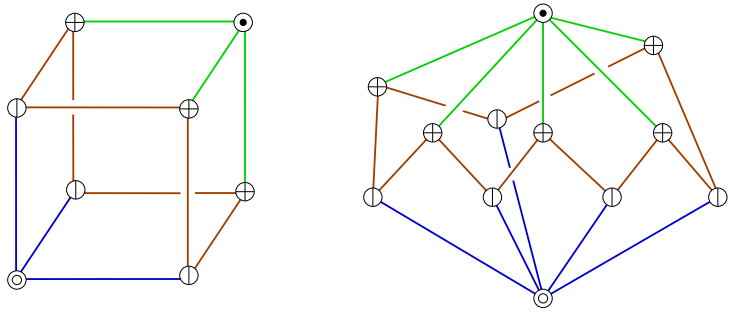

Figure 3.2: Crystals in the Morse-Smale Complex: The one on the left is the prototypical case in the shape of a cube and the one on the right is a more interesting case.

3.2.2 Piecewise Linear 3-Manifolds

Quasi Morse-Smale Complex. We construct a complex that is structurally indistinguishable from the Morse-Smale complex by taking open manifolds made up of simplices in K. It is a decomposition of space into crystals in which the boundary of each crystal is a quadrangulation. The function f has its critical points at the nodes of this complex and is monotonic within all the arcs, quadrangles and crystals. It differs from the Morse-Smale complex because the arcs and quadrangles may not be those of maximal ascent and descent. Let U, V, X and Y be the sets of minima, 1-saddles, 2-saddles and maxima of f, let R, S and T be the sets of arcs that connect minima to 1-saddles, 1-saddles to 2-saddles, and 2-saddles to maxima respectively, and let P and Q be the sets of quadrangles with nodes from U, V, X, V and V, X, Y, X in that order, respectively, around the boundary. We define a *quasi Morse-Smale complex* of f as a decomposition of $\mathbb{M}$ into open cells that satisfies the following properties:

(i) all nodes are from $U \cup V \cup X \cup Y$, all arcs are from $R \cup S \cup T$, and all quadrangles are from $P \cup Q$,

(ii) there are no critical points within the arcs, quadrangles and crystals, and

(iii) each arc in S is on the boundary of four quadrangles, which in a cyclic order alternate between P and Q.

Note that a quasi Morse-Smale complex can be split into complexes defined by U, P and Y, Q. These are complexes that are structurally indistinguishable from those of the descending and ascending manifolds.

Simulating disjointness. Integral lines are not well defined for piecewise linear manifolds. So, following [30], we construct monotonic curves and surfaces that never

cross. These curves and surfaces can merge together and fork later. When a curve or surface merges with another curve or surface, we pretend that they remain infinitesimally close to each other without crossing until they either fork or reach a common critical point.

3.3 Data Structures

We use two main data structures, one for the triangulation K of the 3-manifold $\mathbb{M}$ and the other for the Morse-Smale complex Q of the function $f : \mathbb{M} \to \mathbb{R}$. The two consist of various pieces and are connected to each other as shown in Figure 3.3.

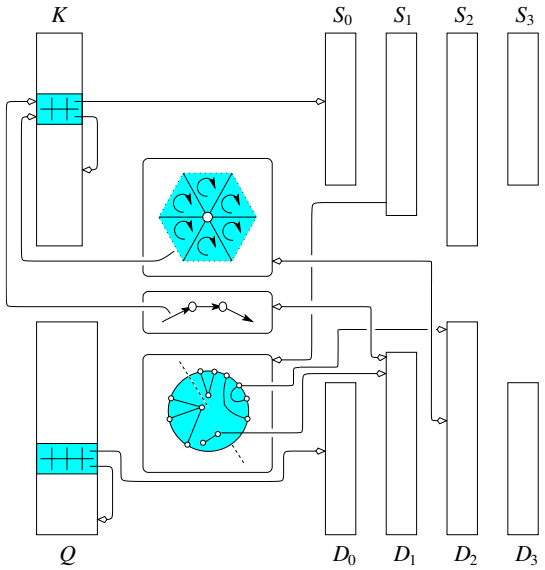

Figure 3.3: The data structures used to represent the triangulation and the Morse-Smale complex.

3.3.1 Triangulation

The triangulation of $\mathbb{M}$ is a 3-dimensional simplicial complex consisting of vertices, edges, triangles, and tetrahedra. Miscellaneous information about the simplices is stored in the arrays S_0, S_1, S_2, and S_3. Each simplex is identified by its dimension and its index in the corresponding array. The connectivity between the simplices is represented by another array, K, whose elements are sextets of pointers that connect the triangles in rings about the edges. This data structure is akin to the edge-facet structure introduced in [23]. As illustrated in Figure 3.4, a sextet is made up of the six ordered versions of a triangle.

The operation enext rotates the ordering of the three vertices cyclically by one position to the left. The operation sym exchanges the first two vertices in the ordering. Both move from one to another ordering in the same sextet, but enext moves within one orientation while sym changes between orientations. Using explicit pointers, a sextet supports the operation fnext, which moves from an ordered triangle abc to the next and similarly ordered triangle abd in the ring about the edge ab. Furthermore, it supports the operation org, which moves from abc to its origin, a. Note that an ordered triangle abc uniquely defines four ordered simplices, namely a, ab, abc, and $abcd$, and can therefore be interpreted as a vertex, an edge, a triangle, or a tetrahedron, whichever is appropriate or convenient.

To illustrate the functionality of this data structure, consider the computation of the link of a vertex $p = p_i$. Letting puv be one of the triangles that share that vertex, we use depth-first search to traverse all triangles in the star. For each visited triangle pxy, the edge xy belongs to the link of p and so do the triangles that precede and succeed pxy in the ring around xy. Given the initial triangle puv, the search takes time proportional to the number of edges in the link.

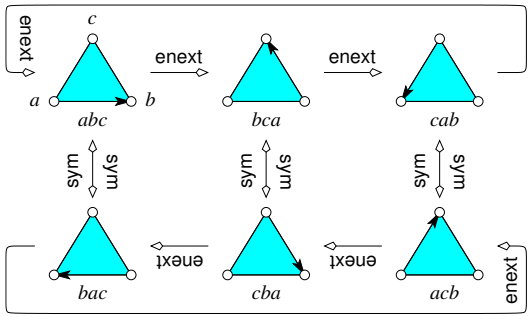

Figure 3.4: Algebra of ordered triangles.

With an additional test of the vertex heights, we can identify the lower link as a subcomplex of the link. We use the reduced Betti numbers of the lower link to classify the vertex p as regular, minimum, 1-saddle, 2-saddle, maximum or multiple saddle. The appendix describes the classification process in detail. We get the reduced Betti numbers by keeping track of the components in the lower link. If there are no components then $\tilde{\beta}_{-1} = 1$ and $\tilde{\beta}_k = 0$ for all $k \neq -1$, so p is a minimum. If the lower link is equal to the link then $\tilde{\beta}_2 = 1$ and $\tilde{\beta}_k = 0$ for all $k \neq 2$, so p is a maximum. Otherwise, $\tilde{\beta}_{-1} = \tilde{\beta}_2 = 0$ and $\tilde{\beta}_0$ is one less than the number of components. We get $\tilde{\beta}_1$ from $\tilde{\beta}_0$ and the Euler characteristic $\chi = s_0 - s_1 + s_2$, where s_k is the number of k-simplices in the lower link of p: $\tilde{\beta}_1 = \tilde{\beta}_0 + 1 - \chi$. p is regular if $\tilde{\beta}_0 = \tilde{\beta}_1 = 0$ and it is a multiple saddle combining $\tilde{\beta}_0$ 1-saddles and $\tilde{\beta}_1$ 2-saddles, otherwise.

3.3.2 Morse-Smale Complex

The Morse-Smale complex of f consists of simple open cells of dimensions 0, 1, 2, and 3, which we refer to as nodes, arcs, quadrangles, and crystals. Miscellaneous related information is stored in arrays D_0, D_1, D_2, and D_3, as shown in Figure 3.3.

Each cell is identified by its dimension and its index in the corresponding array. The connectivity is stored in another array, Q, whose elements are octets of pointers, as illustrated in Figure 3.5. Similar to ordered triangles, we have the operations anext

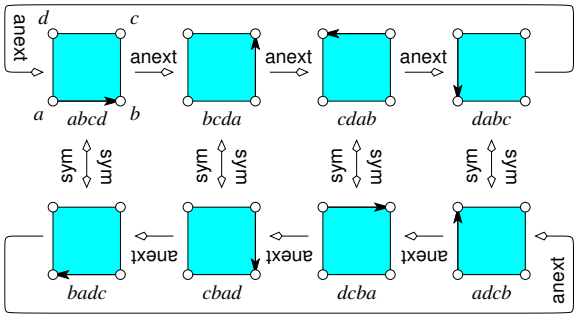

Figure 3.5: Algebra of ordered quadrangles.

and sym, which move between orderings in the same octet. The operation qnext takes us from $abcd$ to the next similarly ordered quadrangle $abef$ in the ring about the edge ab. Finally, the operation org takes us from $abcd$ to its origin, a.

A node in the Morse-Smale complex is just one of the vertices of the triangulation, but arcs and quadrangles are more complicated objects that need elaborate representations. Each arc is an open sequence of edges and, as indicated in Figure 3.3, it is stored as a sequence of oriented edges or pointers into K. Similarly, each quadrangle is an open patch of triangles and, as indicated in Figure 3.3, it is stored as a 2-manifold of oriented triangles or pointers into K. The connectivity can be recovered from the connectivity information in K, so we can get by with simple and compact representations. It is important that each arc and quadrangle has its own fixed orientation, which is then used in the algebra illustrated in Figure 3.5.

We have to be prepared to store partial complexes while constructing the Morse-Smale complex and to add new quadrangles and arcs. As a general policy, we add

a new cell when its description in terms of simplices in K is complete. Another complication is the possibility that cells may fold onto themselves and onto each other. We keep the data structure of the complex oblivious to such events. Instead, we use additional data structures to resolve such degeneracies. We describe these next.

3.3.3 Normal Structures

We say that arc or quadrangle *folds* onto itself if it contains a point of M multiple times. Similarly, arcs and quadrangles *coincide* if they share points of M. While permitting such events, we simulate an infinitesimal separation so that we can reason about sidedness and incidences in an unambiguous manner, as we will be able to without any additional data structures in the absence of folding. A *normal disk* captures the infinitesimal structure around an edge and a *normal interval* captures the infinitesimal structure normal to a triangle.

The normal structures support the decision making during the construction of the Morse-Smale complex. They are transient data structures that are used only while building the Morse-Smale complex. We list the operations supported by the normal structures, treating them as abstract data structures. Specific implementation with varying running times per operation can be found in standard algorithm texts [20].

A normal disk belongs to an edge in K and has the structure of a planar graph. There are points in the interior of the disk, called *base points*, that represent descending, ascending and intersection arcs containing the edge. There are also points on the disk boundary that represent how quadrangles containing the edge enter and exit the disks. The normal disk also contains between these points that represent the quadrangles containing the edge. A path may connect an interior point to a boundary point, two boundary points, or two interior points. The third case is a

degenerate situation where part of a quadrangle collapses onto an arc. The normal disk contains multiple paths if many quadrangles contain the edge. We require that all these paths meet the boundary of the normal disk at distinct points. An easy combinatorial argument shows that for each fixed k and l, there is a finite number of topological types for the set of paths in an normal disk with k base points and l paths. In fact, if we fix N points on the boundary of the normal disk and N_i directions ordered cyclically around each base point, with $2l = N + N_1 + \ldots + N_k$, then the family determines and is determined by a pairing of these points together with a placement of the base points that are not connected to the boundary by any path. However, not every pairing and/or placement can occur.

A difficulty in implementing the normal disk comes from the fact that the graph may have more than one component and these may be nested. The data type supports the following operations:

INSERT/DELETE a point (arc) or a path (quadrangle).

MERGE/SPLIT normal disks.

DECIDE whether a given path separates two components or two points.

A normal interval belongs to an ordered triangle in K and lists the ordered quadrangles that contain it. The data type supports the following operations:

INSERT/DELETE an ordered quadrangle.

MERGE/SPLIT normal intervals.

Note that the information stored in the normal interval of a triangles is duplicated in the normal disks of its edges. So, we need not store the normal intervals explicitly. However, we refer to them in our description for clarity reasons. Figure 3.6 illustrates folding events and the corresponding normal structures.

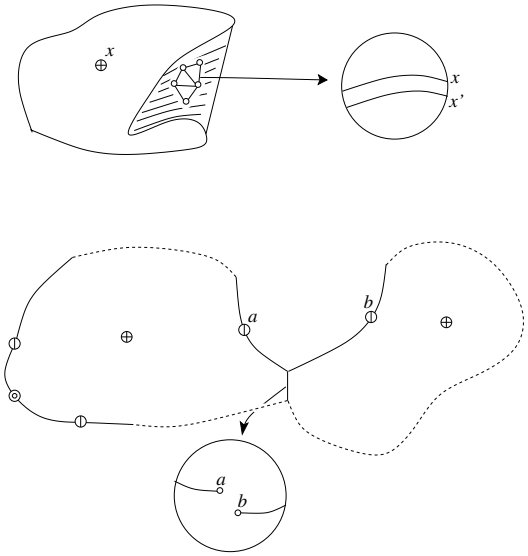

Figure 3.6: Top: a descending disk starting at a 2-saddle x that folds onto itself and the normal disk of an edge in the folded region. The two paths x and x' correspond to the oriented descending disks passing through the edge. Bottom: two descending disks merging along their boundary arcs. The normal disk of the shared edge has two vertices in the interior (base points) corresponding to the two arcs.

3.4 Algorithm

In this section, we first give an overview of the algorithm followed by detailed descriptions of how we construct descending and ascending manifolds.

3.4.1 Overview.

A quasi Morse-Smale complex is constructed during two sweeps over the 3-manifold. The first sweep is in the order of decreasing function value (height) and computes the descending manifolds. The second sweep is in the order of increasing height, which is the preferred order for computing the ascending manifolds. However, instead of

computing the two collections independently, we use the structure provided by the descending manifolds and add the ascending manifolds accordingly.

`Step 1.` Construct the complex formed by the descending manifolds.

`Step 2.` Construct the ascending manifolds in pieces inside the cells formed by the descending manifolds.

Some routing decisions in Step 1 require rudimentary structural information about the ascending 2-manifolds, so we compute that already in Step 1. We compute the intersections between the descending and the ascending 2-manifolds before we construct the latter. It is in fact easier to compute these intersections first and then widen them into the ascending 2-manifolds. An overriding goal is to get the structure of the quasi Morse-Smale complex right, and to achieve that goal we create various kinds of degeneracies and use the normal structures to help in resolving them when necessary. In order to streamline our description of the various steps in the algorithm, we denote the vertices of K by $p_1, p_2, \ldots, p_n$ assuming $f(p_1) > f(p_2) > \ldots > f(p_n)$.

Assuming a quasi Morse-Smale complex without any folding events, we claim a running time that is bounded from above by nlog(n) plus the input size. The nlog(n) term covers the time needed to sort the n vertices by height. The input size is the number of simplices in K. By choice of the data structure representing K, $\mathrm{Lk}\, p_i$ can be computed in time proportional to its size. Similarly, the classification of p_i, which reduces to counting the simplices and the components in the lower link, can be done in time proportional to that size. By definition, the size of the link is the number of simplices it contains, and because it is a two-dimensional sphere, this is $3t_i + 2$, where t_i is its number of triangles. Each triangle belongs to only two links, which

implies that the total size of all vertex links is

$$\sum_{i=1}^{n} 3t_i + 2 \;=\; 6t + 2n,$$

where n is the number of vertices and t is the number of triangles in K. As we will see later, the above time analysis applies to most steps taken by our algorithm. Indeed, we typically work inside a vertex link and compute simple sub-structures, such as shortest-path trees and circles separating oceans and continents from each other. We will see that with the assumption of unit length edges both tasks and miscellaneous others can be performed in time proportional to the size of the link and, in total, proportional to the size of K. The output size is the total number of simplices used to describe the complex. In the assumed absence of any folding events, the output size is less than the input size and can therefore be neglected in the running time analysis.

The assumption of no folding among the arcs and quadrangles in the quasi Morse-Smale complex is however not very realistic. Indeed, we do expect folding in practice and this is the reason why we have introduced the normal structures in the first place. A triangle can belong to arbitrarily many quadrangles and an edge can belong to arbitrarily many arcs. The output size is therefore no longer bounded from above by the input size and hence the running time of the algorithm is bounded from above by nlog(n) plus the input size plus the output size.

3.4.2 Descending Manifold Construction

We compute the descending 1- and 2-manifolds simultaneously during one sweep. To simplify the explanation of how this is done, we first discuss them separately, restricting our attention to simple critical points. One of the delicate parts of the construction is keeping track of degenerate situations and how they should be resolved. These

include both foldings and coincidences where descending and/or ascending manifolds overlap and need to be separated. As mentioned in the previous section, we deal with these degenerate situations with the help of special data structures, and we will explain how this is done as we describe the construction.

Descending 1-manifolds. Each descending 1-manifold is an open interval that belongs to and includes a 1-saddle $p = p_i$. It consists of two descending arcs and we call p the *root* of the 1-manifold and of its arcs. As illustrated in Figure 3.7, the 1-manifold descends from its root on both sides and, by simulation of the Morse-Smale condition, ends at minima of f. It is possible that the two arcs end at the same minimum, but because they do not contain that minimum, their union is still an open interval and not a closed circle. In the Morse-Smale case, all vertices of the

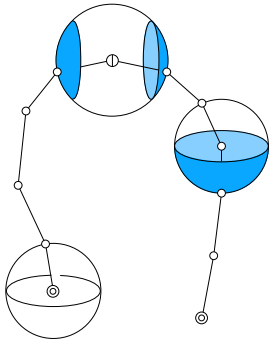

Figure 3.7: The descending 1-manifold rooted at a 1-saddle. The spheres sketch the links of the root, a regular point, and one of the two minima.

1-manifold except for its root are regular, but in the piecewise linear case it is also possible that the 1-manifold passes through another critical point p_j. We have $j > i$ because p_j is necessarily lower than the root. For an arc it makes little difference whether it passes through a regular or a critical point. However, since p_j starts its

own descending manifold, we need to make sure that the arcs descending from p_i and p_j are consistent in the sense of *simulated disjointness*. A consistent choice of edges will be automatic in the descending case because we extend each arc by adding the edge from the current endpoint to the lowest vertex in its lower link. This choice of extension implies, for example, that once two arcs merge, they go together until they both end at the same minimum. The structure can be much more subtle, however, if descending disks and other elements of the structure intersect these arcs as we will see later.

We distinguish between three operations in the construction of the descending 1-manifolds: starting, extending and gluing. The same three operations also occur in the construction of descending 2-manifolds, and they are processed within the same logical structure. The starting operation applies if p is a 1-saddle and starts the two arcs of the corresponding 1-manifold using edges from p to the lowest vertex in each ocean of the link. The extending operation continues all descending arcs ending at p by adding an edge from p to the lowest vertex in its lower link. An exception to this rule occurs if p is a 1-saddle. In this case, we will later start an ascending 2-manifold manifested in the link by a closed cycle in the continent that separates the two oceans. We then extend each descending arc to the lowest vertex in the specific ocean that is not separated from the arc by that ascending 2-manifold. The gluing operation applies if p is a minimum, which it declares a node of the Morse-Smale complex, and glues the descending arcs ending at p to each other.

Structure of a 2-manifold. The construction of the descending 2-manifolds is more complicated than that of descending 1-manifolds. We begin by discussing their structure and by formulating an invariant maintained by the algorithm. Each descending 2-manifold is an open disk that belongs to a 2-saddle, which we call its *root*. The disk descends from the root, which is its highest vertex. Its boundary is a circle

along which we alternately encounter 1-saddles and minima joined by descending arcs. This boundary circle might be partially glued to itself along one or more arcs. Note that this is fundamentally different from the case in which the disk folds onto itself: the folding can be simulated away since it does not happen for smooth functions, while the boundary gluing is an inherent feature of descending 2-manifolds. It is important that the descending 2-manifold does not contain its boundary, else it would not necessarily be a disk. In the most extreme case, the boundary circle is a single vertex so that the closure of the disk is a sphere.

Beyond being an open disk which descends from its root, we require that the restriction of f to the descending 2-manifold has no critical points other than the maximum at its root p. This property is guaranteed by an invariant maintained during the construction. At any moment, we have an open disk whose boundary is partially final or *frozen* and partially *unfrozen*. The frozen boundary grows from the empty set to a collection of *open* segments, which eventually merge to form a complete circle. The unfrozen boundary shrinks from a complete circle to a collection of *closed* segments, until it eventually disappears.

Disk Invariant. Let q be a vertex in the unfrozen portion of the boundary of a disk and let qu be an interior edge. Then u is either an interior vertex or a frozen boundary vertex, and $f(u) > f(q)$.

Note that the Disk Invariant prohibits interior edges that connect two unfrozen boundary vertices. This implies that as long as the entire boundary is unfrozen, there are no interior edges connecting two boundary vertices, and all edges descend from the interior to the boundary. Figure 3.8 illustrates the resulting structure of a descending disk. A regular vertex u in the restriction of f to the disk is characterized by a non-empty connected lower link. In other words, the edges in the star change

between descending from u to descending towards u and back again exactly once around u. The disk is extended at the highest unfrozen vertex q which either lies in

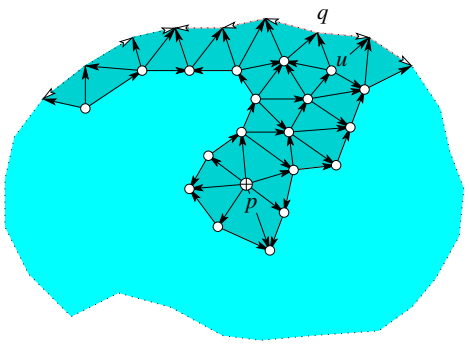

Figure 3.8: A portion of the triangulation of a partially constructed descending 2-manifold. The edges are oriented from the higher to the lower endpoints.

the interior of an unfrozen boundary segment or is the endpoint of a frozen boundary segment. In the former case, all interior edges descend towards q. In both cases we maintain the Disk Invariant by extending the disk such that all newly added edges descend from q. It follows that the only new interior vertex, which is q itself, is a regular point of f restricted to the disk.

Starting a 2-manifold. We start descending disks at 2-saddles and extend them at all unfrozen boundary vertices. Let $p = p_i$ be a 2-saddle, as shown in Figure 3.9, and let q be the lowest vertex in its link. By assumption, the lower link is a retract of the belt-like ocean around the link, and q belongs to that ocean. We start the corresponding descending disk by constructing a circle in the lower link, making sure that the circle contains q as one of its vertices. Even though we call it a circle, it may fold onto itself, and sometimes such folding is unavoidable. There are many ways to construct such a circle. We find a short circle using the shortest-path tree

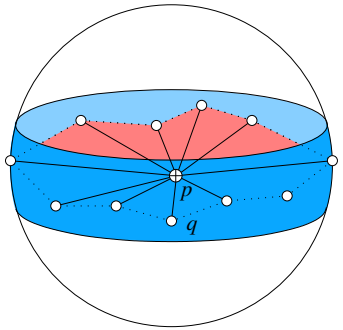

Figure 3.9: The disk rooted at p starts by connecting p to a circle in the belt-like ocean that passes through the lowest vertex q.

rooted at q that spans the lower link. Here we stipulate unit length edges so that shortest translates to minimum number of edges. After constructing the tree, we classify non-tree edges in the lower link depending on whether or not they divide the two continents. The circle is then defined by the dividing non-tree edge in the lower link whose two endpoints minimize the sum of distances to q. Returning to the classification, we note that the tree cuts the link open but keeps it connected. If we cut along a non-tree edge, we split the link into two disks. Consider the case where one of the disks contains both continents while the other is contained inside the ocean. The latter disk is triangulated and, by construction, its triangulation has all vertices on its boundary so it can be collapsed. We can therefore remove the triangles from this disk by repeated collapsing: at each step remove a triangle that has both edges on the boundary and declare the third edge a new boundary edge. The classification of non-tree edges in the lower link thus proceeds by repeated collapsing, which marks all non-dividing edges and leaves all dividing edges unmarked.

Extending a 2-manifold. In the non-degenerate case in which we are currently working, an interior vertex of a disk is typically a regular point of f, although it can

also be a 1-saddle or a 2-saddle. We first consider a regular point $p = p_i$ and assume it belongs to the boundary of a descending disk. Recall that we visit the vertices in the order of decreasing height, so p is the highest unfrozen boundary vertex. There are three cases, illustrated in 3.10. In the first case, p is adjacent to two unfrozen boundary edges and has two neighbors a and b, connected to p by unfrozen boundary edges. In the second case, there is only one unfrozen neighbor, c, with the other neighbor frozen. In the final case both neighbors are frozen, as are the edges, but p is not yet frozen. The algorithm treats the first two cases similarly and simultaneously.

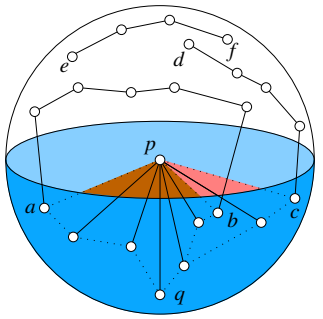

Figure 3.10: Three descending disks that touch p and intersect the link in a path each. One path starts and ends in the ocean, another starts in the continent and ends in the ocean, and yet another starts and ends in the continent.

Specifically, it constructs a shortest-path tree from the lowest vertex q in Lk_p. The points a and b belong to the lower link and are therefore vertices of the tree. We connect a to q along the unique path in the tree and extend the corresponding disk by connecting p to the edges of that path. We do the same for b and c and for all other vertices that are connected to p by unfrozen boundary edges. Note that no two paths cross one another, but it is possible that some paths fold onto each other and we must keep track of sidedness as before. In the third case where both neighbors of

p are frozen, the disk is developing a structure that we call a *spike*. The descending paths that enter at the points e and f, which may be the same, merge or continue merged as they exit from p to q. The disk is continued by imagining an infinitesimal strip that sits between the two merged paths. As the process continues from this point on this spike is then treated for lower vertices as a degenerate version of this third case, where both endpoints and the descending path all degenerate to a single point.

There is no essential difference in the computations if p is a 2-saddle, except that p itself starts an additional descending disk. By using the same tree for starting disks and for extending disks we avoid intersections, but as usual, folding onto themselves or each other is allowed. The case of a 1-saddle p is more interesting. If the two neighbors of p along the boundary of the disk are both unfrozen and belong to opposite polar oceans in the link then we do the same computations within both oceans. The point p remains on the boundary, but its two neighbors change to the vertices that are adjacent along the descending 1-manifold rooted at p. Before continuing, we declare p and the two incident boundary edges frozen for the descending disk. If the two neighbors of p along the boundary of the disk belong to the same ocean, then we proceed as we did for a regular point, working in this single ocean. In this case none of the new edges are frozen.

If either or both neighbors of p are frozen we have a special situation to consider. A frozen endpoint lies on a descending arc that will be continued parallel to one of the ends of the descending 1-manifold started at p. In order to make routing decisions in this case, we require the structure of the ascending disk that originates at p. We discuss this situation later.

Simulated disjointness. Before proceeding to the general case of a multiple-saddle with all the complications that this will bring, it is time to introduce some extra

structure to describe the singular behavior we have encountered so far.

First consider the coincidence of two descending arcs. One way this can happen is when we have a multiple-saddle. In this case a descending arc will be started for each ocean, and descending 1-manifolds will be described by a pairing of some of these arcs. When an arc is paired with more than one other, it will represent a coincidence. The other way that this happens among descending arcs is when one encounters another critical point with more than one ocean. Then it will be continued in parallel to one of the arcs started at this point, creating a coincidence. We can track coincidences by simply assigning a multiplicity to the edges they share. When descending disks meet these descending arcs, however, we need more precise information about which disk intersects which copy, etc.

The coincidence of a descending disk and a descending arc can occur in an essential way, when the descending arc is part of the boundary of the closure of the descending disk, or in an incidental or tangential way, when a descending arc contacts a descending manifold either at its root or at another point. In either case the arc always follows the steepest edge path and the descending disk always contains the lowest point in its link at each step of the construction, so once the contact is made the descending arc remains inside the disk until it terminates at one of the minima on the boundary of the disk. The simulation then "pushes" the arc off in the direction normal to the disk that it entered. When several disks contain a descending arc in their boundary, they are ordered consistently around it to prevent them intersecting. We store this information in the normal disks associated with edges in the arc.

When two descending arcs merge at a point p, their normal disks are joined along a segment of their boundary to make a new normal disk for the outgoing edge. Furthermore, their sets of paths are merged (sometimes a path is unchanged, sometimes two paths join to make a new single path). A coincidence also happens

when one descending disk contains the root of another or when two descending disks come together at some later point in their construction. Both cases show up in the process of extending a 2-manifold. After an extension step at a point p has been completed the various descending disks are ordered in normal intervals of the triangles that they pass through in $\mathrm{St}\, p$.

Simultaneous construction. As mentioned earlier, the descending arcs and disks are really constructed simultaneously, in a single sweep over the 3-manifold. Furthermore, instead of unfolding the multiple-saddles into simple ones, we work with general points where possibly $\tilde{\beta}_0 + \tilde{\beta}_1 \geq 2$. Notice that for both regular and critical points of all types, the link has $\tilde{\beta}_0 + 1$ oceans and $\tilde{\beta}_1 + 1$ continents. We process a vertex p in five steps:

`Step 1.1.` Start $\tilde{\beta}_1$ descending disks.

`Step 1.2.` Prepare $\tilde{\beta}_0$ ascending disks.

`Step 1.3.` Start $\tilde{\beta}_0$ descending 1-manifolds.

`Step 1.4.` Extend descending arcs that touch p.

`Step 1.5.` Extend descending disks that touch p.

Figures 3.19-3.23 illustrate these steps by showing the structural changes within the link of a vertex that has three oceans and three continents. The primary difficulty in the simultaneous construction is the coordination of the descending and ascending arcs and discs so that they all intersect in a locally and globally consistent manner that correctly simulates the structure we would expect for a smooth function. Keeping track of all this requires some terminology that we now introduce.

The initial structure at p. When we arrive at the vertex p during the construction of descending manifolds in the downward sweep, certain structures will already have

been created in its link. We restrict our attention to the structures contained in descending 1- and 2-manifolds that begin above and contain p. Other tangential structures lying on $\mathrm{Lk}\, p$ are handled either at an earlier or a later stage in the sweep.

A *D-point* is the vertex in $\mathrm{Lk}\, p$ where one or more descending arcs that pass through p intersect its link. A D-point x comes equipped with a normal structure, namely that of the edge xp. A *D-path* is the path of intersection between $\mathrm{Lk}\, p$ and a descending disk that contains p. If this intersection is closed then we refer to it as a *D-cycle*. When we arrive at p, each D-path has its vertices in a single continent except for its endpoints which either lie in an ocean or at a D-point. The normal structures give the incidence relationship between the D-paths and D-points. We distinguish between five types of D-paths:

(i) Both endpoints lie in a single ocean.

(ii) Each endpoint lies in a different ocean.

(iii) One endpoint lies in an ocean and one at a D-point.

(iv) Both endpoints lie on D-points.

(v) The entire path lies within a normal disk for a single D-point.

The final descending structure at p. Steps 1.1 to 1.6 will create the following structures at p: when $\tilde{\beta}_0 > 0$, a descending arc is begun for each ocean by adding an edge joining p and an appropriate vertex in the ocean. p now becomes a D-point for each such vertex in the oceans. The newly added edges will have associated normal disks whose base points and paths will be inserted as the rest of the structure is constructed. The descending 1-manifolds that begin at p are described by a set of $\tilde{\beta}_0$ pairs of these newly added edges with each edge appearing in at least one pair.

Each ocean O has a graph consisting of a shortest path tree rooted at the lowest point q plus $\beta_1(O)$ marked edges that along with edges in the tree make $\beta_1(O)$ cycles.

The inclusion of the graph into O is a homotopy equivalence. Hence, the graph is a union of $\beta_1(O)$ D-cycles, each obtained by following the two ends of a marked edge back to q. Each edge of the graph will have an associated normal interval, namely that of the triangle containing the edge and p, that stores the number of oriented times the D-cycles pass through it. Note that only one pass occurs through a marked edge.

Each continent C with $\beta_1(C) > 0$ has a graph consisting of a tree plus $\beta_1(C)$ marked edges that along with edges in the tree make $\beta_1(C)$ *A-cycles*. The tree is rooted at the highest point of its continent. Again each edge has a normal interval structure that has similar properties as the edges of graphs in the oceans.

The descending disks and arcs ending at p are extended into the oceans. p becomes a D-point for each vertex to which a descending arc is extended to. The descending disks meet $\mathrm{Lk}\, p$ in a collection of D-cycles and D-paths that are obtained by extending the D-paths in the initial structure at p. Each D-cycle passes through the lowest point of the unique ocean that it meets and each D-path either ends at the lowest points of the two oceans it connects, connects the lowest point in an ocean to a D-point in a continent, connects two D-points that lie in a single continent, or lies completely within a normal disk as a path connecting two base points. As always, each edge contained in a D-path or D-cycle has an associated normal interval representing the directions and multiplicity of the elements that pass through it. We now describe the five steps in the construction.

Steps 1.1 and 1.2: Constructing families of circles. In Steps 1.1 and 1.2, we start a family of descending disks and prepare the starting of the ascending family. The former are contained in the oceans and separate the continents, while the latter lie on the continents and separate the oceans, as illustrated in Figure 3.11. We extend the algorithm described earlier to construct the first family of circles in the oceans.

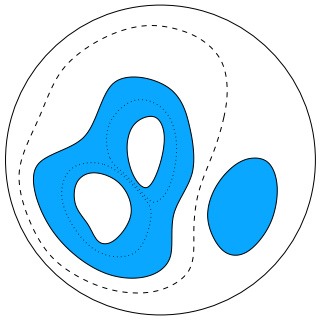

Figure 3.11: We draw $\tilde{\beta}_1 = 2$ (dotted) circles to separate the three continents and $\tilde{\beta}_0 = 1$ (dashed) circle to separate the two oceans. The descending disks that start at p intersect the link in the dotted circles, and the ascending disk intersects the link in the dashed circle.

We start with the tree consisting of shortest-paths from the lowest vertex in each ocean, and classify non-tree edges in the lower link depending on whether or not they divide the continents into two non-empty sets. Once we have selected a dividing edge, we add it to the tree that contains its endpoints, thereby making a cycle, and continue using collapses to eliminate edges that do not divide the continents. We repeat until we have added $\tilde{\beta}_1$ edges to the collection of trees. These edges define the $\tilde{\beta}_1$ cycles required in Step 1.1.

We repeat the same algorithm in $\mathrm{Lk}^+ p$ switching the roles of oceans and continents. This will give the $\tilde{\beta}_0$ circles required in Step 1.2. Note however that the construction of the second family is complicated by the presence of D-paths from the descending disks of higher vertices. We next describe these complications and how we cope with them.

Transversal intersections. Each relevant D-path intersects at most two oceans and one continent. Clip the path to the boundary of that continent and let x and y be the endpoints of the clipped path. The four possible cases are shown in Figure 3.12:

- both x and y lie in the continent (d_1);
- x lies in the continent and y lies on the boundary of the continent (d_2, d_3 and d_4);
- x and y lie on a common boundary component (d_5);
- x and y lie on different boundary components (d_6).

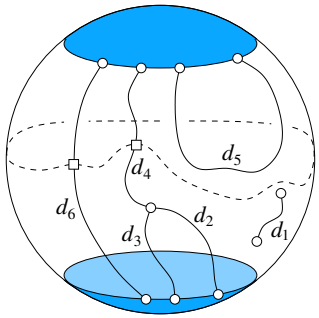

Figure 3.12: Some of the descending disks passing through p form barriers in our effort to draw circles preparing ascending disks within the continents. The squares are gateways at which the dashed circle may cross the paths.

We take precautions to ensure that the ascending disks either do not cross the D-paths or cross them minimally and transversally. In particular, if a D-path meets $\mathrm{Lk}\, p$ in two different oceans then it should cross every ascending disk, which is started by a circle separating the two oceans, exactly once. We cope with this difficulty by modifying the continents before building the circles.

Consider a single continent C, and let G be the graph made up of the edges of the D-paths that meet it. Cut C open along G in order to form a barrier that ensures transversal intersection between a D-path and a circle constructed in the continent. In practice, we create these barriers by duplicating all edges and most vertices of G that lie in C. A few vertices, called *gateways*, are not duplicated. These

gateways are potential points of intersection between the D-paths and circles in the continent. The two copies of a duplicated edge or vertex lie on different sides of the barrier and are connected to simplices in the star lying on the respective side. G could have vertices with degree greater that two. For example, the D-paths d_2, d_3 and d_4 in Figure 3.12 share a D-point. In this case, the vertex could possibly be split into multiple vertices partitioning the neighborhood of the vertex in C into multiple wedges. We now describe how we choose the gateways. The graph G consists of one or more connected components. Consider a component that consists of vertices with degree at most two. This could be an edge path of type (iii) or (iv) or two paths of type (ii) alongwith perhaps a sequence of paths of type (i) that together make a simple path connecting one or two components of the boundary of C. We choose the highest vertex of this graph component as a gateway.

Let V_3 denote the set of vertices in G that have degree at least three. The second type of graph components have at least one vertex from V_3. These components contain two types of paths: type I that connect the boundary of C to a vertex in V_3; type II that connect two vertices in V_3. Let there be k paths of type II in G. If all vertices in V_3 are made gateways then $\beta_1(C)$ increases by k. This is because splitting along a type II path introduces a loop whereas splitting along a type I path leaves $\beta_1(C)$ unchanged. If none of the vertices in V_3 are made gateways then C shatters into multiple regions. We need to choose an appropriate subset of V_3 as gateways such that no loops are introduced and C remains connected. Identifying this subset is easier if we construct a graph, $\tilde{G}$, representing the connectivity between the different regions of C. $\tilde{G}$ is constructed as follows: Add the set of vertices from V_3 and add a vertex for each region of C formed when none of the vertices in V_3 are gateways. Introduce an edge between a vertex associated with a region of C and a vertex from V_3 if the region contains the vertex. Our goal now reduces to finding a subgraph, $\tilde{T}$,

of $\tilde{G}$ that is homotopy equivalent to C and containing all vertices of $\tilde{G}$. The edges of $\tilde{G}$ that are not included in $\tilde{T}$ will correspond to the splits of vertices in V_3. If a vertex of V_3 is not split then it becomes a gateway. We choose all but one boundary component of C and follow the regions adjacent to each component, tracing edges of $\tilde{G}$ in this process. Include these edges into $\tilde{T}$ and then add enough edges to $\tilde{T}$ in order to ensure that all vertices of $\tilde{G}$ are included and no loops are introduced in $\tilde{T}$. Figure 3.13 illustrates the use of $\tilde{G}$ to determine the gateways.

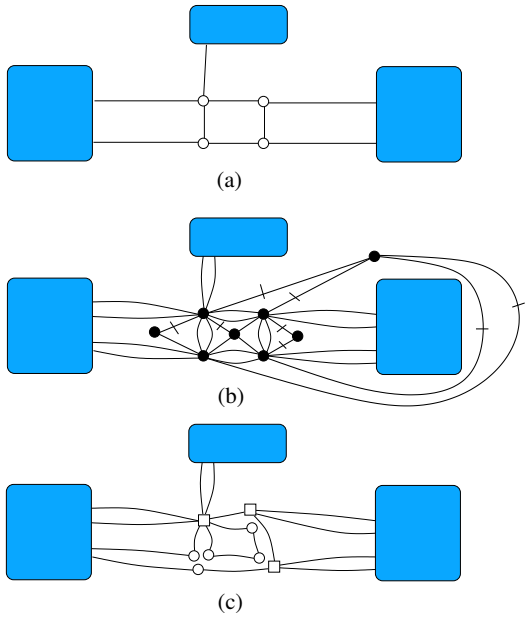

Figure 3.13: Creating barriers in continents to ensure minimal and transversal intersections between ascending and descending disks: an illustration for a continent that has three adjacent oceans. (a) Multiple descending disks could pass through the continent. (b) Compute graphs $\tilde{G}$ and $\tilde{T}$. Vertices of $\tilde{G}$ are shown as black disks and its edges are shown in bold. Edges belonging to $\tilde{T}$ are marked. (c) Split a vertex of V_3 if its incident edges are not included in $\tilde{T}$.

After creating the barriers in C, we can construct the shortest-path trees rooted at the highest point in the continent and route the circles as explained before. This completes steps 1.1 and 1.2. Note that in both cases we make additions to normal intervals and disks where appropriate to track the new elements. Notice that the A-cycles constructed by this procedure may pass through some of the D-points. This is in fact what typically happens at a vertex of G with degree greater than two. When this happens, the circles are routed through the normal disk so that they miss all the base points and intersect the paths inside the normal disk minimally. This routing will be part of what determines the pairings of descending arcs that make descending 1-manifolds.

Steps 1.3 and 1.4: Starting and extending descending arcs. When $\tilde{\beta}_0 > 0$, there is a descending 1-manifold dual to each ascending disk starting at p. We start a descending arc for each ocean by adding an edge from p to its lowest point. The descending 1-manifolds are then described by pairing these arcs. The A-cycles divide $\operatorname{Lk} p$ into regions, each containing a single ocean. The descending arcs for two oceans are then paired if and only if they are separated by a single A-cycle. A base point is added to the normal disk of each edge that constitutes a descending arc originating at p. This normal disk could contain a path representing a descending disk started at p into this ocean. The D-cycle corresponding to this descending disk divides $\operatorname{Lk} p$ into two components and hence separates the descending arcs paired with the one in its ocean into two sets. The path in the normal disk separates the base points in the same way as the D-cycle separates the descending arcs.

The ascending disks originating at p are also used to tell us how to extend descending arcs that pass through p. This is more subtle because the ascending disks can cut through the normal disk associated with a D-point. The descending arc is routed into an ocean without causing any intersections with ascending disks.

Step 1.5: Extending descending disks. We extend the procedure described earlier for extending descending disks to the case of multiple oceans. We start with a D-path that could be one of five types as listed earlier. In the first three types, atleast one end of the D-path lies in an ocean. We extend the end(s) lying in an ocean by a path to the lowest point in the ocean thereby creating a D-cycle. We also add a path in the normal disk of any D-point that the extension meets. Each such path in the normal disk will connect two boundary points and will not meet any base point. If the D-path crosses an A-cycle, then both ends are extended to lowest points in the appropriate oceans. This extension might result in the creation of a spike in which case a path connecting base points is added to the normal disk. Extending the fourth type of D-path restricts it to the normal disk of a D-point creating a spike. The ocean containing this D-point is not separated from the D-path by an A-cycle. We create a path between two new base points in the normal disk of this D-point. In the fifth and final case, the D-path connects two base points in the normal disk of a D-point representing a previously created spike. This spike is simply carried through.

3.4.3 Ascending manifold construction

We construct the ascending manifolds during a sweep of the 3-manifold in the direction of increasing function value.The construction is similar to that of the descending manifolds, except for the complications caused by the fact that the latter already exist. The added constraints are expressed in terms of barriers formed within vertex links. Conceptually, we first construct the ascending 1-manifolds by connecting 2-saddles to maxima and the intersection curves between the descending and ascending 2-manifolds by connecting 1-saddles that lie on the boundary of a descending 2-manifold to its source 2-saddle. The intersection curves and ascending arcs trace the boundary of the quadrangles that constitute the ascending 2-manifolds. These

quadrangles can now be filled in one at a time. The algorithms performs all these constructions simultaneously.

As with descending manifolds, let's first understand the constructions one at a time in the case where all critical points are non-degenerate. We will also suppress the discussion of normal disks and intervals for now.

Intersection curves and ascending arcs. Recall that for a Morse-Smale function on a 3-manifold, the intersection between a descending disk D and an ascending disk A is either empty or is a curve connecting their roots. From D's point of view, the curve starts at a 1-saddle on its boundary and increases strictly monotonically until it ends at its root. The Disk Invariant maintained during the construction of the descending disks implies that the restriction of f to D has no critical points other than the maximum at its root. To construct the curve, we thus start at the 1-saddle and repeatedly extend the path by connecting its endpoint to the highest adjacent vertex in the triangulation of D. The curves starting from various 1-saddles on the boundary may meet but never cross and eventually end at the root of D. Two curves from different descending disks may also meet, but these cases are resolved along with the descending disks by simulation of an infinitesimal separation given by the normal structures.

We start the two arcs of an ascending 1-manifold at every 2-saddle. The algorithm is similar to the one for descending 1-manifolds, except that we now avoid crossing any of the already established descending disks. Consider the D-cycle constructed in the ocean for starting the descending disk from a 2-saddle p. This D-cycle consists of a tree along with two shortest paths from end points of a special edge e to the root of the tree. Furthermore, each descending disk that contains p is extended by adding the path in this tree to its root, thereby forming a new D-cycle or extending the D-path. In particular, no such D-cycle or D-path contains the edge e.

Consider the connected components of $\mathrm{Lk}\, p$ that are cut out by the D-cycles and D-paths created while extending the descending disks that pass through p, not including the D-cycle corresponding to the dist starting at p. We will call the closures of these components as the *slabs* of p. Since none of the D-cycles and D-paths contains e, the slab that contains e meets both continents. We start the two ascending arcs by connecting p to the highest point in each continent that lies in this slab, making no intersection with the disks descending from above. If either or both of these points lie on the boundary of a slab, then the ascending arc is being constructed within one or more descending disks and the choice of which way to route the arc will be determined by the normal disk.

Ascending disks. The lowest point of a quadrangle on an ascending disk is the 1-saddle at which the disk is rooted. This 1-saddle is connected to 2-saddles by two continuous intersection curves emanating from the 1-saddle, and the 2-saddles are connected to a common maximum along ascending arcs. We construct the individual quadrangles, which then fit together to form the ascending disk. Each quadrangle is constructed in a process similar to the one for descending disks. In this case, the frozen part of the boundary occurs when the boundary of the quadrangle meets either an intersection curve or an ascending arc. Edges and vertices on these curves and arcs are frozen, except for the vertices where we transition from frozen to unfrozen edges, which are considered unfrozen. The process also maintains the analogous Disk Invariant with the inequality reversed.

Let $p = p_i$ be a 1-saddle. The ascending disk at p has already been prepared during the descending sweep. It meets the continent in a circle, which is cut into a collection of segments by the descending disks that pass through p. The collection of triangles containing p and edges from each such segment from the initial portion of a quadrangle. The endpoints of these segments lie on intersection curves. To

discuss the extension of an ascending quadrangle, suppose that p is the lowest point on its unfrozen boundary. In this case, the picture is similar to that of Figure 3.10 with the roles of oceans and continents reversed *i.e.* the two points adjacent to p on the boundary of the quadrangle either both lie in the continent of p or one lies in the continent and the other is frozen in the ocean. These points all lie in a single slab. (This is in fact why we work with quadrangles!) In each case, connect the end points in the continent to the highest point in the slab that contains them, using a shortest path tree that spans the slab and is rooted at its highest point. With this background, we are now ready to describe the full construction.

Simultaneous construction. We actually construct the intersection curves, ascending arcs and ascending quadrangles in a single pass from bottom to top without explicitly splitting multiple-saddles into simple ones. As before, we consider a general point $p = p_i$, which is not a maximum or a minimum and has $\tilde{\beta}_0 + 1$ oceans and $\tilde{\beta}_1 + 1$ continents and describe the construction when we reach p during the sweep. We process p in six steps:

`Step 2.1.` Start $\tilde{\beta}_1$ ascending 1-manifolds.

`Step 2.2.` Start $\tilde{\beta}_0$ ascending disks.

`Step 2.3.` Start intersection curves.

`Step 2.4.` Extend ascending arcs that touch p.

`Step 2.5.` Extend intersection curves that touch p.

`Step 2.6.` Extend ascending quadrangles that touch p.

These steps are illustrated in Figures 3.24-3.26 for the example that we considered earlier during the descending manifold construction.

The initial structure at p. Just as for the descending pass, when we arrive at the point p during the construction of the ascending manifolds, certain structures will

already have been created in its link. As before we only care about those structures that contain p (see Figure 3.24).

The initial structure at p contains all the elements of the final descending structure. Additionally, it contains three new elements: ascending arcs, ascending disks and intersections curves that have come from below p. The ascending arcs show up as points, called *A-points*, in the oceans of p. The ascending disks interesect $\mathrm{Lk}\, p$ along paths whose interior vertices lie in the oceans and endpoints lie either in a continent or on an ascending arc in an ocean. We call these paths *A-paths*. The intersection curves appear as single points in $\mathrm{Lk}\, p$ where A-paths intersect D-paths and D-cycles. Figure 3.24 shows an ascending arc, depicted as a square box, lying on the A-path that contains y.

Points describing ascending arcs and intersection curves have associated normal disks. These normal disks contain more information than those encountered in the descending sweep because the disks now store both descending and ascending arcs and disks. First of all, there are two types of base points, those for descending arcs and those for ascending ones. Secondly, there are two types of paths, those that describe descending disks and those that describe ascending disks. All of these meet the boundary of the normal disk in distinct points. Finally, the two types of paths intersect minimally and only at points that correspond to intersection curves.

Steps 2.1 and 2.2: Starting ascending arcs and ascending disks. We start an ascending arc for each continent by adding an edge from p to the highest point in the continent, and create a normal disk corresponding to this edge. These arcs are paired, by duality, with descending disks starting at p. Ascending disks starting at p were prepared during the descending sweep. We now start constructing them by adding in the triangles. In Figure 3.25 ascending arcs are started at the points labeled 7, 8 and 9. Ascending disks are described by the dashed lines in the figure.

Step 2.3: Starting intersection curves. Start new intersection curves by adding the edge from p to points where the A-cycles that start ascending disks intersect D-cycles or D-paths. At this stage of the construction, all of these points lie in continents of p and particularly at a gateway. When this gateway is a D-point or A-point, the intersection curve is represented by an intersection of an ascending path and a descending path in the normal disk. If there are more than one of these in the normal disk, then we start an intersection curve for each. When the gateway does not lie at one of these points, i.e. when it was chosen as the highest vertex of a path connecting two oceans and did not coincide with a D-point or an A-point, then we start a new normal disk and store the intersection pattern in it.

Step 2.4: Extending ascending arcs. Each ascending arc that touches p enters into its link in an ocean and has an associated normal disk. The paths in this normal disk together with the D-paths and D-cycles lying within the oceans divide the link into slabs. We continue each ascending arc by adding an edge from p to the highest point in the slab that contains its entry point and the new edge inherits the normal disk. If a single normal disk contains more than one A-point and these are routed to different A-points in the continents, then we split the normal disk along the paths that separates them and the different edges now inherit the individual sections of the normal disk.

Steps 2.5 and 2.6: Extending intersection curves and ascending disks. We first describe the extension of ascending disks. An A-path can be of five possible types similar to the D-paths while extending descending disks. In the cases where atleast one endpoint lies in a continent, we connect it to the continent's highest point using the shortest path tree that was constructed while preparing ascending disks. During the process of extension, we avoid intersections within normal disks except at gateways. In the cases where one endpoint is an oceanic A-point, it is

possible that the A-path intersects a D-cycle or D-path. In order to avoid this illegal intersection, it is necessary to add a spike starting from the oceanic A-point and following the ascending arc henceforth. For example, the A-path ending at y in Figure 3.26 intersects a D-cycle. One of its endpoints is extended to 8 and a spike is added towards 7 to make the intersection legal. When the entire A-path lies within a normal disk of an A-point, it is carried on along with its associated ascending arc. No additional work needs to be done for the extension of intersection curves. We recognize and extend them along with the ascending disks.

3.5 Experiments

In this section, we report our experimental results and discuss possible ways of using Morse-Smale complexes for the analysis of various datasets. We begin with a description of the datasets and then discuss their visualization using the Morse-Smale complex and isosurfaces.

3.5.1 Datasets

We use synthetic data to illustrate the structure of the Morse-Smale complex and demonstrate their use in visualizing and analyzing scientific data from two different application domains: cryo electron microscopy (cryo-EM) and crystal lattice dislocation studies. All of our data is available as a tetrahedral mesh of the domain with function values specified at vertices. We add a dummy vertex at infinity along with new tetrahedra that have the boundary simplices and the dummy vertex as faces. This ensures that the input to our algorithm is a 3-manifold. We do this just for convenience and to avoid treating the boundary simplices as a special case in each step of the algorithm. Table 3.1 lists the datasets along with their respective sizes. `4pdist` and `3mindist` are synthetic datasets constructed by sampling analytic

functions within a unit cube. For the `4pdist` dataset, the function is the product of distances from four fixed points lying in the interior of the unit cube. The `3mindist` dataset is constructed using a function defined as the minimum distance from three fixed points.

dataset	#vertices	#tetrahedra
`4pdist`	1,728	7,986
`3mindist`	5,832	29,478
`dislocation`	32,768	178,746
`dnaB`	125,000	705,894
`dnaB-dnaC`	125,000	705,894

Table 3.1: List of datasets used for evaluation.

Dislocation data. The `dislocation` dataset contains results from atomistic simulations of dislocations in a crystal lattice. Atomistic simulation have been very useful for studying the formation of complex junction structures in metals undergoing work hardening [12]. Ductile metals bend under tensile stress by performing lateral motions between planes of atoms. Studies of copper crystals reveal a large number of mobile dislocations flowing through the solid that eventually collide with one another to form permanent rigid junctions. If the junction density is sufficiently high, fractures begin to appear because the solid can no longer bend through the dislocation motion. The `dislocation` dataset is available as a potential energy map over a 3D FCC lattice.

Cryo-EM data. Two of our datasets contain cryo-EM images of macromolecules. These datasets are freely available from the macromolecular structure database maintained by the European Bioinformatics Institute [24]. Cryo electron microscopy is a technique used to determine the structure of biological molecules where the sample is suspended in vitreous ice for imaging. The vitrified samples are transferred to an electron microscope where they are imaged using low electron doses. The cryo-EM datasets are available to us as density maps over a cubic grid. Higher density regions

in the data correspond to the biological molecule whereas the lower density regions represent the solution. Cryo-EM datasets are considered to have medium resolution as opposed to high resolution data available from x-ray crystallography. The latter provides atomic resolution data but is significantly more time consuming to generate. We use two datasets: the first is a DNA helicase called `dnaB` and the second is that of a complex [1] between `dnaB` and another protein called `dnaC`. DNA helicase is an enzyme that unravels the DNA double helix and breaks the hydrogen bonds. So, it plays a critical role in the replication process. `dnaB` is the major replicative helicase among the 12 present in E-coli. `dnaC` is also a protein that plays a critical role in the replication of DNA. It delivers `dnaB` to the site of action on the DNA template by first binding with DNA in solution to form a complex. The structure of `dnaB` and `dnaB-dnaC` were first studied by a reconstruction from cryo-EM images of frozen and hydrated molecules [83].

3.5.2 Visualization

The Morse-Smale complex has a rich structure in the presence of numerous topological features and hence visualizing it presents the same problems as in the display of the raw data, namely that of visual clutter. However, sub-structures of the Morse-Smale complex can be displayed individually and efficiently. We now describe the different sub-structures that can be visualized and demonstrate their use in analyzing our datasets.

Ascending/Descending arcs. The ascending (descending) arcs we compute are piecewise linear curves between 2-saddles (1-saddles) and maxima (minima, respectively) passing through edges of the input mesh. It is difficult to perceive depth from

[1] The term complex is used in this paper for two different entities. One is a mathematical structure, namely the Morse-Smale complex. The other is a biological entity, the `dnaB-dnaC` complex, that denotes a molecule made up of a `dnaB` and a `dnaC` molecule. The context determines which one we refer to.

a typical presentation of the arcs as a set of line segments and so we display them as thin tubes. Different descending arcs might merge at regular vertices but once they do, all of them remain together till they terminate at a minimum. This merging property can be easily seen for the descending arcs in the `4pdist` dataset (see Figure 3.14). In most datasets, there are certain regions that are more interesting than others. These regions could be in Euclidean or function space. We provide some filters that allow the user to display arcs lying within such regions. For example, the user can clip the arcs to those that originate from saddles with function values in a given range. A threshold arc length can be chosen to clip away short arcs that typically represent less significant topological features. The user can also prune away the arcs that span a short interval in the function space. This is akin to smoothing of the function to remove minor perturbations. There is an interesting correspon-

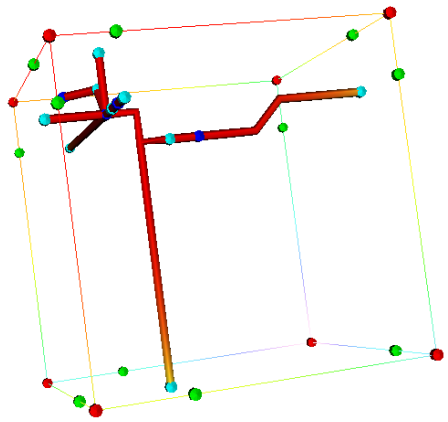

Figure 3.14: Ascending and descending arcs computed for the `4pdist` dataset: The descending arcs are represented as cylinders. All of them begin at the 1-saddles (cyan) and end at the four minima (blue). Some of them merge after which they stay together till their destination. The ascending arcs are represented as thin lines and they trace the edges of the cube beginning at 2-saddles (green) and ending at maxima (red).

dence between isosurface components and the filtered ascending (descending) arcs. Each isosurface component encloses at least one local maximum (minimum) and if the ascending (descending) arcs are filtered at the corresponding isovalue then we are left with the skeletal shape of the isosurface. We found it useful to work with this skeletal shape while exploring the data. For example, we locate interesting isosurfaces by progressively filtering the arcs and finally extracting one isosurface instead of multiple isosurfaces.

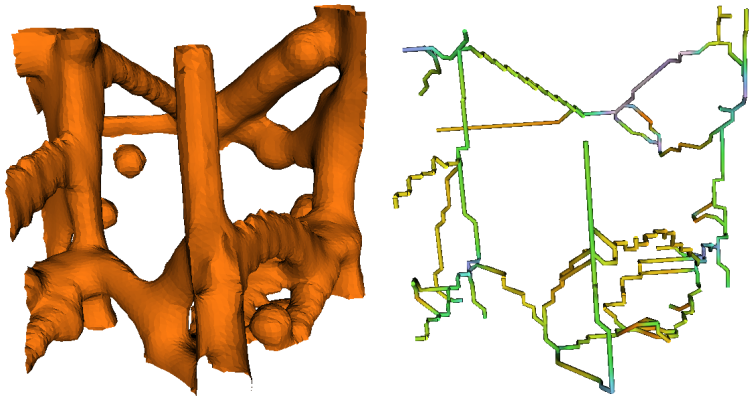

Figure 3.15: Atomistic simulations of dislocations emerging in a copper crystal under stress. The simulation does not have any geometric model of the dislocations. Topological analysis can extract explicit representation of the fractures created in the crystal structure. Left: an isosurface extracted from the dataset. Right: an explicit representation of the fractures in terms of ascending arcs.

This density map in the `dislocation` dataset is from the final time step in the atomistic simulation when fractures have appeared in the crystal. However, there is no explicit representation available for these fractures. The ascending arcs in the `dislocation` data abstract the fractures propagating through the copper crystal. This explicit representation enables further quantitative analysis of the fracture. For example, the number of connected components, length, and spatial extent of the

fracture can be computed.

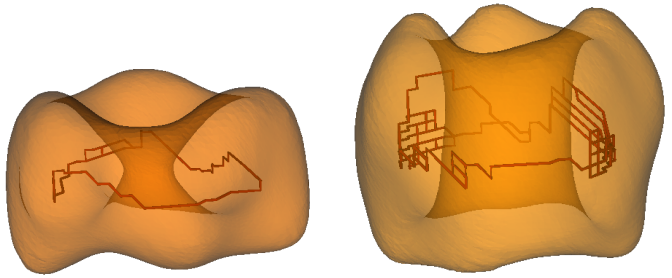

Figure 3.16: Isosurfaces (rendered translucent) extracted from the `dnaB` (left) and `dnaB-dnaC` complex (right) datasets along with the ascending arcs clipped to values above the isovalue. The ascending arcs trace the ring-like shape in both datasets. The `dnaC` molecule attaches itself to one face of the `dnaB` molecule. This is reflected in the isosurface extracted from the `dnaB-dnaC` complex but the tunnel is retained in the complex as can be seen from the ascending arcs.

`dnaB` has a ring shaped hexamer structure. The six subunits do not have the same size. Three of them are significantly larger than the remaining. The two kinds of subunits alternate around the ring. Also, the two faces of the ring-like molecules are different one having threefold symmetry and the other having a sixfold symmetry. The shape of the `dnaB-dnaC` complex is also ring-like, similar to that of `dnaB`. Each one of the six subunits of `dnaC` interacts with two subunits of `dnaB` and vice-versa. So, the `dnaC` molecule sits on the face of `dnaB` that has sixfold symmetry. The shapes of these molecules were originally determined by visually inspecting different isosurfaces and determining an appropriate isovalue. Extracting and displaying multiple isosurfaces is a time consuming task. Instead, we progressively applied our filters on the ascending arcs in a much faster process and observed that the ring-like structure of the DNA helicase is persistent over a sizeable range of density values. Figure 3.16 shows the corresponding isosurface. Figure 3.17 shows how tunnels in the isosurface

correspond to loops in the ascending arcs and when the tunnel opens up then so does the loop.

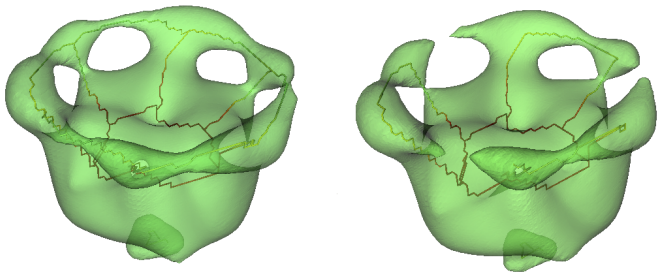

Figure 3.17: Two isosurfaces and ascending arcs clipped at the corresponding isovalue for the `dnaB` dataset. Note how the loops in the ascending arcs break along with the isosurface. Possibly interesting isosurfaces can be located by exploring the data using their skeletal shape traced by the ascending arcs.

Descending arcs exhibit similar behavior in datasets where local minima capture important features. We envision the ascending and descending 1-manifolds as visual aids for the exploration of volumetric data. Note that this complements existing approaches like the contour spectrum [8]. The skeletal shape can be generated for isovalue ranges determined from the contour spectrum.

Ascending/Descending disks. The ascending (descending) disks we compute are piecewise linear surfaces passing through the triangles of the given mesh. These are indeed displayed as triangle meshes that are possibly translucent to show occluded regions. The boundary of each ascending (descending) disk consists of a collection of ascending (descending) arcs and their corresponding nodes. The tube representation of arcs can be used to highlight the boundary of the disks. Figure 3.18 shows an ascending disk originating from one of the saddles in the `3mindist` dataset. Note the two descending arcs starting from this 1-saddle dual to the ascending disk. The disk

is actually a collection of quadrangles from the Morse-Smale complex, namely the ones containing the above 1-saddle and neighboring maxima as nodes. The ascending/descending disks can be used to annotate particular isosurfaces as well. This can be done efficiently by computing the intersection of the disks with an isosurface as an isocontour over the disk.

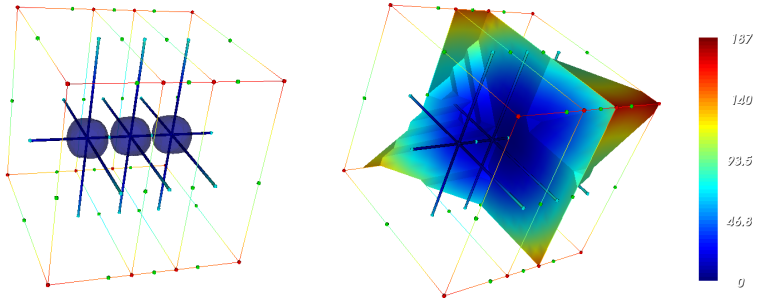

Figure 3.18: Visualizations of a function defined as the minimum distance from a set of three points $(0.25, 0.5, 0.5)$, $(0.5, 0.5, 0.5)$, and $(0.75, 0.5, 0.5)$ within a unit cube. Assume a minimum at infinity connected to all points on the boundary. The critical points are displayed as small spheres in blue, cyan, green, and red for minima, 1-saddles, 2-saddles, and maxima respectively. The edges in the interior are the descending arcs going between 1-saddles and minima. The ascending arcs are on the boundary connecting 2-saddles to maxima. The figure on the left shows an isosurface with three components about to merge at two 1-saddles. The figure on the right shows an ascending disk originating at one of these 1-saddles.

Critical points. The nodes of the Morse-Smale complex represent the critical points (singularities) of the function. They contain information about the local features in the data. We display the four types of critical points as color coded spheres. Figure 3.18 shows the critical points in `3mindist`. There are 3 minima (the fourth is at infinity), 16 maxima lying along the edges of the cube, 16 1-saddles, and 28 2-saddles. Clearly the number of critical points can be substantial even for such a simple dataset.

3.6 Discussion

This chapter introduces the Morse-Smale complex for a function over a 3-manifold as a decomposition of the 3-manifold into crystals with quadrangular faces. It also gives an algorithm to construct a quasi Morse-Smale complex for a piecewise linear function that guarantees structural correctness. Letting n be the number of vertices in the input triangulation, the running time is proportional to $n \log n$ plus the size of the input triangulation plus the total size of the output.

Various interesting issues remain open. We can transform the quasi Morse-Smale complex into the Morse-Smale complex by applying a sequence of operations called handle slides. As described for 2-manifolds in [30], using this approach we obtain a Morse-Smale complex that is numerically as accurate as the local rerouting operations used to control handle slides. For 3-manifolds, it is unclear how to find and order the handle slides that bring us closer to the Morse-Smale complex. In the previous section, we note the presence of substantial number of critical points even for simple datasets. Therefore, it is clearly useful to have a hierarchical representation of the Morse-Smale complex while working with large data sets. In the past, numerical methods [48, 61, 63, 99, 100] have been used on vector fields to generate simplified models of the flow topology introduced by Helman and Hesselink [50]. These methods typically cluster the critical points based on the given filtering parameter and use one critical point to represent each cluster. This approach is prone to computational errors at various stages of the simplification, which is the reason we prefer to follow the approach of Edelsbrunner et al. [30] and Bremer et al. [11] to get a combinatorial algorithm for computing the hierarchy followed by a numerical algorithm to construct the monotonic regions of the simplified Morse-Smale complex. The hierarchy is created by performing a sequence of cancellations of pairs of critical points ordered

by their topological persistence [31]. Extending this cancellation procedure and its geometric realization to 3D is a challenging problem.

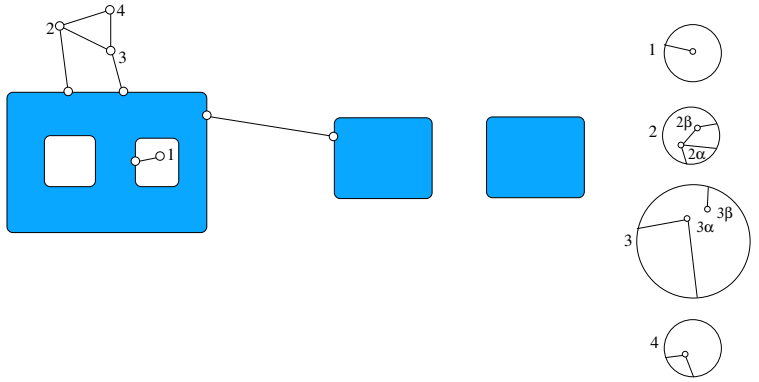

Figure 3.19: Initial structure when the algorithm reaches p in the downward sweep: The figure shows $\mathrm{Lk}\, p$ with three oceans and three continents. One of the continents is shown as the unbounded exterior. Seven D-paths and four D-points are present in the continents. The D-points are numbered 1, 2, 3, 4 and their normal disks are shown on the right. The base points in the normal disks represent descending arcs passing through.

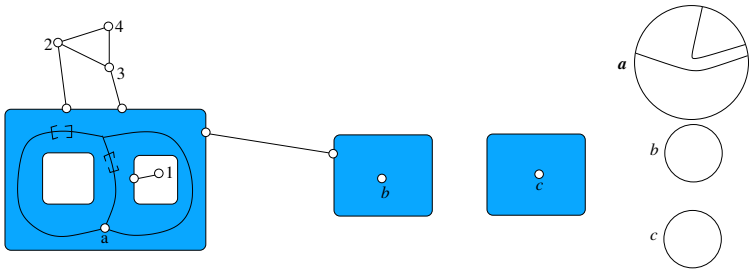

Figure 3.20: Starting descending disks and arcs: Two circles are constructed passing through the lowest point a in the ocean. The two marked edges are added to the shortest path tree to form the circles. The normal disk attached to the D-point a has two paths representing the two disks. Three descending arcs are started as edges to the lowest point in each ocean. These arcs are not yet paired to determine the descending 1-manifolds.

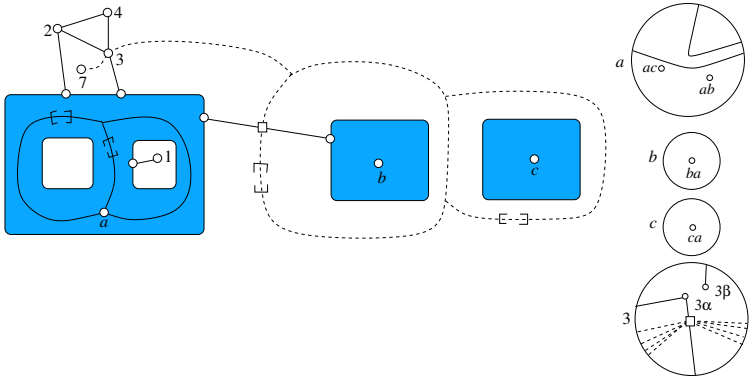

Figure 3.21: Preparing ascending disks: Two circles (dashed) are constructed in the continent to separate the three oceans. These circles intersect existing D-paths at gateways (shown as squares). The two marked edges are added to the shortest path tree rooted at the highest vertex, 7, in the continent to form the circles. Base points are added in the normal disks of a, b, c and labelled to represent the pairing og descending arcs. Both dashed circles pass through the D-point 3 twice and hence show up as four paths in its normal disk.

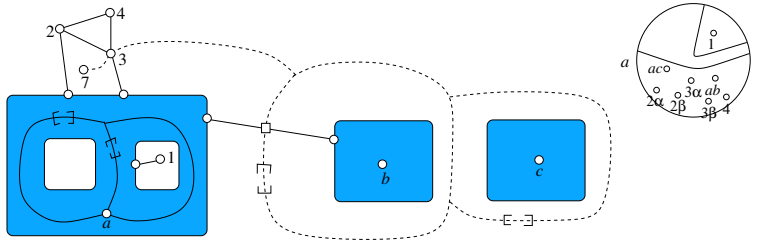

Figure 3.22: Extending descending arcs: Descending arcs ending at p and passing through D-points $2, 3, 4$ are extended to the lowest point a in the adjacent ocean. Note that extension of these arcs to b or c will cause an illegal intersection with an ascending disk.

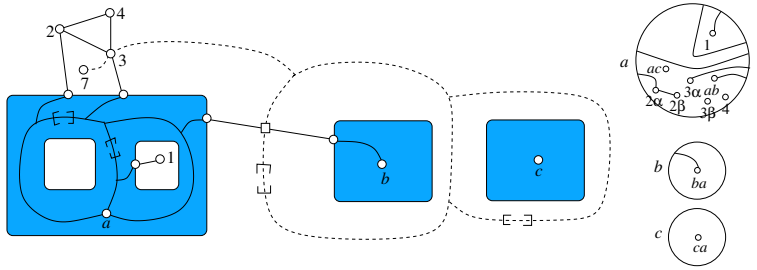

Figure 3.23: Extending descending disks: All D-paths are extended towards the lowest point in the ocean by following the shortest path in the tree. Paths corresponding to these extensions are added to the normal disks of a and b. Note that the path between 2α and 2β is carried through to a.

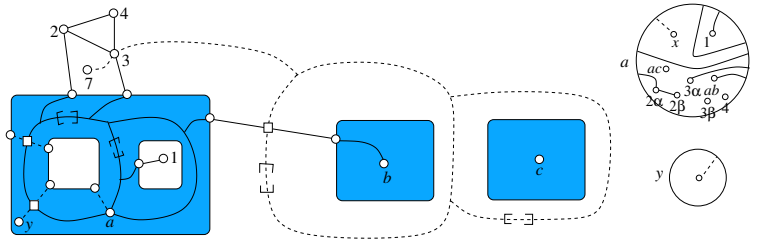

Figure 3.24: Initial structure when algorithm reaches p in the upwards sweep: Three A-paths and two A-points (y and x) are present in the oceans. Note that x is actually a base point lying in the normal disk of a.

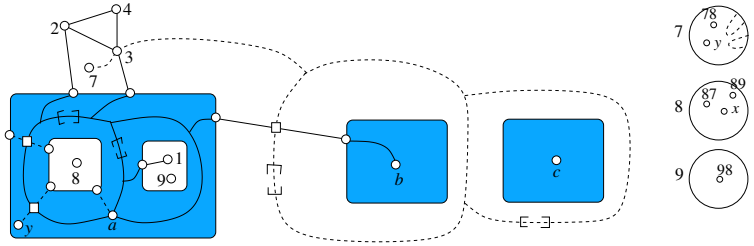

Figure 3.25: Starting and extending ascending arcs: Three ascending arcs are started towards the highest vertex in each continent $(7, 8, 9)$. The two continents corresponding to paired arcs are separated by one of the circles in the ocean. The ascending arcs ending at p are extended to a continent without causing any illegal intersections.

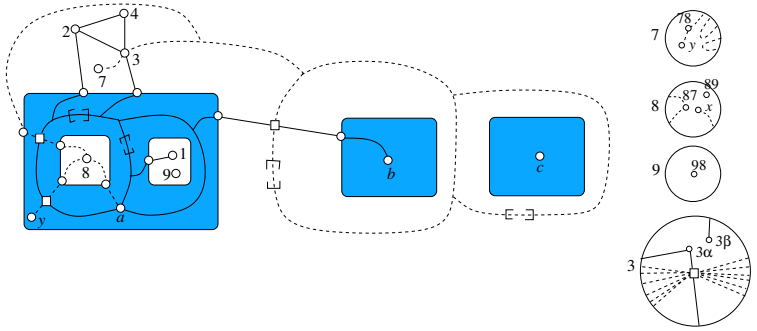

Figure 3.26: Extending ascending disks: The three A-paths are extended towards the highest vertices in the appropriate continents using the path to the root in the shortest path tree. This introduces new paths in the normal disks of $7, 8$, and 3. A spike is introduced to prevent an illegal intersection between one of the descending disks starting at p and the ascending disk represented by the A-path between y and 8. This spike is represented within the normal disk of 7 by the path between base points 78 and y.

Chapter 4

Scalar Function Comparison

In this chapter, we introduce a measure for comparing scalar functions defined over a common domain. We begin the chapter by explaining the use of such a comparison measure. We then define our measure, give algorithms to compute it and describe how we use it in our visualization software for studying scientific datasets. We end the chapter with some ideas for future work.

4.1 Introduction

4.1.1 Motivation

Scientists try to understand physical phenomena by studying the relationship between multiple functions measured over a region of interest. For example, the temperature, volume, pressure distribution etc. can be used to predict the flow behavior of a fluid. Some of these functions may be redundant *i.e.* two or more functions have similar behavior and hence it is enough to study one of them. The redundant functions may be filtered out provided they can be detected, say by using a measure that compares local similarities. Another application of such a measure would be in the study of time-varying functions. For example, distributions of hydrogen, oxygen, and other gases are measured at various time steps during a combustion process. Ideally, we would want fine discretization of the time scale when there are interesting changes in the function and coarse discretization otherwise. A graph plot of a measure that compares functions at successive time steps can help identify interesting time steps and hence guide the level of discretization. Another example of a time-varying

function is when we have data from multiple computations over the period of a few years when the software is being modified and different versions of the code used to produce the output. A comparison measure between outputs from successive versions of the code could identify significant changes made to the software.

4.1.2 Approach and Results

Given k scalar functions, $F = (f_1, f_2, \ldots, f_k)$, defined over a common d-manifold, we introduce new local and global comparison measures that compare the gradients of the functions. The average value of the local measure over the d-manifold specifies a global comparison measure $\kappa(F)$. For the case when $k = d = 2$, we describe alternative formulations of the measure in terms of the Jacobi set [27], which consists of the critical points of one function restricted to the isocontours of the other. We also describe a visualization tool for exploring datasets that contain multiple, possibly time-varying, function. We illustrate the local and global measures by applying them to both synthetic and scientific datasets.

4.1.3 Related Work

We compare our measure to two concepts: the correlation coefficient and Earth mover's distance. The correlation coefficient is a standard and popular statistical measure used to determine if two sets of values are linearly related by comparing their deviations from the respective mean values [33, Chapter 8]. When two functions are sampled at discrete points, the correlation coefficient is computed as

$$\varrho \;=\; \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{(\sum_{i=1}^{n}(x_i - \bar{x})^2) \cdot (\sum_{i=1}^{n}(y_i - \bar{y})^2)}}$$

where x_i, y_i are the corresponding values of the two functions and $\bar{x}, \bar{y}$ are the mean values of the two functions. Two functions have a high correlation coefficient if they

deviate consistently from their respective mean values *i.e.* if one function takes a value close to its mean then so does the other function at the same point on the domain. However, this measure is unsuitable to identify local similarities, because connectivity information is not recorded in its computation. The definition for ϱ can be extended to multiple functions in a straightforward manner. Note that the correlation coefficient is a global measure and there is no notion of a local comparison of the functions as compared to our measure. Our measure instead uses the gradients for comparing the local variation of the functions and uses it to define a global measure as well. The correlation coefficient can determine if the functions co-vary positively or negatively. If the functions are positively correlated over some regions of the domain and negatively correlated over other regions, then the correlation coefficient attains a value near zero and the functions are reported as independent. Our measure is not able to differentiate between positive and negative co-variation.

The correlation coefficient gives equal weight to all points on the domain whereas in many applications we want to match the features of the functions alone. These features are typically specified by the critical points. The *Earth mover's distance*, introduced by Rubner et al., computes the distance between the critical points of the two functions under a special metric and is therefore a measure that compares only the features. Given two sets of critical points, one is considered as a mass of earth spread in space and the other as a collection of holes in that same space. The Earth mover's distance measures the least amount of work needed to fill the holes with earth. A unit of work corresponds to transporting a unit of earth by unit ground distance. The ground distance measurement depends on the problem at hand. Originally introduced for comparing color patterns [82], it has been since used in modified forms for a variety of applications like contour matching [40], object tracking [108], polyhedral shape matching [95], and for comparing vector functions [9, 60]. Nearness under the

Earth mover's distance does not require similar features to be present within the same region of the domain. Therefore, it does not capture local similarity the way our measure does.

4.2 The Measure

Consider k scalar functions defined over a d-manifold: $F = (f_1, \ldots, f_k) : \mathbb{M}^d \to \mathbb{R}^k$. We first introduce k-forms and then use them to define a comparison measure between the k functions. For a more detailed introduction to k-forms, we refer to the book by Weintraub [106].

4.2.1 k-Forms

The expression

$$\mathrm{d}f_i \;=\; \frac{\partial f_1}{\partial x_1}\mathrm{d}x_1 + \frac{\partial f_1}{\partial x_2}\mathrm{d}x_2 + \cdots + \frac{\partial f_1}{\partial x_d}\mathrm{d}x_d$$

is called a *differential 1-form* (or simply a 1-form) in d variables. The 1-form is very similar to a vector field. In fact, we can set up a correspondence between the two such that the gradient of f_i, ∇f_i, corresponds to $\mathrm{d}f_i$. The notation is powerful because we can now talk about higher degree forms, which also have corresponding vector fields but these are more cumbersome to describe. A *wedge product* $\mathrm{d}f_1 \wedge \mathrm{d}f_2$ between two 1-forms gives a 2-form. The wedge product can thus be used to generate a *k-form*:

$$\mathrm{d}f_1 \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k.$$

This k-form can be written in terms of the basis of the associated $\binom{d}{k}$-dimensional vector space:

$$\mathrm{d}f_1 \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k \;=\; \sum_{1 \le i_1 < \cdots < i_k \le d} \omega_{i_1,\ldots,i_k}\; \mathrm{d}x_{i_1} \wedge \ldots \wedge \mathrm{d}x_{i_k}$$

$$\text{where } \omega_{i_1,\ldots,i_k} \;=\; \begin{vmatrix} \frac{\partial f_1}{\partial x_{i_1}} & \frac{\partial f_1}{\partial x_{i_2}} & \cdots & \frac{\partial f_1}{\partial x_{i_k}} \\ \frac{\partial f_2}{\partial x_{i_1}} & \frac{\partial f_2}{\partial x_{i_2}} & \cdots & \frac{\partial f_2}{\partial x_{i_k}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_k}{\partial x_{i_1}} & \frac{\partial f_k}{\partial x_{i_2}} & \cdots & \frac{\partial f_k}{\partial x_{i_k}} \end{vmatrix}$$

We define the norm of the k-form as

$$\| \mathrm{d}f_1 \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k \| \;=\; \sqrt{\sum_{1 \le i_1 < \cdots < i_k \le d} \omega^2_{i_1,\ldots,i_k}}\; \mathrm{d}x.$$

4.2.2 Definition and Properties

For a domain $D \subseteq \mathbb{M}$ we define the comparison measure over D as the normalized integral of the value of the k-form,

$$\kappa_D(F) \;=\; \int_{x \in D} \| \mathrm{d}f_1 \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k \| \;/\; \mathrm{vol}(D).$$

We obtain the *global comparison measure* as $\kappa(F) = \kappa_{\mathbb{M}}(F)$ and note that $0 \le \kappa(F) < \infty$. We may also shrink D toward a point $x \in \mathbb{M}$ and obtain the value $\kappa_x(F)$ in the limit This furnishes the *local comparison measure*, which is the function $\kappa : \mathbb{M} \to \mathbb{R}$ defined by $\kappa(x) = \kappa_x(F)$. Note that the global measure is the average local measure:

$$\kappa(F) \;=\; \int_{x \in \mathbb{M}} \kappa(x)\, \mathrm{d}x \;/\; \mathrm{vol}(\mathbb{M}).$$

Using this relationship we can deduce properties of the global from properties of the local measure. The local comparison measure can be evaluated directly at a point as the norm of the k-form

$$\kappa(x) = \sqrt{\sum_{1 \le i_1 < \cdots < i_k \le d} \omega^2_{i_1,\ldots,i_k}}\;.$$

For example, if we have $k = 2$ functions on a 2-manifold embedded in $\mathbb{R}^3$, then $\kappa(x)$ is the length of the cross-product of the two gradients at x.

The comparison measure satisfies a number os useful algebraic properties. However, it does not satisfy the triangle inequality. This can be easily seen from a counter example for the case $k = 2$. Let g be a constant function over $\mathbb{M}^d$. This implies $\kappa(f_1, g)$ and $\kappa(g, f_2)$ are equal to zero whereas $\kappa(f_1, f_2)$ can attain an arbitrarily large positive value.

1. `Symmetry`: $\kappa(\ldots, f_i, \ldots, f_j, \ldots) = \kappa(\ldots, f_j, \ldots, f_i, \ldots)$.
2. `Degeneracy`: $\kappa(F) = 0$ if $\mathrm{d}f_i = \mathrm{d}f_j$ for $1 \leq i \neq j \leq k$.
3. `Scaling`: $\kappa(\alpha f_1 + \beta, f_2, \ldots, f_k) = |\alpha| \cdot \kappa(f_1, f_2, \ldots, f_k)$, with $\alpha, \beta \in \mathbb{R}$.
4. `Sub-additivity`: $\kappa(f_1 + g_1, f_2, \ldots, f_k) \leq \kappa(f_1, f_2, \ldots, f_k) + \kappa(g_1, f_2, \ldots, f_k)$.
5. `Sub-multiplicativity`: $\frac{\kappa(f_1, \ldots, f_i, f_{i+1}, \ldots, f_k)}{\mathrm{vol}(\mathbb{M})} \leq \kappa(f_1, \ldots, f_i) \cdot \kappa(f_{i+1}, \ldots, f_k)$.

The degeneracy, scaling, and symmetry properties follow immediately from the definition. The wedge product is bilinear. Therefore, the k-form $(\mathrm{d}(f_1+g_1) \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k)$ can be written as a sum of $(\mathrm{d}f_1 \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k)$ and $(\mathrm{d}g_1 \wedge \mathrm{d}f_2 \wedge \ldots \wedge \mathrm{d}f_k)$. Writing out the expression for the norm, we can see that the sub-additivity property holds for $\kappa(x)$ at every point on the d-manifold and hence for $\kappa(F)$.

The proof of the sub-multiplicativity property requires more work. We look at the special case $i = 1$ in detail. We show that $\kappa_x(f_1, \ldots, f_k) \leq \kappa_x(f_1) \cdot \kappa_x(f_2, \ldots, f_k)$ for every point $x \in \mathbb{M}$. To this end, we develop each determinant $\omega_{i_1,\ldots,i_k}$ along the first row. Writing r for the sequence of k indices from $1, 2, \ldots, d$, we get

$$\omega_r \;=\; \sum_s (-1)^{\pi_r(s)+1} \omega_s \cdot \omega_{\hat{s}},$$

where s ranges over all indices in r (or subsequences of r of length 1), $\hat{s}$ is the complementary subsequence of $k-1$ indices from r, $1 \leq \pi_r(s) \leq |r|$ is the position of

s in r, ω_s denotes elements in the first row and $\omega_{\hat{s}}$ denotes the corresponding minors. Squaring the determinant and summing over all choices of k indices from $1, 2, \ldots, d$ we get

$$\begin{aligned}\sum_r \omega_r^2 &= \sum_r \left(\sum_s (-1)^{\pi_r(s)+1} \omega_s \cdot \omega_{\hat{s}} \right)^2 \\ &= \sum_r \sum_s \omega_s^2 \cdot \omega_{\hat{s}}^2 + \sum_r \sum_{s \neq t} (-1)^{\pi_r(s)+\pi_r(t)} \omega_s \cdot \omega_{\hat{s}} \cdot \omega_t \cdot \omega_{\hat{t}}\end{aligned}$$

Let A denote the second sum. We rewrite this sum, A, over the cross terms by pairing ω_s with $\omega_{\hat{t}}$ instead of $\omega_{\hat{s}}$. Note that the sequence $\hat{t}$ contains s. Given two disjoint sequences, α and β, we combine them by sorting the indices obtained from the union and denote this new sequence by $\alpha;\beta$. Let m be a subsequence of $k-2$ indices from r. We express $\hat{t}$ as the union of indices in s and m to get

$$A \;=\; \sum_r \sum_{\substack{s \neq t \\ s;t;m=r}} (-1)^{\pi_r(s)+\pi_r(t)} \omega_s \cdot \omega_{s;m} \cdot \omega_t \cdot \omega_{t;m}$$

Let a be a sequence of $k-2$ indices from $1, \ldots, d$ and b be a sequence of unit length that is not contained in a. In order to get an upper bound on the above sum, consider the following sum of squares:

$$\begin{aligned}B &= \sum_a \left(\sum_b (-1)^{\pi_{b;a}(b)} \omega_b \cdot \omega_{b;a} \right)^2 \\ &= \sum_a \sum_b \omega_b^2 \cdot \omega_{b;a}^2 + \sum_a \sum_{b \neq c} (-1)^{\pi_{b;a}(b)+\pi_{c;a}(c)} \omega_b \cdot \omega_{b;a} \cdot \omega_c \cdot \omega_{c;a}\end{aligned}$$

by merely expanding the squares. In the next step we extract all terms containing ω_u^2, $1 \leq u \leq d$ and therefore switch the order of the summation in the first sum. We switch the order of summation in the second sum as well. $b;c;a$ forms a sequence

of length k. Denoting this sequence as r, we reorganize the terms as a sum over all sequences r. m, s and t replace a, b and c respectively.

$$\begin{aligned} B &= \sum_{u,a} \omega_u^2 \cdot \omega_{u;a}^2 + \sum_r \sum_{\substack{s \neq t \\ s;t;m=r}} (-1)^{\pi_r(s)+\pi_r(t)-1} \omega_s \cdot \omega_{s;m} \cdot \omega_t \cdot \omega_{t;m} \\ &= \sum_{u,a} \omega_u^2 \cdot \omega_{u;a}^2 \; - \; A \end{aligned}$$

Note that either $\pi_{b;a}(b) = \pi_r(s) - 1$ or $\pi_{c;a}(p) = \pi_r(t) - 1$ depending on whether the index in the sequence b is greater or smaller than the one in c. Since B is non-negative, we have

$$A \;\leq\; \sum_{u,a} \omega_u^2 \cdot \omega_{u;a}^2$$

Therefore, we get an upper bound on $\sum_r \omega_r^2$, namely

$$\begin{aligned} \sum_r \omega_r^2 &\leq \sum_r \sum_s \omega_s^2 \cdot \omega_{\bar{s}}^2 + \sum_{u,a} \omega_u^2 \cdot \omega_{u;a}^2 \\ &= \left(\sum_u \omega_u^2 \right) \left(\sum_v \omega_v^2 \right) \end{aligned}$$

letting v range over all subsequences of $k-1$ indices from $1, 2, \ldots, d$. This is because a term $\omega_u^2 \cdot \omega_v^2$ appears either as $\omega_s^2 \cdot \omega_{\bar{s}}^2$ if u and v do not have any common indices or as $\omega_u^2 \cdot \omega_{u;a}^2$ otherwise. Because this inequality holds for all points $x \in \mathbb{M}$, we have

$$\kappa(f_1, \ldots, f_k) \cdot \mathrm{vol}(\mathbb{M}) \;\leq\; \kappa(f_1) \cdot \mathrm{vol}(\mathbb{M}) \cdot \kappa(f_2, \ldots, f_k) \cdot \mathrm{vol}(\mathbb{M}),$$

which implies the sub-multiplicativity of κ for the case $i = 1$.

4.2.3 PL Algorithm

In practice, functions are measured at discrete points in the manifold and linearly interpolated within simplices in a triangulation of the manifold. In such a setting,

$\kappa(F)$ is computed in a loop over the d-simplices in the triangulation. Since all functions are linear over a d-simplex, their differentials are constant within the d-simplex. The norm of the k-form is evaluated at a point within the d-simplex directly from the formula and weighted by the volume of the d-simplex. We divide this weighted sum by the volume of the triangulation to get $\kappa(F)$.

4.3 Jacobi Set Interpretation

In this section, we give an alternate interpretation for our comparison measure in terms of critical points of the functions. We have a result only for the case when $k = d = 2$. We begin with an introduction to Jacobi sets, which play an important role in this alternate interpretation.

4.3.1 Jacobi Sets

The *Jacobi set* of two Morse functions defined over a common 2-manifold is the set of critical points of the restrictions of one function to the level sets of the other function. For a generic pair of Morse functions, the Jacobi set is a smoothly embedded 1-manifold. Equivalently, the Jacobi set is the set of points where the gradients of the functions are parallel and is hence symmetric with respect to the two functions. The Jacobi set of two Morse functions, f and g, is denoted by $\mathbb{J} = \mathbb{J}(f, g) = \mathbb{J}(g, f)$. We think of piecewise linear functions as the limit of a series of smooth functions and use this intuition to transport the definition of Jacobi sets from the smooth to the piecewise linear setting.

Now, let f and g be two piecewise linear functions defined over a triangulation of a 2-manifold. The Jacobi set of the two functions is a one-dimensional subcomplex of the triangulation that can be thought of as the limit of Jacobi sets for the corresponding pairs of smooth functions. The algorithm identifies edges lying on the

Jacobi set and returns the union of these edges. Each edge ab is individually classified to be in the Jacobi set or not by determining if it is critical with respect to the function $h_\lambda = f + \lambda g$, where the value of the parameter λ is given by the condition $h_\lambda(a) = h_\lambda(b)$. Criticality testing of an edge is done similar to that of a vertex namely, based on the structure of the lower link of the edge. In a 2-manifold, the link of an edge consists of two vertices. The function IsJacobi does the criticality testing.

`integer` IsJacobi (`Edge` ab)
$\lambda = \frac{f(b)-f(a)}{g(a)-g(b)}$;
Let x, y be vertices in $\mathrm{Lk}\, ab$;
`if` $(h_\lambda(x) < h_\lambda(a)$ and $h_\lambda(y) < h_\lambda(a))$ `then return` Maximum `endif`
`if` $(h_\lambda(x) > h_\lambda(a)$ and $h_\lambda(y) > h_\lambda(a))$ `then return` Minimum `endif`
`return` Regular.

Jacobi sets were introduced by Edelsbrunner and Harer [27] and we refer to their paper for detailed proofs of the assertions and the algorithm.

4.3.2 Alternative Formulations

Consider two Morse functions $f, g : \mathbb{M}^2 \to \mathbb{R}$. Assuming that $\mathbb{M}^2$ is embedded[1] in $\mathbb{R}^3$, we first rewrite $\| \mathrm{d}f \wedge \mathrm{d}g \|$ as the length of the cross product between the gradients of f and g. $\kappa(F)$ is equal to

$$\kappa(f,g) = \frac{\int_{x\in\mathbb{M}^2} \| \nabla f(x) \times \nabla g(x) \| \, \mathrm{d}x}{\int_{x\in\mathbb{M}^2} \mathrm{d}x}.$$

We now rewrite $\kappa(f, g)$ as an integral over the Jacobi set, which can also be expressed as the set of critical points of f restricted to level sets of g. Let v be a point on the

[1] All results in the paper hold without this assumption as well because the tangent plane at each point in $\mathbb{M}^2$ can be embedded in $\mathbb{R}^3$. However, notation required to represent the gradients and their different embeddings becomes too cumbersome.

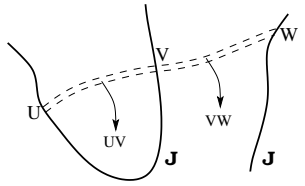

Figure 4.1: UV and VW are two strips on $\mathbb{M}^2$ connecting subsets of the Jacobi set.

Jacobi set and let $u, w \in \mathbb{J}$ be its neighbors along the isocontour $g^{-1}(g(v))$. By definition, v is either a maximum or a minimum of the restriction of f. If v is a local maximum of f restricted to $g^{-1}(g(v))$, then u and w are local minima. Let V be a connected subset of $\mathbb{J}$ containing v such that the neighbors of points in V along isocontours of g form two connected subsets U and W, of $\mathbb{J}$ (see Figure 4.1). $\mathbb{M}^2$ can be partitioned into strips like UV and VW that are union of isocontour segments connecting points in V with those of U and W respectively. $\| \nabla f \times \nabla g \|$ is equal to the product of $\| \nabla_{T(g)} f \|$ and $\| \nabla g \|$, where $\nabla_{T(g)} f(p)$ is the directional derivative of f at the point p along the tangent to the isocontour $g^{-1}(g(p))$. This is because $\nabla_{T(g)} f$ is exactly the projection of ∇f in the direction normal to ∇g. The integral of $\| \nabla f \times \nabla g \|$ over the strip UV can be expressed as double integral to give

$$\begin{aligned}\int_{x\in UV} \| \nabla f(x) \times \nabla g(x) \| &= \int_{x\in UV} \| \nabla_{T(g)} f(x) \| \, \| \nabla g(x) \| \, \mathrm{d}x \\ &= \int_{v\in V} \left(\int_{x\in uw} \| \nabla_{T(g)} f(x) \| \right) \| \nabla g(v) \| \, \mathrm{d}v.\end{aligned}$$

Here, uw is the section of the isocontour $g^{-1}(g(v))$ between u and w and passing through v. f is monotonic along the isocontour and so the inner integral is computed as the difference between the values that f evaluates to at u and v, the boundary points.

$$\int_{x\in UV} \| \nabla f(x) \times \nabla g(x) \| = \int_{v\in V} |f(v) - f(u)| \, \| \nabla g(v) \| \, \mathrm{d}v.$$

This equality holds for the strip VW also. Adding the integrals over UV and VW, we get

$$\begin{aligned}
&\int_{x\in UV} \parallel \nabla f(x) \times \nabla g(x) \parallel + \int_{x\in VW} \parallel \nabla f(x) \times \nabla g(x) \parallel \mathrm{d}x \\
= &\int_{v\in V} |f(v) - f(u)| \parallel \nabla g(v) \parallel \mathrm{d}v + \int_{v\in V} |f(v) - f(w)| \parallel \nabla g(v) \parallel \mathrm{d}v \\
= &\int_{v\in V} |2f(v) - f(u) - f(w)| \parallel \nabla g(v) \parallel \mathrm{d}v.
\end{aligned}$$

The sum of integrals over all such subsets V of $\mathbb{J}$ is equal to the sum of integrals of $\parallel \nabla f \times \nabla g \parallel$ over the corresponding strips UV and VW. The latter sum equals twice the integral of $\parallel \nabla f \times \nabla g \parallel$ over $\mathbb{M}^2$ because each strip is counted twice in the above sum. We now have our second formulation for $\kappa(f, g)$ namely

$$\kappa(f, g) = \frac{\int_{v\in\mathbb{J}} |2f(v) - f(u) - f(w)| \parallel \nabla g(v) \parallel \mathrm{d}v}{2\int_{x\in\mathbb{M}^2} \mathrm{d}x}. \tag{4.1}$$

where $u, w \in \mathbb{J}$ are neighbors of v along $g^{-1}(g(v))$. This formulation shows that $\kappa(f, g)$ captures the deviations of one function over isocontours of the other.

We now derive a third formulation again as an integral over the Jacobi set. We split the integral over $\mathbb{J}$ in Equation (4.1) into two integrals:

$$\begin{aligned}
&\int_{v\in\mathbb{J}} |2f(v) - f(u) - f(w)| \parallel \nabla g(v) \parallel \mathrm{d}v \\
= &\int_{v\in\mathbb{J}} |f(v) - f(u)| \parallel \nabla g(v) \parallel \mathrm{d}v + \int_{v\in\mathbb{J}} |f(v) - f(w)| \parallel \nabla g(v) \parallel \mathrm{d}v
\end{aligned}$$

Each point in $\mathbb{J}$ is counted four times, twice in each integral. The sum of these two integrals can written as one integral by introducing a sign function to give a formulation of $\kappa(f, g)$ that does not use any explicit pairing along the isocontours:

$$\kappa(f, g) = \frac{2\int_{v\in\mathbb{J}} sgn(v)\, f(v) \parallel \nabla g(v) \parallel \mathrm{d}v}{\int_{x\in\mathbb{M}^2} \mathrm{d}x} \tag{4.2}$$

$$\text{where } sgn(v) = \begin{cases} 1 & \text{if } v \text{ is a maximum of } f \text{ in } g^{-1}(g(v)) \\ -1 & \text{if } v \text{ is a minimum of } f \text{ in } g^{-1}(g(v)) \end{cases}$$

Each generic level set $g^{-1}(t)$ is a collection of topological circles and f restricted to this level set has equally many minima and maxima. Letting k be this common number, we form a pairing $\{(u_i, v_i) \mid 1 \le i \le k\}$ between the minima u_i and the maxima v_i such that $f(v_i) - f(u_i) > 0$ for all i. We further develop the integral in Equation (4.1) to find a formulation in which the pairing directly relates to a ranking of these critical points. Writing $\mathrm{pers}(u_i) = \mathrm{pers}(v_i) = f(v_i) - f(u_i)$, we get

$$\kappa(f, g) = \frac{\int_{v\in\mathbb{J}} \mathrm{pers}(v)\, \mathrm{d}g}{\int_{x\in\mathbb{M}^2} \mathrm{d}x}. \tag{4.3}$$

Indeed, Equation (4.1) is a special case in which the integration is done over two pairings of the points in the Jacobi set. A more meaningful (single) pairing is obtained using the concept of persistent homology [31]. It is easy to explain for a Morse function $f_t : \mathbb{S}^1 \to \mathbb{R}$, which has equally many minima and maxima. Sweeping the circle in the direction of increasing function value, we get a new component whenever we pass a minimum and we merge two components whenever we pass a maximum, except that we complete the circle when we pass the last maximum. Each component is represented by its oldest minimum (the one with smallest function value).

`Rule 1.` If a maximum merges two components we pair it with the younger of the two minima representing the two components. The older minimum stays on to represent the merged component.

`Rule 2.` The last maximum is paired with the first minimum.

The persistence is a notion of importance of a critical point that has found use in a number of applications involving smooth functions, see for example [2, 13, 30].

4.3.3 Alternative Algorithms

The alternative formulae for κ also lead to algorithms that compute $\kappa(f, g)$ for piecewise linear functions f and g on a triangulation K of $\mathbb{M}$. We first describe an algorithm that follows directly from Equations (4.2).

Integral over Jacobi set. In the piecewise linear setting, the integral over $\mathbb{J}$ in Equation (4.2) becomes a sum over all edges in $\mathbb{J}$. The contribution of an edge to $\kappa(f, g)$ can now be expressed in a closed form because f varies linearly over the edge. Let a, b be the end points of an edge in $\mathbb{J}$. IsJacobi (ab) classifies an edge $ab \in \mathbb{J}$ as a maximum or a minimum. The contribution of the edge is equal to $\frac{1}{2} \cdot (f(a) + f(b)) \cdot |g(b) - g(a)|$ if the edge is a maximum and is equal to $-\frac{1}{2} \cdot (f(a) + f(b)) \cdot |g(b) - g(a)|$ if the edge is a minimum.

Isocontour sweep algorithm. We need to sweep K in the direction of increasing value of g, maintaining the level set, $g^{-1}(t)$ in order to implement both Equations (4.1) and (4.3). An edge in the Jacobi set is identified using the IsJacobi subroutine. We can then use the persistence algorithm to compute pers(v) for all critical points v of $g^{-1}(t)$, which are the intersection points between the level set and the Jacobi set. However, in order to implement (4.3), we need to identify the level sets of g where the persistence pairing changes. This is unclear and we mention this problem as future work. The characterization of when the pairing changes is easier in the case of Equation (4.1). Before describing the implementation of Equation (4.1), we derive its analogous expression in the piecewise linear setting.

A connected subset of an edge in the input triangulation is called a *segment*. The end points of a segment on a given edge can be specified by the value of the function g because it is linearly interpolated within the edge. The Jacobi set of two functions defined over a triangulation of a 2-manifold consists of critical edges

of the triangulation. Each point $v \in \mathbb{J}$ is paired with two other points $u, w \in \mathbb{J}$ namely the critical points of f restricted to $g^{-1}(g(v))$ that are adjacent to v in the isocontour. We call (u, v) and (v, w) as $\mathbb{J}$-*pairs*. Extending this concept to segments, if s and t are segments on $\mathbb{J}$ whose points form $\mathbb{J}$-pairs then we call $[s, t]$ a *segment-pair*. Figure 4.2 shows two components of $\mathbb{J}$ and a few segment-pairs. Note that two segment-pairs have at most one common $\mathbb{J}$-pair. $\kappa(f, g)$ is expressed as an integral

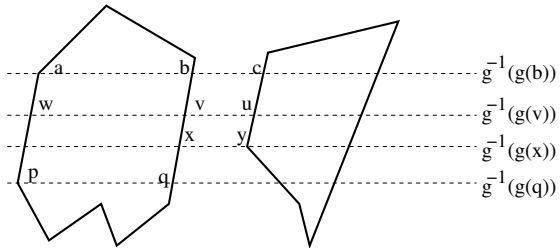

Figure 4.2: $[g(a), g(p)]$ and $[g(b), g(q)]$ are a segment-pair within the same component of $\mathbb{J}$ and $[g(b), g(x)]$ and $[g(c), g(y)]$ are a segment-pair between different components of $\mathbb{J}$. The solid lines are components of $\mathbb{J}$ and the dashed lines are the isocontours.

over $\mathbb{J}$ in Equation (4.1). We split this integral into two as before and rewrite it as a sum over all segment-pairs:

$$\begin{aligned} \kappa(f, g) &= 2 \int_{(u,v) \in P} |f(v) - f(u)| \parallel \nabla g(v) \parallel \mathrm{d}v \\ &= 2 \sum_{[s,t] \in P_s} \int_{u \in s, v \in t} |f(v) - f(u)| \parallel \nabla g(v) \parallel \mathrm{d}v \end{aligned}$$

where P is the set of all $\mathbb{J}$-pairs and P_s is the set of all segment-pairs. Each $\mathbb{J}$-pair is counted twice in the integral over $\mathbb{J}$ and hence the factor 2 in the integral over P. The integral over a segment-pair can be rewritten using the mean value theorem because f and g vary linearly within a segment.

$$\int_{u \in s, v \in t, [s,t] \in P_s} |f(v) - f(u)| \parallel \nabla g(v) \parallel \mathrm{d}v = |f_{mid}(s) - f_{mid}(t)| \cdot \Delta g(t).$$

Here, $\Delta g(t)$ is the difference between the values of g at the end points of segment t and $f_{mid}(s)$ denotes the value of f at the midpoint of segment s.

Determining the segment-pairs is an important step in evaluating $\kappa(f, g)$. The topology of the isocontour changes as we sweep the manifold using isocontours of g with increasing isovalues. At any stage of the sweep, a segment of $\mathbb{J}$ could start, stop, or continue to intersect the current isocontour. The first two events happen only when the isocontour passes through a vertex. We are now ready to describe the isocontour sweep algorithm.

We compute $\kappa(f, g)$ by maintaining a list of segment-pairs that intersect the current isocontour during the sweep. The sweep is done by traversing the vertex list sorted on the value of the function g. The contribution of a segment-pair is added when it stops intersecting the current isocontour. The processing done at each step of the sweep depends on the configuration of the upper and lower stars of the current vertex v with respect to the function g.

***Regular*:** The new edges that the isocontour crosses when the sweep passes through a regular vertex are exactly the ones in the upper star. If $\mathbb{J}$ does not pass through the vertex then there is nothing to do. If it does, then at least one segment-pair stops intersecting the isocontour. For each one of the segment-pairs that is destroyed, we add its contribution to $\kappa(f, g)$. Figure 4.3 shows a scenario where the segment-pair $[a, b]$ is deleted and two new segment-pairs $[a', x]$ and $[y, b']$ are inserted into the list of segment-pairs.

***Minimum*:** $\mathbb{J}$ passes through the minimum by definition. No segment pairs are destroyed because the isocontour does not intersect any edges in $\mathrm{St}\, v$ before the sweep past the minimum. At least two segment-pairs are born and we insert them into the list. Figure 4.4 shows the neighborhood of a minimum after the

sweep crosses it.

***Maximum*:** $\mathbb{J}$ passes through a maximum by definition. We compute and add contributions of the segment-pairs that are destroyed and delete them from the list. Note that no segment-pair is born because the isocontour component is destroyed as we sweep past the maximum (see Figure 4.4).

***Saddle*:** Figure 4.5 shows the neighborhood of a saddle before and after the isocontour sweeps past the vertex and how the connectivity of the isocontour changes. The change in connectivity destroys some segment-pairs and introduces new ones. In addition, other segment-pairs could be created or destroyed by segments of $\mathbb{J}$ passing through the saddle. The new segment-pairs are inserted into the list and contributions of the segment-pairs that get destroyed are added to $\kappa(f, g)$ before they are deleted from the list.

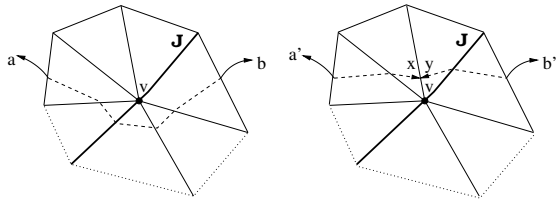

Figure 4.3: The segment-pairs before and after the isocontour passes through a regular vertex. The dark line is a portion of $\mathbb{J}$ passing through v, the dashed line is the isocontour and the dotted lines are edges of the lower link.

$\kappa(f, g)$ is computed as the sum of contributions from each segment-pair at the end of the sweep. There are a couple of issues that need to be handled in practice which, if neglected, could lead to erroneous results. The first one deals with consistent handling of boundary edges and the second deals with degeneracies in the data. The former can be easily handled while the latter requires a non-trivial solution. We discuss the former issue and describe how to handle degeneracies in Section 4.3.5.

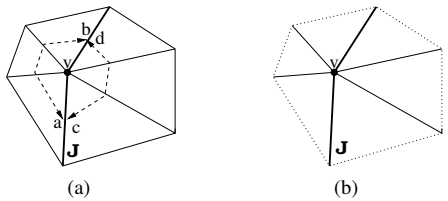

Figure 4.4: The segment-pairs after the isocontour passes through a local minimum (a) and local maximum (b). The dark line is a portion of $\mathbb{J}$ passing through v, the dashed line is the isocontour and the dotted lines are edges of the lower link.

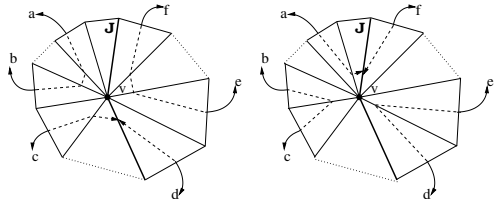

Figure 4.5: The segment-pairs before and after the isocontour passes through a 3-way saddle. The dark line is a portion of $\mathbb{J}$ passing through v, the dashed line is the isocontour and the dotted lines are edges of the lower link.

A basic assumption we make is that each point in $\mathbb{J}$ has a pair. This is true if our domain is a 2-manifold. But, often the input functions are defined over a surface mesh that has a boundary. The isocontours in such a case may no longer be closed 1-manifolds. One way to resolve this issue is to introduce vertices at infinity corresponding to each boundary component and add simplices containing boundary simplices and the new vertices as faces. This modified the domain to become a 2-manifold. A drawback of this approach is the increase in number of simplices to be processed. Instead, we simulate this construction by including the edges lying on the boundary into $\mathbb{J}$. These edges are paired only in one direction *i.e* towards the interior.

4.3.4 Local Contributions

$\kappa(f, g)$ can be computed using the definition or by implementing one of the alternative formulae. We define functions express the contributions to $\kappa(f, g)$ from triangles in the mesh or edges in $\mathbb{J}$ as the case maybe. The contribution from triangles in the mesh to $\kappa(f, g)$ equals the area of the triangle times $\kappa_x(F)$ for a point x lying inside the triangle. κ_t assigns this value to each triangle t. Each edge in $\mathbb{J}$ consists of multiple segments each of which contribute to $\kappa(f, g)$ when it is computed using Equation (4.1). $\tilde{\kappa}_e$ assigns the sum of contributions of these segments to the edge in $\mathbb{J}$. These local contributions help rank the Jacobi set edges and triangles in the manifold based on their importance. Visualizing these functions helps identify interesting regions of the manifold. The contributions from the Jacobi set edges to $\kappa(f, g)$ when computed using Equation (4.2) are not useful in this respect because they could be negative as well.

4.3.5 Handling Degeneracies

Degeneracies should be handled consistently for a correct implementation of the algorithms described in this chapter. In the generic case,

1. No two vertices have the same function value and
2. No vertex in the link of an edge ab has the same value of $h_\lambda\,(= f + \lambda g)$ as the edge. Here, the value of the parameter is computed by determining when ab becomes horizontal *i.e.* $h_\lambda(a) = h_\lambda(b)$.

The degenerate cases have to be handled while computing the lower link in IsJacobi. We resolve the degenerate cases by simulating a perturbation scheme similar to SoS [32] but applied to two functions f and g. Let f^i and $g^i, i = 1, \ldots, n$ denote

the value of the functions f and g at vertex i of the triangulation. We perturb both functions as follows:

$$\tilde{f^i} = f^i + \epsilon_i \,,\, \tilde{g^i} = g^i + \delta_i$$
$$\text{with } \epsilon_i = \epsilon^{2^i} \,,\, \delta_i = \epsilon^{2^{n+i}}, i = 1, \ldots, n$$

for a suitably small but positive value of ϵ. Clearly

$$\epsilon_1 >> \epsilon_2 >> \ldots >> \epsilon_n >> \delta_1 >> \delta_2 >> \ldots >> \delta_n.$$

Let ab be an edge and v a vertex in its link with $1 \leq a < b < v \leq n$. The value of λ for which ab becomes horizontal is given by

$$\lambda = \frac{f^b - f^a}{g^a - g^b}.$$

The vertex v lies in the lower link of ab if $h_\lambda(v) \leq h_\lambda(a)$ for the above value of λ[2]. We have

$$f^v + \frac{f^b - f^a}{g^a - g^b} g^v \leq f^a + \frac{f^b - f^a}{g^a - g^b} g^a.$$

Multiplying both sides with $(g^a - g^b)$ and rearranging the terms, we get

$$(f^a - f^v)(g^a - g^b) + (f^b - f^a)(g^a - g^v) \geq 0 \text{ and } (g^a - g^b) > 0 \text{ or}$$
$$(f^a - f^v)(g^a - g^b) + (f^b - f^a)(g^a - g^v) \leq 0 \text{ and } (g^a - g^b) < 0.$$

Let

$$X = (f^a - f^v)(g^a - g^b) + (f^b - f^a)(g^a - g^v).$$

We can rewrite it as

$$X = f^a(g^v - g^b) + f^b(g^a - g^v) + f^v(g^b - g^a).$$

[2]Note that we can use $h_\lambda(b)$ instead of $h_\lambda(a)$ in this inequality

Replacing f^i, g^i with their perturbed versions in the expression for X, we get

$$\begin{aligned}\tilde{X} = X \quad &+ \quad \epsilon_a(g^v - g^b) + \epsilon_b(g^a - g^v) + \epsilon_v(g^b - g^a) \\ &+ \quad \delta_a(f^b - f^v) + \delta_b(f^v - f^a) + \delta_v(f^a - f^b) \\ &+ \quad \delta_a(\epsilon_b - \epsilon_v) + \delta_b(\epsilon_v - \epsilon_a) + \delta_v(\epsilon_a - \epsilon_b)\end{aligned}$$

We can determine if v is in the lower link of ab from the sign of X and $(g^a - g^b)$. However, one or both X and $(g^a - g^b)$ could be zero, making it difficult to classify v. This problem does not arise if we use the perturbed versions instead *i.e.* $\tilde{X}$ and $(\tilde{g}_a - \tilde{g}_b)$. The values of X and $\tilde{X}$ are the same no matter which edge-vertex pair is chosen from the triangle abv. We use this fact to unify the lower link test for all inputs from the triangle abv. Table 4.1 lists the test for each edge-vertex pair from the triangle abv $(a < b < v)$. Computing the perturbed functions is not feasible

Input		Test
Edge	Vertex	
(a, b)	v	$\tilde{X} \cdot (\tilde{g}_a - \tilde{g}_b) > 0$
(v, a)	b	$\tilde{X} \cdot (\tilde{g}_v - \tilde{g}_a) > 0$
(b, v)	a	$\tilde{X} \cdot (\tilde{g}_b - \tilde{g}_v) > 0$

Table 4.1: Lower link tests for edge-vertex pairs from a triangle abv.

because we need very high precision. So, we simulate the perturbation scheme by a series of comparisons. We now give the algorithm that classifies a vertex present in the link of an edge.

`boolean` LowerLink (`Edge` ij, `Vertex` k)

Sort i, j, k and store in a, b, v $(a < b < v)$

Let $index_i \leftarrow$ Index of i in sorted list

Let $index_j \leftarrow$ Index of j in sorted list

`if` $index_j = (index_i + 1) \bmod 3$
  `then` return $((\text{Pos1}(\tilde{X})$ and $\text{Pos2}(\tilde{g}_i - \tilde{g}_j))$ or $(\text{Neg1}(\tilde{X})$ and $\text{Neg2}(\tilde{g}_i - \tilde{g}_j)))$
  `else` return $((\text{Pos1}(\tilde{X})$ and $\text{Pos2}(\tilde{g}_j - \tilde{g}_i))$ or $(\text{Neg1}(\tilde{X})$ and $\text{Neg2}(\tilde{g}_j - \tilde{g}_i)))$

`endif`

Pos1 and Pos2 return True if and only if their inputs are positive and Neg1 and Neg2 return True if and only if their inputs are negative (*i.e.* Pos1 and Pos2, respectively, are False). Pos2 uses the indices i and j to compare the perturbed functions if g^i is equal to g^j. More comparisons are required to simulate perturbation of X in Pos1.

`boolean` Pos1 (`Expression` $\tilde{X}$)

`if` $X \neq 0$ `then return` $(X > 0)$ `endif`

`if` $g^v \neq g^b$ `then return` $(g^v > g^b)$ `endif`

`if` $g^a \neq g^v$ `then return` $(g^a > g^v)$ `endif`

`if` $f^b \neq f^v$ `then return` $(f^b > f^v)$ `endif`

`return` True

The sequence of comparisons is determined by the fact that $\epsilon_a >> \epsilon_b >> \epsilon_v >> \delta_a >> \delta_b >> \delta_v$ and the respective terms will dominate the expression for $\tilde{X}$ if they are not multiplied by zero.

4.4 Experiments

We perform multiple computational experiments to illustrate properties of our comparison measure. The visualizations show how the local measure helps study the

relationships between functions.

4.4.1 Synthetic Functions

We first use a set of five analytic functions to get a feel for our global measure. All functions are sampled on a regular grid over the region $[-2\pi, 2\pi] \times [-2\pi, 2\pi]$. This point set is triangulated and a piecewise linear approximation of the function is available as our data. Table 4.2 lists the various functions that we use along with the values of κ for each function pair. We have $\kappa(\texttt{cup}) = \kappa(\texttt{sad}) = 9.61$, $\kappa(\texttt{sin}) = \kappa(\texttt{cos}) = 0.96$, and $\kappa(\texttt{abs}) = 1.0$. Note that although `sin` can be obtained from `cos` by shifting the axes, the two functions differ according to our measure (*i.e.* κ does not go to zero) because we require local similarity as well. κ is not scale invariant and therefore it cannot be used to decide whether a given pair of functions is more similar than another pair.

		cup	sad	sin	cos	abs
$x^2 + y^2$	cup	0.00	78.96	5.76	5.24	6.28
$x^2 - y^2$	sad	78.96	0.00	5.76	6.28	6.28
$\sin x + \sin y$	sin	5.76	5.76	0.00	0.63	0.64
$\cos x + \cos y$	cos	5.24	6.28	0.63	0.000	0.63
$\lvert x \rvert$	abs	6.28	6.28	0.64	0.63	0.00

Table 4.2: Table of k-values for pairs of analytic functions. The matrix is symmetric and diagonal entries are equal to zero illustrating the symmetry and degeneracy properties of our global measure.

4.4.2 Testing Algebraic Properties

We illustrate the algebraic properties satisfied by our measure using the five analytic functions mentioned above and their variants. The entries in the diagonal of the matrix in Table 4.2 gives the value of $\kappa(f, f)$ for different functions f and hence they are all equal to 1. Also, note that the $(i, j)^{th}$ entry in the matrix is equal to the $(j, i)^{th}$ entry because our measure is symmetric. Three examples illustrate the

scaling property in Table 4.3. Evidence for the inequality under function addition and sub-multiplicativity are also shown using three examples each in Tables 4.4 and 4.5.

$f(x,y)$	$g(x,y)$	$h(x,y)$	$\kappa(f,g)$	$\kappa(h,g)$
$\cos x + \cos y$	$\sin x + \sin y$	$3(\cos x + \cos y) + 5$	0.632	1.897
$x^2 + y^2$	$x^2 + y^2$	$-x^2 - y^2$	0.000	0.000
$\cos x + \cos y$	$2(\sin x + \sin y) + 3$	$3(\cos x + \cos y) + 5$	1.265	3.795

Table 4.3: Testing the scaling property: The value of $\kappa(F)$ scales with the functions and does not change when one of the functions is modified by adding a scalar. This is shown this using three examples where f_1 has been scaled and shifted to get f_2.

$f_1(x,y)$	$f_2(x,y)$	$g_1(x,y)$	LHS	RHS
$x^2 + y^2$	$x^2 - y^2$	$\sin x + \sin y$	79.20	84.72
$\sin x + \sin y$	$\cos x + \cos y$	$x^2 + y^2$	5.29	5.87
$\sin x + \sin y$	$\cos x + \cos y$	$\lvert x \rvert$	0.81	1.26

Table 4.4: Testing sub-additivity: Three examples give evidence for sub-additivity property. LHS and RHS refer to the left and right side expressions of the inequality in the statement of the property.

$f_1(x,y)$	$f_2(x,y)$	LHS	RHS
$x^2 + y^2$	$x^2 - y^2$	0.50	92.43
$\sin x + \sin y$	$\cos x + \cos y$	0.004	0.91
$\sin x + \sin y$	$\lvert x \rvert$	0.004	0.96

Table 4.5: Testing the sub-multiplicativity property: Three examples give evidence for the inequality under wedge product. LHS and RHS refer to the left and right side expressions of the inequality in the statement of the property.

4.4.3 Comparative Visualization

We develop visualization software for studying scientific data consisting of multiple functions measured over a common 2-manifold. Individual functions can be visualized using a color map, a terrain map, or isocontours. A visual editor allows a particular pair of functions to be chosen from the data. We can compare the two functions using, for example, a terrain map for one and a color map for the other. On the other hand, we can do the comparison by visualizing κ_t and $\tilde{\kappa}_e$, which give the local

contributions from mesh triangles and Jacobi set edges to the comparison measure. A viewing parameter panel lets the user choose one of these options. Figure 4.6 shows a screenshot of the visualization tool. We first compare some of the analytic functions mentioned above.

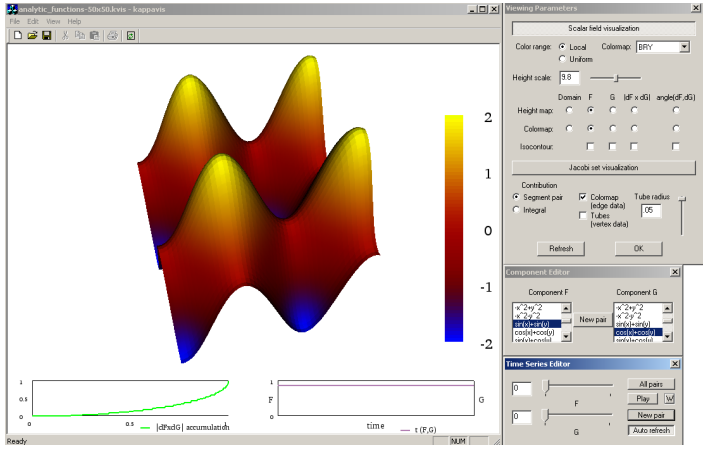

Figure 4.6: Screenshot of the visualization tool used to compare multiple time-varying functions. The panels on the right can be used to select different pairs of functions or time steps and adjust different view parameters.

cup/sad. A comparative visualization reveals that the `cup` and `sad` functions are increasingly dissimilar along diagonals towards the corners of the square domain (see Figure 4.7). The isocontours are orthogonal here and this is reflected in κ_t, which takes on high values in the neighborhood of the diagonal. There exist some regions where the two functions are similar but the dissimilar regions outweigh the effect of the former.

sin/cos. Figure 4.8 shows results from a comparative visualization between `sin` and `cos`. Note again that the isocontours in regions with high values of κ_t are orthogonal and parallel where κ_t takes on low values. The two functions are periodic and this

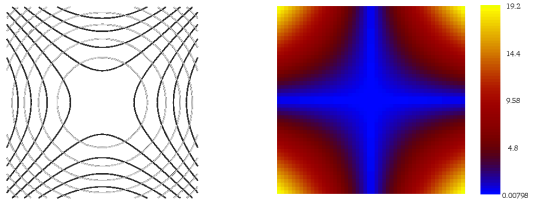

Figure 4.7: (left) Isocontours extracted from the `parabola` (gray) and `sad` (black). (right) A color-mapped visualization of the local contributions from triangles to κ. Note that the contribution is high in regions where the isocontours are orthogonal and low where they are parallel.

can be seen in the visualizations. The edges of the Jacobi set are colored to represent their contribution to κ. The pairing between edges of the Jacobi set can be obtained by tracing isocontours. We can also trace regions where the variation of `sin` over `cos` is large, by following isocontours between pairs of Jacobi set with higher values of $\tilde{\kappa}_e$.

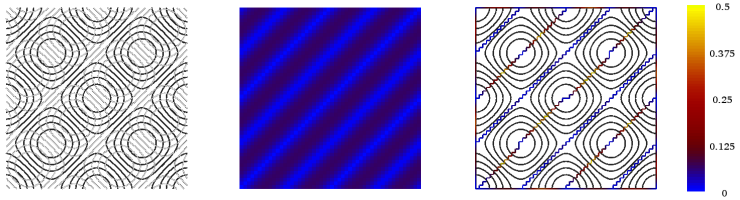

Figure 4.8: (left) Isocontours extracted from the `sin` (gray) and `cos` (black). (middle) Visualization of the local contributions from triangles to κ using a color map. The regions where the isocontours of the two functions are orthogonal contribute more. (right) The Jacobi set with edges colored to represent their contribution to κ.

Time-varying data. The visualization tool can handle time-varying data as well. We study data from the simulation of a combustion process within an engine. Multiple quantities are measured during the numerical simulation of the combustion with the objective of understanding the influence of turbulence on ignition, flame propagation and burnout in compression ignition engine environments [25]. The simulation

is done on a dilute air-fuel mixture that ignites upon compression. Inhomogeneity in the mixture causes ignition to occur at multiple spots. After ignition, the flame propagates from these spots outwards or burns out depending upon the air-fuel ratio. We look at two quantities measured during the simulation, namely `prog` (progress: measure of completion of combustion) and H_2 (hydrogen: the fuel). The quantities are available on a 600×600 grid over 67 time steps. We triangulate the grid and linearly interpolate the quantities within the triangles to obtain time-varying piecewise linear functions.

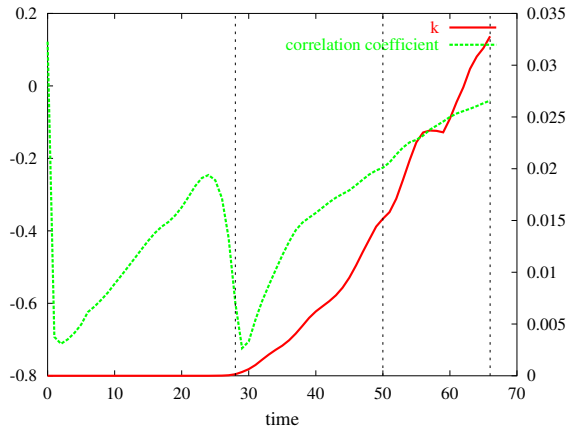

Figure 4.9: The comparison measure κ (scale on right vertical axis) and the correlation coefficient (scale on left vertical axis), both as functions of time as `prog` and H_2 from the combustion dataset change. The vertical markers at time steps 28, 50 and 66 indicate the ones shown in Figure 4.10. The plot of the correlation coefficient shows that `prog` and H_2 are negatively correlated, which is expected because the fuel depletes with progress in combustion.

A time series editor allows the user to choose the time steps for two quantities. Although this level of control is good to have, it may not be useful in practice because of the sheer number of possible pairs of functions. As an aid in choosing time steps

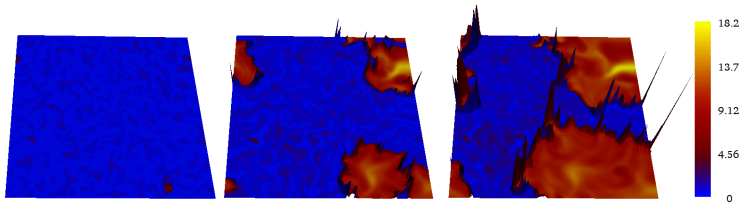

Figure 4.10: Local comparison of the functions `prog` and H_2 from the combustion dataset. The function κ_t, which measures contributions from triangles to κ, is shown using a terrain map and `prog` is mapped to color. From left to right: ignition phase, burning phase, and the end of combustion. The fronts of the flames are tracked by a region that has large contribution to κ. This region is represented by the peaks that enclose the burnt region.

that may be "interesting", we compute κ between the selected quantities at each time step and plot it as a graph. Time steps where κ changes rapidly or attains a global minimum or global maximum are possibly when an interesting event happens.

For the H_2-`prog` pair, the time steps where κ begins to rises from near zero value corresponds to the ignition phase. `prog` and H_2 are turbulent initially. However, both the hydrogen concentration as well as the progress of combustion is low throughout the domain. Therefore, κ has a very low value. After ignition, there is a growing region, namely the front of the flame, over which H_2 concentration decreases sharply as compared to the slower increase in `prog`. This causes an increase in κ till the end of the simulation. Figure 4.9 shows the plot of κ between `prog` and H_2. The different phases of combustion cannot be immediately detected from the plot of κ. We just get an indication that something interesting happens when κ begins to take larger values after time steps 28. The phases become apparent from the visualization of κ_t. Figure 4.10 shows snapshots of the simulation with κ_t mapped to a terrain. Three time steps are shown: the ignition, burning, and the final phase. Note that the flame front is tracked by regions with large contributions. The functions are dissimilar at the front because there is a sharp change for both functions while crossing it, albeit at vastly different rates. The higher peaks in the terrain correspond to sections of

the front that are progressing faster.

Electrostatic potentials. In order to study our local and global comparison measures for functions defined over a three-dimensional domain, we look at an application in biology. A protein-protein complex consists of two or more proteins docked in a stable conformation. For example, the barnase-barstar complex (1BRS) consists of two proteins. The electrostatic potential defined by barnase (N) and barstar (S) individually in their docked conformation and the potential defined by the complex are available to us as functions sampled over the space. We triangulate the space and linearly interpolate to obtain three piecewise-linear functions f_N, f_S, and f_{1BRS}.

	N	S	1BRS	N,S	N,1BRS	S,1BRS	N,S,1BRS
κ	4.01	3.22	7.22	2.30	6.83	5.17	18.66

Table 4.6: The global measure κ computed for all combinations of the three electrostatic datasets.

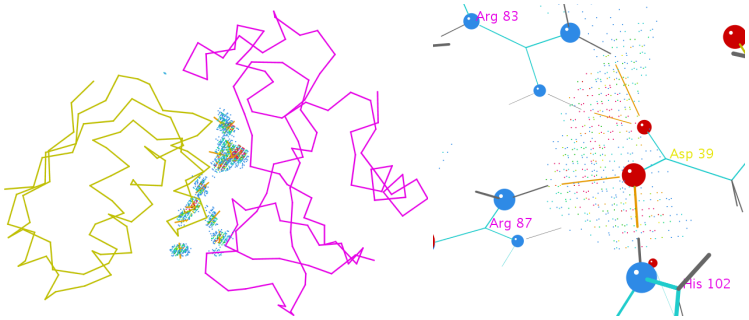

Figure 4.11: Visualization of local comparison measure between electrostatic potentials defined by barnase and barstar in the complex 1BRS. Top: an overview of the regions with high values of κ in the complex. The proteins are shown as alpha-carbon traces, with barnase in magenta and barstar in yellow. Bottom: a closeup of a hydrogen bond cluster. `Asp 39` of barstar hydrogen bonds with `Arg 87`, `Arg 83`, and `His 102` of barnase. All four residues are highly important in the interaction between barnase and barstar.

Table 4.6 lists the values of our global measure for the individual functions, the three pairs, and the triplet. Initial observations show that regions where our local compar-

ison measure between f_N and f_S is high correspond to salt bridges/strong hydrogen bonds. Figure 4.11 shows a visualization of the local contributions to $\kappa(f_N, f_S)$. The colored dots in the figure indicate high values of κ values, namely those in the range $[0.002, 0.0207]$ and are mapped from blue to red. The dots with values lower than 0.002 are not displayed. The gold lines indicate the hydrogen bonds corresponding to those regions of space.

4.4.4 Robustness

In order to get a feeling of the sensitivity of our measure to noise in the data, we compare the values of κ for both the synthetic functions as well as the combustion data after introducing noise with varying amplitudes. The noise is introduced in each function as follows: let R be the maximum allowed amplitude for the noise. R is specified as a percentage of the range of the function. Random noise r is introduced at each vertex as a uniform distribution in the interval $[-R, R]$. The new function value at each vertex is obtained by adding $(r/100)$ times the range of the function to the original value. Table 4.7 shows the error introduced in the values of κ computed

		cup	sad	sin	cos	abs
cup	1%	0.00	0.28	0.03	0.07	0.07
	5%	0.00	8.31	0.98	1.11	2.36
sad	1%	0.28	0.00	0.01	0.02	0.07
	5%	8.31	0.00	0.34	0.35	1.85
sin	1%	0.03	0.01	0.00	0.00	0.00
	5%	0.98	0.34	0.00	0.02	0.11
cos	1%	0.07	0.02	0.00	0.00	0.01
	5%	1.11	0.35	0.02	0.00	0.01
abs	1%	0.07	0.07	0.00	0.01	0.00
	5%	2.36	1.85	0.11	0.12	0.00

Table 4.7: Error introduced in κ when 1% and 5% random noise is added to the synthetic functions.

for the synthetic functions. Figure 4.12 shows the plot of κ for both the original

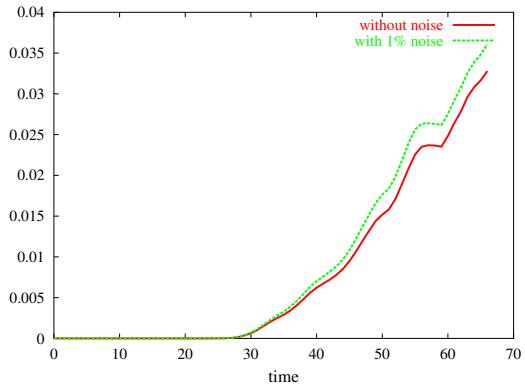

Figure 4.12: κ computed for the original pair and perturbed pair of functions (with 1% noise) in the combustion dataset. There is only a small difference between the graph plots.

pair of functions as well as the pair of perturbed functions (with noise percentage $R = 1\%$). Note that the noise causes only a minor change in the plot.

4.5 Discussion

Various questions related to the extension of our comparison measures remain open. We mention three problems:

- Our definition restricts the number of functions to at most the dimension of the manifold. It would be interesting to extend it to the case $k > d$.
- For the particular case $k = d = 2$, we give alternate interpretations of κ using Jacobi sets. These interpretations generalize to the case $k = d > 2$, but what about $k < d$?
- What is the sensitivity of our measure to the triangulation of the manifold? A detailed understanding of this question is useful in situations where functions

are given on different triangulations of the same manifold. All functions need to be first specified over a common mesh, ideally having a small size, before we are able to compute κ.

We use our visualization tool for a local comparison of two functions measured during the simulation of a combustion process and are able to track the front of the flames after ignition. In future, we want to extend the software to be able handle data in 3D.

Chapter 5

Conclusions

We use a topological approach to develop new methods for extracting features from scientific data. We argue that it is no longer viable to study the properties of this data using traditional visualization techniques because the datasets are becoming increasingly complex . Automatic computation of features followed by a visualization of the function annotated with these features is helpful in understanding the behavior of the function. We restrict our attention to scalar functions measured at discrete points in space and linearly interpolated elsewhere. All our algorithms are combinatorial in nature and numerical issues are handled by a simulation of the smooth setting thereby leading to a robust and efficient implementation. The main contributions of this thesis are:

- an algorithm for simplifying a 3D scalar function that improves the quality of the underlying mesh and a study of the preservation of topological features during this simplification process;
- the introduction of Morse-Smale complexes for piecewise linear 3-manifolds and a combinatorial algorithm to compute them;
- the development of a visualization tool that displays the Morse-Smale complex;
- new local and global measures for comparing scalar functions that are defined over a common manifold domain;
- development of software for visual comparison of multiple, and possibly time-varying, scalar functions.

Open problems raised by the work presented in this dissertation are discussed at the end of the appropriate chapters. The success of the methods proposed in this thesis when applied to large models depends on how they are extended to two important paradigms: the data streaming model and the I/O model. We end this chapter by discussing these extensions.

With rapidly increasing data sizes, many of the high performance computing systems built today are based on a client-server architecture. The server is typically a supercomputer that performs the expensive computation and streams data to the various clients that have limited resources. Data transfer and conversion are the major bottlenecks in these systems. The data stream model abstracts this system and is used to develop efficient schemes for data transfer. The visualization of Morse-Smale complexes in this model is non-trivial because it is possible that the number of critical points is large. A multi-resolution representation of the Morse-Smale complex needs to be developed. Note that the different resolutions are required both in domain as well as function space.

The I/O model analyzes the efficiency of an algorithm by counting the number of disk read/write operations as opposed to the number of comparisons or computation steps. While working with large datasets that do not fit into the main memory, it pays to be conscious about the expensive disk operations. I/O efficient algorithms have been developed in the past within the field of visualization [1]. The popular approaches to the design of such algorithms include a change of the data layout or the use of divide and conquer techniques. Developing I/O efficient versions of the algorithms described in this thesis will be a set of interesting research projects.

Appendix A

Algebraic Topology Basics and Morse Theory

In an effort to ensure that this dissertation is self-contained, we collect the definitions of various terms used in previous chapters and present them in this appendix. Most of these definitions are from topology. We group them into manifolds, simplicial complexes, and algebraic topology in Sections A.1, A.2, and A.3 respectively. We also introduce some Morse theoretic concepts both for smooth functions defined on 3-manifolds (in Section A.1) and for the piecewise linear category (in Section A.3). We refer to the books by Matsumoto [65] and Milnor [68] for further background on Morse theory for smooth functions and to the books by Munkres [70] and Alexandrov [3] for background on related concepts from algebraic and combinatorial topology.

We deal with scalar functions in this thesis. Let us first define them before moving on. A function is a relation which uniquely associates members of one set with members of another set. A *set* is a finite or infinite collection of objects in which order has no significance, and multiplicity is not allowed. The objects that constitute the set are called *elements*. Formally, a *function* $f : X \rightarrow Y$ associates each element in set X with a unique element of set Y. X is called the *domain* and Y the $co-domain$ of the function. *Scalar functions* have a one-dimensional co-domain and *vector functions* have a two- or higher-dimensional co-domain. We are interested in functions where the domain looks like the Euclidean space (at least locally) and the co-domain is the one-dimensional real line, $\mathbb{R}$.

A.1 Manifolds

Definition A.1 (Homeomorphic Spaces) Two topological spaces X and Y are *homeomorphic* or have the same *topological type* if there is a homeomorphism $X \to Y$.

Definition A.2 (k-Manifold) A topological space M is a *k-manifold* if every point $x \in M$ has an open neighborhood homeomorphic to $\mathbb{R}^k$.

Definition A.3 (k-Manifold with Boundary) A topological space N is a *k-manifold with boundary* if every point $x \in N$ has an open neighborhood homeomorphic to $\mathbb{R}^k$ or to the *closed halfspace*, $\mathbb{H}^k = \{(x_1, x_2, \ldots, x_k) \in \mathbb{R}^k \mid x_1 \geq 0\}$.

Definition A.4 (Boundary of a k-Manifold with Boundary) The boundary, $\mathrm{Bd}\, N$, of a k-manifold with boundary is a subset consisting of points whose neighborhood is homeomorphic to $\mathbb{H}^k$.

Let $\mathbb{M}$ be a smooth compact 3-manifold without boundary. Examples are the 3-sphere, which consists of all points at unit distance from the origin in $\mathbb{R}^4$, and the 3-torus, which can be obtained by identifying opposite square faces of a three-dimensional cube. Let $f : \mathbb{M} \to \mathbb{R}$ be a smooth map. The differential of f at a point $p \in \mathbb{M}$ is a linear map from the tangent space at p to $\mathbb{R}$, $\mathrm{d}f_p : \mathrm{T}\mathbb{M}_p \to \mathbb{R}$. The criticality of p is determined by the differential of f at p.

Definition A.5 (Critical Point) A point $p \in \mathbb{M}$ is *critical* if $\mathrm{d}f_p$ is the zero map.

Definition A.6 (Regular Point) A point $p \in \mathbb{M}$ is *regular* if $\mathrm{d}f_p$ is not the zero map.

Given a local coordinate system, the *Hessian* at p is the matrix of second order partial derivatives:

$$H(p) \;=\; \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(p) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(p) & \frac{\partial^2 f}{\partial x_1 \partial x_3}(p) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(p) & \frac{\partial^2 f}{\partial x_2^2}(p) & \frac{\partial^2 f}{\partial x_2 \partial x_3}(p) \\ \frac{\partial^2 f}{\partial x_3 \partial x_1}(p) & \frac{\partial^2 f}{\partial x_3 \partial x_2}(p) & \frac{\partial^2 f}{\partial x_3^2}(p) \end{bmatrix}.$$

Definition A.7 (Non-Degenerate Critical Point) A critical point p is *non-degenerate* if the Hessian at p is non-singular.

Definition A.8 (Morse Function) A function f is called a *Morse function* if all of its critical points are non-degenerate and $f(p) \neq f(q)$ whenever $p \neq q$ are critical.

Lemma A.1 (Morse Lemma) If p is non-degenerate we can choose local coordinates and signs such that

$$f(x_1, x_2, x_3) \;=\; f(p) \pm x_1^2 \pm x_2^2 \pm x_3^2$$

in a local neighborhood of p.

critical point	index
minimum	0
1-saddle	1
2-saddle	2
maximum	3

Table A.1: The index for each type of critical point.

Note that the Morse lemma implies that non-degenerate critical points are isolated. The number of minuses is the *index* of the critical point. It is independent of the coordinate system and equals the number of negative eigenvalues of $H(p)$. In three dimensions, there are four types of non-degenerate critical points: *minima*, 1-*saddles*,

2-*saddles*, and *maxima*. The index for each type of critical point is shown in Table A.1. We get intuitive local pictures by drawing a small sphere around the point p. The level curve of points x with $f(x) = f(p)$ decomposes the sphere into *oceans*, consisting of points x with $f(x) < f(p)$, and *continents*, consisting of points x with $f(x) > f(p)$. Figure A.1 shows the local pictures of a regular point and of the four types of non-degenerate critical points.

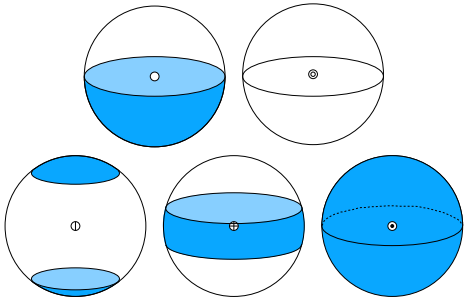

Figure A.1: The local pictures with shaded oceans and white continents of a regular point, a minimum, a 1-saddle, a 2-saddle, and a maximum. Take notice of the symbols used to mark the different types of vertices at the centers of the spheres.

A.2 Simplicial Complexes

Definition A.9 (k-Simplex) A *k-simplex* σ is the convex hull of $k+1$ affinely independent points.

Definition A.10 (Face) A *face* τ of a simplex σ is the simplex defined by a nonempty subset of the $k+1$ points and is denoted as $\tau \leq \sigma$.

Definition A.11 (Coface) A simplex σ is a *coface* of τ if τ is a face of σ.

Definition A.12 (Cone) The *cone* from a vertex x to a k-simplex σ is the convex hull of x and σ, which is the $(k+1)$-simplex $x\sigma$. The operation is defined only if x is not an affine combination of the vertices of σ.

Definition A.13 (Simplicial Complex) A *simplicial complex* K is a finite collection of non-empty simplices for which $\sigma \in K$ and $\tau \leq \sigma$ implies $\tau \in K$ and $\sigma_1, \sigma_2 \in K$ implies that the intersection $\sigma_1 \cap \sigma_2$ is either empty or a face of both, σ_1 and σ_2.

Definition A.14 (Underlying Space) The *underlying space* of K is the union of simplices: $|K| = \bigcup_{\sigma \in K} \sigma$.

Definition A.15 (Triangulation) A *triangulation* of a topological space X is a simplicial complex K whose underlying space is homeomorphic to X.

Definition A.16 (Closure) The *closure* of a subset L of a simplicial complex K is the smallest subcomplex of K that contains L

$$\overline{L} = \{\tau \in K \mid \tau \leq \sigma \in L\}.$$

Definition A.17 (Star) The *star* of a subset L is the set of cofaces of simplices

$$\operatorname{St} L = \{\sigma \in K \mid \sigma \geq \tau \in L\}.$$

Definition A.18 (Link) The *link* of a subset L is the set of all faces of simplices in its star that are disjoint from simplices in L

$$\operatorname{Lk} L = \overline{\operatorname{St} L} - \operatorname{St} \overline{L}.$$

Let K be a simplicial complex that triangulates the k-manifold $\mathbb{M}$. This means there is a homeomorphism between $\mathbb{M}$ and the underlying space of K, but to simplify the discussion, we assume that $\mathbb{M}$ *is* the underlying space. Let $f : \mathbb{M} \to \mathbb{R}$ be

a continuous real-valued function that is linear on every simplex of K. To say this more formally, we note that every point x in a simplex is a unique convex combination of its vertices u_ℓ: $x = \sum_\ell \lambda_\ell u_\ell$ with $1 = \sum_\ell \lambda_\ell$ and $\lambda_\ell \geq 0$ for all ℓ. Assuming f is given at the vertices, we have $f(x) = \sum_\ell \lambda_\ell f(u_\ell)$. We refer to f as a height function and use relative terms such as 'upper' and 'lower' to identify subsets of the star and link of a vertex.

Definition A.19 (Lower Star) The *lower star* of a vertex u contains all simplices in the star for which u is the highest vertex.

$$\mathrm{St}_- u = \{\tau \in \mathrm{St}\, u \mid x \in \tau \Longrightarrow f(x) \leq f(u)\}.$$

Definition A.20 (Upper Star) The *upper star* of a vertex u contains all simplices in the star for which u is the lowest vertex.

$$\mathrm{St}^+ u = \{\tau \in \mathrm{St}\, u \mid x \in \tau \Longrightarrow f(x) \geq f(u)\}.$$

Definition A.21 (Lower Link) The *lower link* contains all simplices in the link that are faces of the lower star.

$$\mathrm{Lk}_- u = \{v \in \mathrm{Lk}\, u \mid v \leq \tau \in \mathrm{St}_- u\}.$$

Definition A.22 (Upper Link) The *upper link* contains all simplices in the link that are faces of the upper star.

$$\mathrm{Lk}^+ u = \{v \in \mathrm{Lk}\, u \mid v \leq \tau \in \mathrm{St}^+ u\}.$$

Let f denote a piecewise linear function give at vertices of a 3-dimensional simplicial complex and linearly interpolated within each simplex. Strictly speaking, critical points of f are not defined, but we may use small bump functions and think of f

as the limit of a series of smooth maps. This is the intuition we use to transport concepts and results from the smooth to the piecewise linear category. We use the topology of the lower link to distinguish regular from critical vertices and to classify the latter.

A.3 Algebraic Topology

The topology of the lower link is expressed using ranks of reduced homology groups. We explain this after giving some background on chain complexes and homology groups. Hatcher [46] gives a good introduction to homology groups and we refer to Artin [5] for more background on group theory.

Definition A.23 (Group) A *group* is a set G together with a binary operation $+ : \mathsf{G} \times \mathsf{G} \to \mathsf{G}$ that is associative and has an identity element and each element of G has an inverse element.

Definition A.24 (Abelian Group) A *abelian group* is a group whose operation is commutative.

Definition A.25 (Subgroup) A subset $\mathsf{H} \subseteq \mathsf{G}$ of a group $(\mathsf{G}, +)$ is called a *subgroup* if $(\mathsf{H}, +)$ is a group.

The infinite set of integers with addition, $(\mathbb{Z}, +)$ is an abelian group. The set of even numbers with addition is a subgroup of $(\mathbb{Z}, +)$. The finite set of non-negative integers less than k together with addition modulo k, $(\mathbb{Z}_k, +_{\bmod k})$, is a finite abelian group. Let $(\mathsf{G}, +)$ be an abelian group and $(\mathsf{H}, +)$ be a subgroup. We can partition G using equivalence classes, called cosets, generated by the *congruence* relation

$$a \equiv b \text{ if } b = a + h, \text{ for some } h \in \mathsf{H}.$$

Definition A.26 (Coset) A coset is a subset of the form $x + \mathsf{H} = \{x + h \mid h \in \mathsf{H}\}$.

Two cosets can be added using their representatives. It does not matter which representative is chosen for addition. The resulting coset is always the same.

Definition A.27 (Quotient Group) The *quotient group* $\mathsf{G}|\mathsf{H}$ is the collection of cosets with addition defined by $(x + \mathsf{H}) + (y + \mathsf{H}) = (x + y) + \mathsf{H}$.

Note that there is a bijective map between H and each coset sending h to $x + h$. Further, if G is finite then all cosets have the same size. Now, since the congruence relation generates a partition of G, we have the following relationship between the cardinalities of G, H, and $\mathsf{G}|\mathsf{H}$:

$$\operatorname{card} \mathsf{G}|\mathsf{H} = \operatorname{card} \mathsf{G}/\operatorname{card} \mathsf{H}.$$

Definition A.28 (Group Homomorphism) A *homomorphism* between groups G and H is a function $h : \mathsf{G} \to \mathsf{H}$ that commutes with addition: $h(x+y) = h(x)+h(y)$.

Definition A.29 (Kernel of Homomorphism) The *kernel* of a homomorphism h between groups G and H is the subset of G that is mapped to the identity element $0 \in \mathsf{H}$.

Definition A.30 (Image of Homomorphism) The *image* of a homomorphism h between groups G and H is the subset of H whose elements have preimages in G.

Definition A.31 (Isomorphism) An *isomorphism* between groups G and H is a bijective homomorphism.

G and H are said to be *isomorphic* ($\mathsf{G} \cong \mathsf{H}$) if there is an isomorphism between them.

Groups can be constructed on collections of simplices by defining an addition operation. We restrict ourselves to addition modulo 2.

Definition A.32 (k-Chain) A *k-chain* is a subset of k-simplices.

The *sum* of two k-chains is the symmetric difference of the two sets:

$$c + d = (c \cup d) - (c \cap d).$$

This is addition modulo 2 because a simplex lies in $c + d$ iff it belongs to exactly one of c or d. Let C_k be the set of k-chains and $(\mathsf{C}_k, +)$ the group of k-chains.

Definition A.33 (Boundary of a Simplex) The *boundary* of a k-simplex is the set of its $(k-1)$-simplex faces: $\partial\sigma = \{\tau \leq \sigma \mid \dim \tau = \dim \sigma - 1\}$.

Definition A.34 (Boundary of a k-Chain) The *boundary* of a k-chain is the sum of boundaries of its simplices: $\partial c = \sum_{\sigma \in c} \partial\sigma$.

We connect chain groups of different dimensions by homomorphisms ∂_k that map chains C_k to their boundary C_{k-1}.

Definition A.35 (Chain Complex) The *chain complex* of a simplicial complex K is the sequence of its chain groups connected by boundary homomorphisms,

$$\ldots \xrightarrow{\partial_{k+2}} \mathsf{C}_{k+1} \xrightarrow{\partial_{k+1}} \mathsf{C}_k \xrightarrow{\partial_k} \mathsf{C}_{k-1} \xrightarrow{\partial_{k-1}} \ldots$$

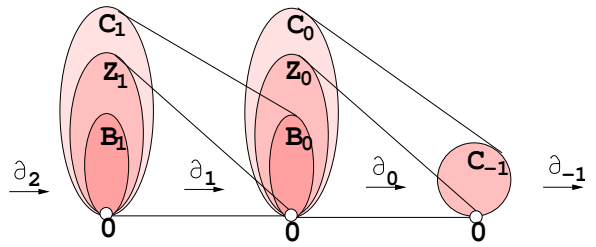

Figure A.2: The chain complex and the connecting boundary homomorphisms.

Two types of chains are important for defining homology groups: the ones without boundary and the ones that bound.

Definition A.36 (k-Cycle) A *k-cycle* is a k-chain c with $\partial c = 0$.

Definition A.37 (k-Boundary) A *k-boundary* is the boundary of a k-chain.

k-cycles and k-boundaries are clearly subgroups of k-chains. The boundary of every k-boundary is empty. This can be proved directly using the definition of the boundary of a k-chain. Figure A.2 shows the chain complex and the nested boundaries and cycles (B_k and Z_k) contained in the chain groups. Since we use reduced homology groups, we extend the list of non-trivial chain groups by adding $\mathsf{C}_{-1} \cong \mathbb{Z}_2$. We define the boundary of a vertex to be $\partial_0(u) = 1$, addition within C_{-1}: $1 + 1 = 0$, and boundary of the non-zero element in C_{-1}: $\partial_{-1}(1) = 0$.

Definition A.38 (k-th Reduced Homology Group) The *k-th reduced homology group* is the quotient defined by the k-cycles and k-boundaries: $\tilde{\mathsf{H}}_k = \mathsf{Z}_k | \mathsf{B}_k$.

The size of $\tilde{\mathsf{H}}_k$ measures the number of k-cycles that are not k-boundaries. The ranks of the homology groups are the most useful aspects of homology groups because they have intuitive interpretations in terms of the connectivity of the space. Given any subset $Y \subseteq \mathsf{G}$, we can form all sums of elements in Y to form a subgroup of G. The subset Y is a *basis* if it is a minimal set that generates the entire group. All bases of G have the same size, called the *rank* of G.

Definition A.39 (k-th Reduced Betti Number) The *k-th reduced Betti number* is the rank of the k-th reduced homology group, $\tilde{\beta}_k = \operatorname{rank} \tilde{\mathsf{H}}_k$.

Since we add modulo 2 for all groups (chains, cycles and boundaries), they are finite and $\tilde{\beta}_k$ is the binary logarithm of the size of $\tilde{\mathsf{H}}_k$. This is true for the ranks of B_k and Z_k too, which

$$\operatorname{rank} \mathsf{H}_k = \operatorname{rank} \mathsf{Z}_k - \operatorname{rank} \mathsf{B}_k.$$

We classify critical points using the reduced Betti numbers of the lower link. The reduced homology groups and Betti numbers ($\tilde{\mathsf{H}}_k$ and $\tilde{\beta}_k$) differ from the more common non-reduced versions (H_k and β_k) only in dimensions 0 and -1. Specifically, $\tilde{\beta}_0 = \beta_0 - 1$ for non-empty lower links, and $\tilde{\beta}_{-1} = 1$ for empty lower links. Since lower links of 2-manifolds are two-dimensional, only $\tilde{\beta}_{-1}$ through $\tilde{\beta}_2$ can be non-zero. As shown in Table A.2, the simple critical points are the ones that have exactly one non-zero reduced Betti number, which is equal to one. The reason that we prefer reduced over non-reduced Betti numbers is this simple correspondence with the index of the critical point. A *multiple saddle* is a vertex that falls outside the classification

	$\tilde{\beta}_{-1}$	$\tilde{\beta}_0$	$\tilde{\beta}_1$	$\tilde{\beta}_2$
regular	0	0	0	0
minimum	1	0	0	0
1-saddle	0	1	0	0
2-saddle	0	0	1	0
maximum	0	0	0	1

Table A.2: The classification of regular and simple critical points using reduced Betti numbers.

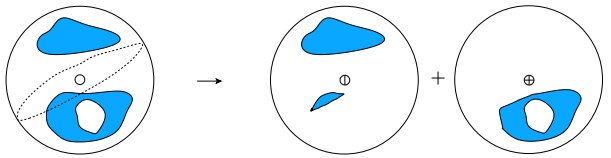

Figure A.3: A multiple saddle is split into a simple 1-saddle and 2-saddle. Note that drawing a circle in the link passing through the other ocean would result in splitting the multiple saddle into a regular vertex and yet another multiple saddle and is hence not useful for unfolding. However, a cut as shown in the figure can always be found.

of Table A.2 and therefore satisfies $\tilde{\beta}_{-1} = \tilde{\beta}_2 = 0$ and $\tilde{\beta}_0 + \tilde{\beta}_1 \geq 2$. It can be unfolded into simple 1-saddles and 2-saddles. One way to do that is to repeatedly cut the link along a circle that intersects the level curve separating the oceans and continents in exactly two points. The reduced Betti numbers on the two sides add up to the

original ones: $\tilde{\beta}_k = \tilde{\beta}_{kL} + \tilde{\beta}_{kR}$, for $k = 0, 1$. We can always choose the circle such that the sum of the reduced Betti number are non-zero on both sides. It follows that the reduction ends after $\tilde{\beta}_0 + \tilde{\beta}_1 - 1$ cuts and generates $\tilde{\beta}_0$ 1-saddles and $\tilde{\beta}_1$ 2-saddles. Figure A.3 shows how a multiple saddle with two continents and two oceans can be unfolded into a 1-saddle and a 2-saddle.

Bibliography

[1] J. Abello and J. S. Vitter. (Eds.) *External Memory Algorithms and Visualization.* DIMACS series in discrete mathematics and theoretical computer science. American Mathematical Society Press, Providence, Rhode Island, 1999. 5

[2] P. K. Agarwal, H. Edelsbrunner, J. Harer and Y. Wang. Extreme elevation on a 2-manifold. *In* "Proc. 20th Ann. Sympos. Comput. Geom., 2004", 357–365. 4.3.2

[3] P. S. Alexandrov. *Combinatorial Topology.* Dover, Mineola, New York, 1998. A

[4] `http://www.ansys.com` 2.1.2

[5] M. Artin. *Algebra.* Prentice Hall, 1991. A.3

[6] C. L. Bajaj, V. Pascucci and D. Schikore. Fast isocontouring for improved interactivity. *In* "Proc. IEEE Sympos. Vol. Viz., 1996", 39–46. 3.1.2

[7] C. L. Bajaj, V. Pascucci and D. Schikore. Visualization of scalar topology for structural enhancement. *In* "Proc. IEEE Conf. Visualization, 1996", 51–58. 1.1

[8] C. L. Bajaj, V. Pascucci and D. Schikore. The contour spectrum. *In* "Proc. IEEE Conf. Visualization, 1997", 167–175. 3.5.2

[9] R. Batra and L. Hesselink. Feature comparisons of 3-D vector fields using earth mover's distance. *In* "Proc. IEEE Conf. Visualization, 1999", 105–114. 4.1.3

[10] H. Blum. *Models for the Perception of Speech and Visual Form.* MIT Press, Cambridge, 1967, 362–380. 3.1.2

[11] P. T. Bremer, H. Edelsbrunner, B. Hamann and V. Pascucci. A multi-resolution data structure for two-dimensional Morse functions. *In* "Proc. IEEE Conf. Visualization, 2003", 139–146. 2.1.2, 2.5, 3.6

[12] V. Bulatov, F. F. Abraham, L. Kubin, B. Devincre and S. Yip. Connecting atomistic and mesoscale simulations of crystal plasticity. *Nature*,

391 (1998), 669–672. 3.5.1

[13] G. Carlsson, A. Zomorodian, A. Collins and L. Guibas. Persistence barcodes for shapes. *In* "Proc. Eurographics Sympos. Geom. Process., 2004". 4.3.2

[14] H. Carr, J. Snoeyink and U. Axen. Computing contour trees in all dimensions. In "Proc. 11th Ann. SIAM-ACM Sympos. Discrete Algorithms, 2000", 918–926. 3.1.2

[15] A. Cayley. On contour and slope lines. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science* **XVIII** (1859), 264–268. 3.1.2

[16] Y. J. Chiang and X. Lu. Progressive simplification of tetrahedral meshes preserving all isosurface topologies. *Computer Graphics Forum* **22** (2003), 493–504. 2.1.2

[17] P. Cignoni, D. Constanza, C. Montani, C. Rocchini and R. Scopigno. Simplification of tetrahedral meshes with accurate error evaluation. *In* "Proc. IEEE Conf. Visualization, 2000", 85–92. 2.1.2, 2.1.3

[18] P. Cignoni, C. Montani, E. Puppo and R. Scopigno. Multiresolution representation and visualization of volume data. *IEEE Trans. Visualization Comput. Graphics* **3** (1997), 352–369. 2.1.2

[19] P. Cignoni, C. Montani and R. Scopigno. A comparison of mesh simplification algorithms. *Computers and Graphics* **22** (1998), 37–54. 2.1.2

[20] T. H. Cormen, C. E. Lieserson and R. L. Rivest. *Introduction to Algorithms.* MIT Press, Cambridge, Massachusetts, 1994. 3.3.3

[21] T. Culver, J. Keyser and D. Manocha. Accurate computation of the medial axis of a polyhedron. *In* "Proc. ACM Sympos. Solid Model. Appl., 1999", 179–190. 3.1.2

[22] T. K. Dey, H. Edelsbrunner, S. Guha and D. V. Nekhayev. Topology preserving edge contraction. *Publ. Inst. Math. (Beograd) (N. S.)* **66** (1999), 23–45. 2.1.3, 2.2.1, 2.2.1

[23] D. P. Dobkin and M. J. Laszlo. Primitives for the manipulation of three-dimensional subdivisions. *Algorithmica* **4** (1989), 3–32. 2.3.1, 3.3.1

[24] EBI Macromolecular Structure Database. `http://www.ebi.ac.uk/msd/` 3.5.1

[25] T. Echekki and J. H. Chen. Direct numerical simulation of autoignition in inhomogeneous hydrogen-air mixtures. *In* "Proc. Second Joint Meeting of the U.S. Sections of the Combustion Institute, 2001". 4.4.3

[26] H. Edelsbrunner. *Geometry and Topology for Mesh Generation.* Cambridge Univ. Press, England, 2001.

[27] H. Edelsbrunner and J. Harer. Jacobi sets of multiple Morse functions. *Foundations of Computational Mathematics, Minneapolis 2002*, 37–57, eds. F. Cucker, R. DeVore, P. Olver, E. Süli, Cambridge Univ. Press, England, 2004. 4.1.2, 4.3.1

[28] H. Edelsbrunner, J. Harer, V. Natarajan and V. Pascucci. Morse-Smale complexes for piecewise linear 3-manifolds. *In* "Proc. 19th Ann. Sympos. Comput. Geom., 2003", 361–370. 1.3, 2.4.6, 2.5

[29] H. Edelsbrunner, J. Harer, V. Natarajan and V. Pascucci. Local and global comparison of continuous functions. *In* "Proc. IEEE Conf. Visualization, 2004". 1.3

[30] H. Edelsbrunner, J. Harer and A. Zomorodian. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. *Discrete Comput. Geom.* **30** (2003), 87–107. 2.5, 3.1.2, 3.2.1, 3.2.2, 3.6, 4.3.2

[31] H. Edelsbrunner, D. Letscher and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.* **28** (2002), 511–533. 2.4.6, 2.5, 3.6, 4.3.2

[32] H. Edelsbrunner and E. P. Mücke. Simulation of Simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graphics* **9** (1990), 66–104. 4.3.5

[33] W. Feller. *An Introduction to Probability Theory and Its Applications. Volume I.* Third edition, John Wiley & Sons, New York, 1968. 4.1.3

[34] A. T. Fomenko and T. L. Kunii. eds. *Topological Modeling for Visualization.* Springer Verlag. 1997. 1.1

[35] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *In* "Proc. SIGGRAPH, 1997", 209–216. 2.1.2, 2.1.3, 2.2.3, 2.2.3

[36] M. Garland and P. S. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. *In* "Proc. IEEE Conf. Visualization, 1998", 263–269. 2.1.2

[37] T. Gerstner and R. Pajarola. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. *In* "Proc. IEEE Visualization, 2000", 259–266. 2.1.2

[38] T. S. Gieng, B. Hamann, K. I. Joy, G. L. Schussman and I. J. Trotts. Constructing hierarchies for triangle meshes. *IEEE Trans. Visualization Comput. Graphics* **4** (1998), 145–161. 2.1.2

[39] A. Globus, C. Levit and T. Lasinski. A tool for visualizing the topology of three-dimensional vector fields. *In* "Proc. IEEE Conf. Visualization, 1991", 33–40. 3.1.2

[40] K. Grauman and T. Darrell. Fast contour matching using approximate earth mover's distance. *In* "Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2004", to appear. 4.1.3

[41] A. Guéziec. Surface simplification with variable tolerance. *In* "Second Ann. Intl. Sympos. Medical Robotics and Computer Assisted Surgery, 1995", 132–139. 2.1.2

[42] A. Guéziec Locally Toleranced Surface Simplification. *IEEE Trans. Visualization Comput. Graphics* **5** (1999), 168–189. 2.1.2

[43] A. Guéziec and R. Hummel. Exploiting triangulated surface extraction using tetrahedral decomposition. *IEEE Trans. Visualization Comput. Graphics* **1** (1995), 328–342. 3.1.2

[44] L. Guibas, R. Holleman and L. E. Kavraki. A probabilistic roadmap planner for flexible objects with a workspace medial axis based sampling approach. *In* "Proc. IEEE/RSJ Intl. Conf. Intell. Robots Systems, 1999", 254–260. 3.1.2

[45] I. Guskov and Z. Wood. Topological noise removal. *In* "Proc. Graphics Interface, 2001", 19–26. 2.1.2

[46] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2001. A.3

[47] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. *In* "SIGGRAPH '97 Course Notes, 1997". 2.1.2

[48] B. Heckel, G. Weber, B. Hamann and K. I. Joy. Construction of vector field hierarchies. *In* "Proc. IEEE Conf. Visualization, 1999", 19–25. 3.6

[49] T. He, L. Hong, A. Varshney and S. W. Wang. Controlled topology simplification. *IEEE Trans. Visualization Comput. Graphics* **2** (1996), 171–184. 2.1.2

[50] J. L. Helman and L. Hesselink. Visualizing vector field topology in fluid flows. *IEEE Computer Graphics & Appl.* **11** (1991), 36–46. 1.1, 3.1.2, 3.6

[51] H. Hoppe. Progressive meshes. *In* "Proc. SIGGRAPH, 1996", 99–108. 2.1.2

[52] H. Hoppe. New quadric metric for simplifying meshes with appearance attributes. *In* "Proc. IEEE Conf. Visualization 1999", 59–66. 2.1.2, 2.2.3

[53] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle. Mesh optimization. *In* "Proc. SIGGRAPH, 1993", 19–26. 2.1.2

[54] C. Johnson, M. Burnett and W. Dunbar. Crystallographic topology and its applications. *In* P. Bourne and K. Watenpaugh (Eds.), *Crystallographic Computing 7: Macromolecular Crystallographic Data*, 1999. 3.1.2

[55] D. B. Karron, J. Cox and B. Mishra. New findings from the spiderWeb algorithm: toward a digital morse theory. *Visualization in Biomedical Computing* **2359** (1994), 643–657. 3.1.2

[56] `http://www.kitware.com` 2.1.2

[57] J. J. Koenderink and A. J. Doorn. The structure of two-dimensional scalar fields with applications to vision. *Biological Cybernetics* **33** (1979), 151–158. 3.1.2

[58] M. van Kreveld, R. van Oostrum, C. Bajaj, V. Pascucci and D. Schikore. Contour trees and small seed sets for iso-surface traversal. *In* "Proc. 13th Ann. Sympos. Comput. Geom., 1997", 212–220. 3.1.2

[59] Y. Lavin. *Topology Based Visualization for Vector and Tensor Fields*. PhD. thesis, Dept. Physics, Stanford Univ. , California, 1998. 1.1

[60] Y. Lavin, R. Batra and L. Hesselink. Feature comparisons of vector fields using earth mover's distance. *In* "Proc. IEEE Conf. Visualization, 2000", 413–415. 4.1.3

[61] W. de Leeuw and R. van Liere. Collapsing flow topology using area metrics. *In* "Proc. IEEE Conf. Visualization, 1999", 349–354. 3.6

[62] P. Lindstrom and G. Turk. Fast and memory efficient polygonal simplification. *In* "Proc. IEEE Conf. Visualization, 1998", 279–286. 2.1.3

[63] S. K. Lodha, J. C. Renteria and K. M. Roskin. Topology preserving compression of 2D vector fields. *In* "Proc. IEEE Conf. Visualization, 2000", 343–350. 3.6

[64] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *Comput. Graphics* **21**, Proc. siggraph 1987, 163–169. 3.1.2

[65] Y. Matsumoto. *An Introduction to Morse Theory.* Translated from Japanese by K. Hudson and M. Saito, Amer. Math. Soc., 2002. 3, A

[66] S. V. Matveyev. Approximating of isosurface in the marching cube: ambiguity problem. *In* "Proc. IEEE Conf. Visualization, 1994", 299–292. 3.1.2

[67] J. c. Maxwell. On hills and dales. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science* **XL** (1870), 421–427. 3.1.2

[68] J. Milnor. *Morse Theory.* Princeton Univ. Press, New Jersey, 1963. 3, A

[69] E. P. Mücke. *Shapes and Implementations in Three-Dimensional Geometry.* PhD. thesis, Dept. Comput. Sci., Univ. Illinois, Urbana, Illinois, 1993. 2.3.1, 2.4.7

[70] J. R. Munkres. *Elements of Algebraic Topology.* Addison-Wesley, Redwood City, California, 1984. A

[71] B. K. Natarajan On generating topologically consistent isosurfaces from uniform samples. *Visual Computer*, **11** (1994), 52–62. 3.1.2

[72] V. Natarajan and H. Edelsbrunner Simplification of three-dimensional density maps. *IEEE Trans. Visualization Comput. Graphics* **10** (2004), 587–597. 1.3

[73] G. M. Nielson and B. Hamann. The asymptotic decider: Resolving the ambiguity of marching cubes. *In* "Proc. IEEE Conf. Visualization, 1991", 83–91. 3.1.2

[74] R. L. Ogniewicz. Skeleton-space: A multi-scale shape description combining region and boundary information. *In* "Proc. Comput. Vision Pattern Recogn., 1994", 746–751. 3.1.2

[75] V. Pascucci and K. Cole-McLaughlin. Parallel computation of the topology of level sets. *Algorithmica* **38** (2003), 249–268. 3.1.2

[76] J. L. Pfaltz. Surface networks. *Geographical Analysis.* **8** (1976), 77–93. 3.1.2

[77] J. Popovic and H. Hoppe. Progressive simplicial complexes. *In* "Proc. SIGGRAPH, 1997", 217–224. 2.1.2

[78] S. Rana (Ed.) *Topological Data Structures for Surfaces: An Introduction to Geographical Information Science.* Wiley, 2004. 3.1.2

[79] G. Reeb. Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique. *Comptes Rendus de L'Académie ses Séances, Paris* **222** (1946), 847–849. 3.1.2

[80] K. J. Renze and J. H. Oliver. Generalized unstructured decimation. *IEEE Computer Graphics & Appl.* **16** (1996), 24–32. 2.1.2

[81] J. Rossignac and P. Borrel. Multi-resolution 3D approximations for rendering complex scenes. *Modeling in Computer Graphics: Methods and Applications*, Springer-Verlag, eds. B. Falcidieno and T. Kunii, 1993, 455–465. 2.1.2

[82] Y. Rubner, C. Tomasi and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision* **40** (2000), 99–121. 4.1.3

[83] C. San Martin, M. Radermacher, B. Wolpensinger, A. Engel, C. S. Miles, N. E. Dixon and J. M. Carazo Three-dimensional reconstructions from cryoelectron microscopy images reveal an intimate complex between helicase DnaB and its loading partner DnaC. *Structure* **6** (1998), 501–509. 3.5.1

[84] W. J. Schroeder, J. A. Zarge and W. E. Lorensen. Decimation of triangle meshes. *In* "Proc. SIGGRAPH **26**, 1992", 65–70. 2.1.2

[85] M. Schwartz. *Morse Homology.* Birkhäuser Verlag, Basel, 1993. 3.1.2

[86] D. Sheehy, C. Armstrong and D. Robinson. Shape description by medial axis construction. *IEEE Trans. Visualization Comput. Graphics* **2** (1996), 62–72. 3.1.2

[87] A. Sheffer, M. Etzion, A. Rappoport and M. Bercovier. Hexahedral mesh generation using the embedded Voronoi graph. *Engineering Comput.* **15** (1999), 248–262. 3.1.2

[88] S. Smale. Morse inequalities for a dynamical system. *Bull. Amer. Math. Soc.* **66** (1960), 43–39. 3.1.2

[89] S. Smale. The generalized poincaré conjecture in higher dimensions. *Bull. Amer. Math. Soc.* **66** (1960), 373–375. 3.1.2

[90] S. Smale. On gradient dynamical systems. *Ann. of Math.* **74** (1961), 199–206. 3.1.2

[91] S. Smale. Generalized poincaré's conjecture in dimensions greater than four. *Ann. of Math.* **74** (1961), 391–406. 3.1.2

[92] M. Spivak. *A Comprehensive Introduction to Differential Geometry* **I**. Publish or Perish, TX. Third edition, 1999.

[93] O. G. Staadt and M. H. Gross. Progressive tetrahedralizations. *In* "Proc. IEEE Conf. Visualization, 1998", 297–402. 2.1.2

[94] D. Storti, G. Turkiyyah, M. Ganter, C. Lim and D. Stal. Skeleton-based modeling operations on solids. *In* "Proc. ACM Sympos. Solid Model. Appl., 1997", 141–154. 3.1.2

[95] J. W. H. Tangelder and R. C. Veltkamp. Polyhedral Model Retrieval Using Weighted Point Sets. *International Journal of Image and Graphics* **3** (2003), 209–229. 4.1.3

[96] S. Tarasov and M. N. Vyali. Construction of contour trees in 3D in $O(n \log n)$ steps. *In* "Proc. 14th Ann. Sympos. Comput. Geom., 1998", 68–75. 3.1.2

[97] `http://www.tgs.com` 2.1.2

[98] R. Thom. Sur une partition en cellules associée à une fonction sur une variété. *Comptes Rendus de L'Académie de Sciences, Paris* **228** (1949), 973–975. 1.3, 3.1.2

[99] X. Tricoche. *Vector and Tensor Field Topology Simplification, Tracking, and Visualization.* PhD. thesis, Schriftenreihe Fachbereich Informatik (3), Universität Kaiserslautern, Germany, 2002. 1.1, 3.6

[100] X. Tricoche, G. Scheuermann and H. Hagen. A topology simplification method for 2D vector fields. *In* "Proc. IEEE Conf. Visualization, 2000", 359–366. 3.6

[101] I. J. Trotts, B. Hamann, K. I. Joy and D. F. Wiley. Simplification of tetrahedral meshes. *In* "Proc. IEEE Conf. Visualization, 1998", 287–295. 2.1.2

[102] A. Van Gelder, V. Verma and J. Wilhelms. Volume decimation of irregular tetrahedral grids. *Computer Graphics International* (1999), 222–230. 2.1.2

[103] A. Van Gelder and J. Wilhelms. Topological considerations in isosurface generation. *ACM Trans. Graphics* **13** (1994), 337–375. 3.1.2

[104] Visualization Toolkit. `http://www.vtk.org` 2.1.2

[105] W. Warntz. The topology of a socio-economic terrain and spatial flows. *Papers of the Regional Science Association* **17** (1966), 47–61. 3.1.2

[106] S. H. Weintraub *Differential Forms: A Complement to Vector Calculus.* Academic Press, 1996. 4.2

[107] J. Wilhelms and A. van Gelder. Topological consideration in isosurface generation. *ACM Trans. Comput. Graphics* **24** (1990), 57–62. 3.1.2

[108] D. Wojtaszek and R. Laganiere. Tracking and recognizing people in colour using the earth mover's distance. *In* "Proc. IEEE Intl. Workshop on Haptic Virtual Environments and their Applications, 2002", 91–96. 4.1.3

[109] G. W. Wolf. A mathematical model of cartographic generalization. *Geoprocessing* **2** (1984), 271–286. 3.1.2

[110] A. Y. Wu, S. K. Bhaskar and A. Rosenfeld. Computation of geometric

properties from the medial axis transform in $O(n \lg n)$ time. *Comput. Vision, Graphics, Image Process.* **34** (1986), 76–92. 3.1.2

[111] Y. Zhou, W. Chen and Z. Tang. An elaborate ambiguity detection method for constructing isosurfaces within tetrahedral meshes. *Computers and Graphics* **19** (1995), 355–364. 3.1.2

[112] A. Zomorodian. *Computing and Comprehending Topology: Persistence and Hierarchical Morse Complexes.* PhD. thesis, Dept. Comput. Sci., Univ. Illinois, Urbana, Illinois, 2001. 1.1

Index

Biography

Vijay Natarajan was born in Tuticorin, India on May 11, 1977. He received the B.E. degree in computer science and M.Sc. degree in mathematics, both from Birla Institute of Technology and Sciences, Pilani, India in 1999. He then began his graduate studies in computer science at Duke University in 1999.

Vijay was awarded a fellowship by the department of computer science, Duke university in 1999-2000. His research interests include computational geometry, computational topology, scientific visualization, geometric modeling and meshing. He has published and presented his research findings in several journals and conferences.

1. Local and global comparison of continuous functions. Written with H. Edelsbrunner, J. Harer, and V. Pascucci. *In* "Proc. IEEE Conf. Visualization, 2004".
2. Simplification of three-dimensional density maps. Written with H. Edelsbrunner. *IEEE Transactions on Visualization and Computer Graphics*, 2004.
3. Loops in Reeb graphs of 2-manifolds. Written with K. Cole-Mclaughlin, H. Edelsbrunner, J. Harer, and V. Pascucci. *Discrete and Computational Geometry*, to appear.
4. Loops in Reeb graphs of 2-manifolds. Written with K. Cole-Mclaughlin, H. Edelsbrunner, J. Harer, and V. Pascucci. *In* "Proc. ACM Symposium on Computational Geometry, 2003".
5. Morse-Smale complexes for piecewise linear 3-manifolds. Written with H. Edelsbrunner, J. Harer, and V. Pascucci. *In* "Proc. ACM Symposium on Computational Geometry, 2003".