Attila Gabor Gyulassy
December 2008
Computer Science

Combinatorial Construction of Morse-Smale Complexes

Abstract

Scientific data is becoming increasingly complex, and sophisticated techniques are required for its effective analysis and visualization. The Morse-Smale complex is an efficient data structure that represents the complete gradient flow behavior of a scalar function, and can be used to identify, order, and selectively remove features. This dissertation presents two algorithms for constructing Morse-Smale complexes in any dimensions. The first algorithm uses persistence-based simplification to remove excess topology from an artificially generated Morse-Smale complex, with important topological features preserved. The second algorithm uses discrete Morse theory to generate an explicit representation of the discrete gradient flow of a scalar function, and uses this representation to compute the Morse-Smale complex directly. This second method enables a divide-and-conquer strategy for handling large data, and is presented in a general framework that admits many common data formats, such as simplicial, gridded, and adaptive multi-resolution (AMR) meshes. Practical considerations are also presented, such as data structures, proper handling of boundary conditions, strategies to accelerate cancellations, and a method to extract a better-quality representation of the topology. A real-world example is also included, where the algorithms and techniques presented in this dissertation are applied to extract the core structure of a porous solid.

Combinatorial Construction of Morse-Smale Complexes for Data Analysis and Visualization

By

ATTILA GABOR GYULASSY
B.A. (University of California, Berkeley) 2003

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Committee in Charge
2008

To Miklós, Györgyi, László, and Katalin

Acknowledgments and Thanks

This dissertation has been made possible by the support of many others, whom I would like to thank here:

First, I would like to thank my advisor Professor Bernd Hamann, and technical mentor Valerio Pascucci. I distinctly remember how targeted Bernd was at my graduate school orientation, zeroing in on me, believing that I had the potential to tackle some really hard problems, and as a result instilling in me a belief that I just might be able to grasp the difficult concepts in this field. From that first day, he has been an unyielding advocate for me, making connections that first led to my receiving a GANN fellowship, then a summer internship at Lawrence Livermore National Laboratory, and finally to be part of the prestigious Lawrence Scholar Program. I am particularly grateful for the improvement he made possible in my technical writing skill through efficient, clear, and untiring revisions. Valerio Pascucci, my technical mentor at Lawrence Livermore National Laboratory, guided the technical side of my development, and his easygoing manner, even in times of huge stress, was an inspiration. Both Bernd and Valerio gave me incredible freedom to explore the topics that interested me, and went above the call of duty to bring me new opportunities for further exploration.

I would like to thank my fellow students and post-docs at UC Davis and LLNL, whose contribution to my work has been immense. In particular, I would like to thank Peer-Timo Bremer and Vijay Natarajan, whose help has been indispensable. They have had a large influence on my thinking. Vijay got me through my first Viz paper, at a time when I had a very dim conception of the work that went into such a publication. Timo has been indispensable for advise, or when I needed to work through a particularly difficult topic.

The staff at LLNL and UC Davis have been a great help in dealing with any administrative matters. In particular, the tireless efforts on behalf of students by Joanna Allen, Linda Becker, Pamela Mears, and James McGraw created a positive environment that fosters the technical growth of students.

I would like to thank my brother Laci for his boundless enthusiasm, and for making Davis feel like home for the first three years of my study. Thanks go also to my family and friends in the area,

who have made my stay at Davis the best time of my life.

Finally, I would like to thank my funding sources. The GANN fellowship supported my work through my first year in graduate school. The Student Employee Graduate Research Fellowship/ Lawrence Scholar Program funded my work for the rest of my time, which gave me the opportunity to focus completely on my research. Additional funding came from: U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, and the National Science Foundation, grant CCF-0702817.

Contents

List of Figures

Abstract

Scientific data is becoming increasingly complex, and sophisticated techniques are required for its effective analysis and visualization. The Morse-Smale complex is an efficient data structure that represents the complete gradient flow behavior of a scalar function, and can be used to identify, order, and selectively remove features. This dissertation presents two algorithms for constructing Morse-Smale complexes in any dimensions. The first algorithm uses persistence-based simplification to remove excess topology from an artificially generated Morse-Smale complex, with important topological features preserved. The second algorithm uses discrete Morse theory to generate an explicit representation of the discrete gradient flow of a scalar function, and uses this representation to compute the Morse-Smale complex directly. This second method enables a divide-and-conquer strategy for handling large data, and is presented in a general framework that admits many common data formats, such as simplicial, gridded, and adaptive multi-resolution (AMR) meshes. Practical considerations are also presented, such as data structures, proper handling of boundary conditions, strategies to accelerate cancellations, and a method to extract a better-quality representation of the topology. A real-world example is also included, where the algorithms and techniques presented in this dissertation are applied to extract the core structure of a porous solid.

Chapter 1

Introduction

Much of modern scientific progress and innovation is enabled by understanding phenomena through exploration of data. A data set is the record of an experimental observation or the output of a simulation process, and in either case it is the quantitative description of some state(s) of a phenomena. As such, scientific data is most often available as a discrete set of values embedded in some underlying space. It can come from virtually any application domain, and therefore data exploration techniques have a potential for far-reaching impact. Data acquisition alone is not sufficient for scientific discovery; it is the analysis of data that provides the answers to fundamental scientific questions. Visualization is the process that takes data from its abstract form and converts it into an image or sequence of images that can convey an immense amount of information efficiently to a trained user by utilizing our human ability to interpret images. As data becomes larger and more complex, however, an image alone can be insufficient to impart the relevant semantics in a data set. In fact, it is becoming a more common practice in visualization to perform an in-between data analysis step, to focus the attention of the user to the interesting parts of the data. In fact, in many applications, the semantics of a data set can be summarized as the answers to simple queries. Using data to answer even simple questions, however, can be extremely difficult.

Consider the following hypothetical example: a geologist with a data set representing a surface scan of a mountain range might reasonably ask how many mountains there are in the data. Providing an answer to this question is surprisingly difficult; the solution depends on how one defines a mountain. It would require qualifications such as: a mountain (versus a simple "bump") has a peak above x feet; two peaks must be apart by more than y feet laterally to be considered separate mountains; peaks separated by saddles less than z feet below the peaks are considered part of the same mountain; and so on. This problem is a relatively simple one: given sufficient time and a reasonable visualization, a trained geologist could identify the mountains in the data by hand. However, oftentimes the features a scientist is interested in are so non-intuitive that they can only be described within some mathematical framework. As computational power continues to increase, data sets become larger and more complex, and a rigorous semi-automated method of describing and identifying features becomes a necessity. This is the fundamental motivation for using topology-based analysis and visualization techniques. A topological framework permits the formalization of

parameters to define a "feature," and hence enables the automation of scientific data analysis.

Scalar functions are ubiquitous in scientific computing and much time and effort are spent in their analysis. For a very wide range of applications, features in scalar functions correspond to either the shape and structure of level sets or the gradient flow behavior. Morse theory provides a well-studied framework for analysis of scalar valued functions. In a Morse function, the topology of level sets is related to the critical points and gradient behavior of the function, and this relationship provides a framework in which rigorous definition and identification of features is possible. The relationship between the critical points and shape of level sets was identified as early as 1859 by Arthur Cayley. Marston Morse formalized the theory in 1963, and since then, Morse theory has matured and its results are widely used in analysis of scalar functions. Recently, Morse theory has been extended to piecewise-linear data by Herbert Edelsbrunner in 2003. A new formulation of the theory for discrete cellular meshes was presented in 2001 by Robin Forman, with a focus on computational solutions to theoretic questions. The extension to piecewise linear functions and discrete meshes allows us to use Morse theory to solve problems practically in real-world data sets.

In this dissertation, we will focus on the Morse-Smale complex as a representation of the topology of a scalar function. As opposed to other topology-based methods, such as contour trees and Reeb graphs, the Morse-Smale complex provides a complete description of the topology of level sets of a function, and additionally provides a segmentation of the domain into regions of uniform gradient flow behavior. Features can therefore be defined rigorously in terms of level sets and the cells of the Morse-Smale complex. Furthermore, simplification of a Morse-Smale complex enables the computation of a multi-resolution representation of the topology of a scalar function: a well defined hierarchy of resolutions, where important features are preserved in all resolutions. Simplification and filtering of the complex is a very powerful tool that can extract features from data in a well-defined and reproducible manner. The cost of using such a powerful analysis tool is significant: computation of a Morse-Smale complex for real-world data has traditionally been challenging.

In this dissertation, we present two algorithms for extracting the Morse-Smale complex for scalar data of any dimensions, as well as simplification and filtering techniques to extract features in practice. Our first algorithm computes the Morse-Smale complex by simplifying a known "ar-

tificial" complex. We show how to improve the efficiency of this approach, and present a case study where we use simplification and filtering of the computed Morse-Smale complex to extract the core structure of a simulated porous material, solving a problem in material sciences. We also present an algorithm based on discrete Morse theory, and show that it enables in-practice computation of the Morse-Smale complex for large data sets. Finally, we discuss some of the continuing challenges in computing Morse-Smale complexes and present practical solutions to these problems. The remainder of this chapter will give an overview of the dissertation and its organization.

1.1 Organization of Dissertation

In chapter 2, the basic mathematical foundations are presented that are necessary for the rest of the dissertation. This includes the definition of sets and functions (section 2.1.1), topology (section 2.1.2), manifolds (section 2.1.3), and CW-complexes (section 2.2).

Next, we present basic definitions and theorems in Morse theory in chapter 3. We introduce Morse functions (section 3.1), and describe several techniques for analysis of these functions, including relevant references. These include handle bodies (section 3.2), Reeb graphs/contour trees (section 3.3), surface segmentations (section 3.4), Morse complexs (section 3.5), Morse-Smale complexes (section 3.6), and Morse-Smale complexes for piecewise-linear domains (section 3.7). At the end of the chapter, we present a definition for topological features, and how a function can be simplified by repeated removal of features (section 3.8).

In chapter 4, we present an algorithm for constructing Morse-Smale complexes through the simplification of an artificial complex. An integral part of this algorithm is the ability to cancel critical points in a three-dimensional Morse-Smale complex. We introduce these cancellation operations (section 4.3), and describe the algorithm (section 4.5). We present some improvements to the algorithm to increase efficiency, and describe the data structures used for completeness (section 4.6).

Chapter 5 shows a practical example of how the Morse-Smale complex can be used for analysis of real-world data. In this example, we identify the core structure of a simulated porous solid as the "stable" 2-saddle-maximum arcs of the Morse-Smale complex. We perform this analysis for

multiple time steps of a simulated particle striking the material to understand how the connectivity and structure of the material is affected.

We present another algorithm based on discrete Morse theory in chapter 6 that computes a discrete Morse-Smale complex. We present the necessary background in discrete Morse theory (section 6.1), and then describe an algorithm that generates a discrete gradient field from a data set where scalar values are given as the vertices of a CW-complex (section 6.4). We use this algorithm in a divide-and-conquer approach for computing the Morse-Smale complex for large data (section 6.5). This approach is presented in a framework that supports data of many formats and of any dimensions (section 6.6).

Construction of a Morse-Smale complex involves complex data structures, and resolution of practical details, such as boundary conditions, and simulation of simplicity. In chapter 7, we discuss some techniques to resolve boundary conditions (section 7.5), accelerate cancellations (section 7.3), and produce a better simulation of simplicity (section 7.4). The techniques in this chapter are in fact crucial for in-practice computation and simplification of the Morse-Smale complex.

1.2 Published Material

The algorithms, techniques, and results in this dissertation have appeared or are to appear as the following published works:

1. A. Gyulassy, V. Natarajan, V. Pascucci, P.T. Bremer and B. Hamann. **Topology-based simplification for feature extraction from 3d scalar fields**. In *Proc. IEEE Conf. Visualization (IEEE Visualization 2005)*, pages 535–542, 2005.

2. A. Gyulassy, V. Natarajan, P.T. Bremer, V. Pascucci, and B. Hamann. **A topological approach to simplification of three-dimensional scalar fields**. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):474–484, 2006.

3. A. Gyulassy, V. Natarajan, V. Pascucci, and B. Hamann. **Efficient computation of Morse-Smale complexes for three-dimensional scalar fields (IEEE Visualization 2007)**. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1440–1447, 2007.

4. A. Gyulassy, M. Duchaineau, V. Natarajan, V. Pascucci, E. Bringa, A. Higgingotham, and B. Hamann. **Topologically clean distance fields**. *IEEE Transactions on Visualization and Computer Graphics (IEEE Visualization 2007)*, 13(6):1432–1439, 2007.

5. A. Gyulassy, P.T. Bremer, V. Pascucci, and B. Hamann. **A practical approach to Morse-Smale complex computation: scalability and generality**. *IEEE Transactions on Visualization and Computer Graphics (IEEE Visualization 2008)*, 14(6):1619–1626, 2008.

6. A. Gyulassy, P.T. Bremer, V. Pascucci, and B. Hamann. **Practical considerations in Morse-Smale complex computation**. *to appear*, 2009.

Chapter 2

Mathematical Foundations

This dissertation presents methods and algorithms that rely on elements of algebraic and differential topology, computational geometry, and scientific visualization. This chapter introduces some of the standard definitions and results in these fields that are most relevant to this dissertation, and provides a mathematical foundation for the rest of this work. These fields have slightly different nomenclatures, and we present definitions and theorems in this chapter to remove any ambiguity. Section 2.1 introduces some basic concepts in sets, functions, topology, and manifolds, and section 2.2 provides definitions regarding CW-complexes.

2.1 Topological Spaces and Manifolds

Topology is the branch of mathematics that studies the properties of spaces that are invariant to continuous deformation. Sets are the building blocks of the study of topology, and functions are the mappings between sets. In this dissertation, we will be studying the structure of functions in terms of the topology of its level sets. Manifolds are topological spaces that provide an effective theoretical framework in which to define functions.

2.1.1 Sets and Functions

A fundamental building block in mathematics is the concept of a set.

Definition 2.1.1. (Set) A **set** S is a finite or infinite collection of objects in which order has no significance, and multiplicity is ignored. Members of a set are often referred to as **elements** and the notation $a \in S$ denotes that a is an element of S. The set without any elements is called the **empty set**, and is denoted $\emptyset$.

Definition 2.1.2. (Subset) A set U is called a **subset** of a set S, denoted $U \subseteq S$, if every element of U is in S. Formally, $a \in U$ implies $a \in S$. A **proper** subset $U \subseteq S$, $U \neq S$ is denoted by $U \subset S$.

A set is a collection of objects, upon which some structure can be imposed through a partition.

Definition 2.1.3. (Partition) A **partition** of a set is a decomposition of the set into subsets (cells) such that every element of the set is in one and only one of the subsets.

The power set represents all possible sets of a collection of objects.

Definition 2.1.4. (Power Set) If A is a set then 2^A denotes the **power set** of A, which is the collection of all subsets of A: $2^A = \{B|B \subseteq A\}$.

Two basic operations on sets are union and intersection.

Definition 2.1.5. (Union, Intersection) The **union** of two sets A, B, is the set of elements belonging to either A or B: $A \cup B = \{x|x \in A \text{ or } x \in B\}$. The **intersection** of A and B is the set of elements that belong to both sets: $A \cap B = \{x|x \in A \text{ and } x \in B\}$.

Elements of different sets can be related by a function.

Definition 2.1.6. (Relation) A **relation** φ between sets A and B is a collection of ordered pairs $[a, b]$ such that $a \in A$ and $b \in B$.

Next are some properties that can be satisfied by a relation.

Definition 2.1.7. (Symmetry, Antisymmetry, Transitivity) A relation φ is called:

1. **symmetric** $(a, b) \in \varphi$, implies that $(b, a) \in \varphi$;
2. **antisymmetric** $(a, b) \in \varphi$ and $(b, a) \in \varphi$, implies that $a = b$;
3. **transitive** $(a, b), (b, c) \in \varphi$, implies that $(a, c) \in \varphi$.

Functions are relations between sets that uniquely map an element in a set to another element.

Definition 2.1.8. (function, map) A **function** or **map** f from A to B is a rule that assigns to each element $a \in A$ exactly one element $b \in B$. We say f maps A into B and f maps a to b, denoted by $f(a) = b$. A function from A to B can be denoted by $f : A \to B$. A is the **domain** of f and B is the **codomain** or **range** of f. The element b is the **image** of a under f and the set $im(f) = f(A) = \{f(a)|a \in A\}$ is the image of A under f.

The following are properties that can be satisfied by a function.

Definition 2.1.9. (injective, surjective, bijective) A function from a set A to a set B is:

1. **injective** or **one-to-one** if each element of $b \in B$ has at most one element mapped into it;

2. **surjective** if each element $b \in B$ has at least one element mapped into it;

3. **bijective** if it is both injective and surjective.

2.1.2 Topology

A topology implies a structure on a collection of sets that is useful for their analysis.

Definition 2.1.10. (Topology) A **topology** on a set A is a subset $T \subseteq 2^A$ such that

1. If $S_1, S_2 \in T$, then $S_1 \cap S_2 \in T$.

2. If $\{S_j | j \in J\} \subseteq T$, then $\bigcup_{j \in J} S_j \in T$.

3. $\emptyset \in T$ and $A \in T$.

A topology describes the connectivity of a set, and divides subsets into open and closed sets.

Definition 2.1.11. (Open, Closed Sets) Let A be a set and T be a topology. All sets $S \in T$ are **open sets**. All other sets $U = A \setminus S$, where $S \in T$ are called **closed sets**.

Definition 2.1.12. (Topological Space) The pair (A, T) of a set A and a topology T is called the topological space $\mathbb{A}$.

Sets of topological spaces are related by the following definitions:

Definition 2.1.13. (Interior, Closure, Boundary) The **interior** $\mathring{A}$ of set $A \subseteq \mathbb{X}$ is the union of all open sets contained in A. The **closure** $\overline{A}$ of A is the intersection of all closed sets containing A. The **boundary** $\delta A = \overline{A} - \mathring{A}$ is the closure minus the interior.

Definition 2.1.14. (Neighborhood) A **neighborhood** of $a \in S$ is any $A \subseteq S$ such that $a \in \mathring{A}$. A **basis of neighborhoods** at $a \in S$ is a collection of neighborhoods of a such that every neighborhood of a contains one of the basis neighborhoods.

Definition 2.1.15. (Induced Topology) For a subset A of a set X with topology T, the **relative** or **induced** topology T_A is defined as $T_A = \{S \cap A | S \in T\}$.

Definition 2.1.16. (Subspace) A subset $A \subseteq X$ with topology T_A is a topological **subspace** of $\mathbb{X}$.

A metric is a function that operates on pairs of elements of a set.

Definition 2.1.17. (Metric) A **metric** or **distance function** $d : S \times S \to \mathbb{R}$ is a function satisfying the following conditions:

1. **Positivity** For all $x, y \in S$, $d(x, y) \geq 0$.
2. **Non-Degeneracy** If $d(x, y) = 0$, then $x = y$.
3. **Symmetry** For all $x, y \in S$, $d(x, y) = d(y, x)$.
4. **Triangle Inequality** For all $x, y, z \in S$, $d(x, y) + d(y, z) \geq d(x, z)$.

Definition 2.1.18. (Open Ball) The **open ball** $B(x, r)$ with center x and radius $r > 0$ with respect to metric d is defined as $B(x, r) = \{y | d(x, y) < r\}$.

Definition 2.1.19. (Metric Space) A set S with a metric function d is called a **metric space**.

A metric space is a topological space where open balls serve as basis neighborhoods for a topology of a set. In practice, applications deal with metric spaces, such as Euclidean spaces.

Definition 2.1.20. (Euclidean Space) The Cartesian product of n copies of R, the set of real numbers, together with the **Euclidean metric** $d(x, y) = \sqrt{\sum_{i=0}^{n}(x_i - y_i)^2}$ is the n**-dimensional Euclidean space** $\mathbb{R}^n$.

The following definition provides a way to compare whether two topological spaces have the same structure.

Definition 2.1.21. (Homeomorphism) A **homeomorphism** is a bijective function $f : \mathbb{X} \to \mathbb{Y}$ such that f and f^{-1} are continuous, where f^{-1} is the inverse of f. If such an f exists, $\mathbb{X}$ is said to be **homeomorphic** to $\mathbb{Y}$, and that $\mathbb{X}$ and $\mathbb{Y}$ have the same topological type.

2.1.3 Manifolds

In practical applications, the topological spaces under consideration are typically subsets of Euclidean space. It is useful to define spaces that locally appear to be a subset of the plane $\mathbb{R}^d$ but are not necessarily homeomorphic to it. These spaces are called manifolds. *Charts* describe the property of being locally homeomorphic to $\mathbb{R}^d$.

Definition 2.1.22. (Chart) A **chart** at $x \in \mathbb{X}$ is a function $\varphi : U \to \mathbb{R}^d$, where $U \subseteq \mathbb{X}$ is an open set containing x, and φ is a homeomorphism onto an open subset of $\mathbb{R}^d$. The **dimension** of the chart φ is d.

Definition 2.1.23. (Hausdorff) A topological space $\mathbb{X}$ is **Hausdorff** if for every $x, y \in \mathbb{X}$, $x \neq y$, there are neighborhoods A, B of x, y, respectively, such that $A \cap B = \emptyset$.

Definition 2.1.24. (Separable) A topological space $\mathbb{X}$ is **separable** if it has a countable basis of neighborhoods.

Definition 2.1.25. (Manifold) A separable Hausdorff space $\mathbb{X}$ is called a (topological) d**-manifold** if at every point $x \in \mathbb{X}$ there is a d-dimensional chart. This means that every point $x \in \mathbb{X}$ has a neighborhood homeomorphic to $\mathbb{R}^d$. When the neighborhood at every point $x \in \mathbb{X}$ is homeomorphic to either $\mathbb{R}^d$ or the Euclidean halfspace $\mathbb{H}^d = \{x \in \mathbb{R}^d | x_0 \geq 0\}$ then $\mathbb{X}$ is called a d**-manifold with boundary**. The **boundary** of $\mathbb{X}$ is the set of points with neighborhood homeomorphic to $\mathbb{H}^d$. The **dimension** of the manifold is d.

Theorem 2.1.26. *The boundary of a d-manifold with boundary is a $(d-1)$-manifold without boundary.*

Definition 2.1.27. (Compact) a **covering** of $A \subseteq X$ is a family of sets $\{C_j | j \in J\}$ in 2^X, such that $A \subseteq \bigcup_{j \in J} C_j$. An **open covering** is a covering consisting of open sets. A **subcovering** of a covering $\{C_j | j \in J\}$ is a covering $\{C_k | k \in K\}$, where $K \subseteq J$. A subspace $\mathbb{A} \subseteq \mathbb{X}$ is **compact** if every open covering of A has a finite subcovering.

Morse theory typically deals with smooth manifolds.

Definition 2.1.28. **(C^∞)** Let U, $V \subseteq \mathbb{R}^n$ be open. A function $f : U \to \mathbb{R}^d$ is **smooth** or **C^∞-continuous** or C^∞ **(continuous of order ∞)** if all coordinate functions $f_0, ..., f_d$ have partial derivatives for all orders and types. Two charts $\varphi : U \to \mathbb{R}^d$, $\psi : V \to \mathbb{R}^d$ are **C^∞-related** if either $U \cap V = \emptyset$ or $\varphi \circ \psi^{-1}$ and $\psi \circ \varphi^{-1}$ are smooth. A C^∞ **atlas** is once for which every pair of charts is C^∞-related. A chart is **admissible** to a C^∞ atlas if it is C^∞-related to every chart in the atlas.

Definition 2.1.29. **(C^∞ Manifold)** A C^∞ **manifold** is a topological manifold together with all the admissible charts of some C^∞ atlas.

In the following, the term "manifold" is used for compact C^∞ manifolds unless explicitly stated. The primary focus are functions defined over n-dimensional manifolds. Manifolds that are not simply connected are considered to be embedded in $\mathbb{R}^{n+1}$.

Definition 2.1.30. **(Embedding)** An embedding $f : \mathbb{X} \to \mathbb{Y}$ is a map whose restriction to its image $f(\mathbb{X})$ is a homeomorphism.

2.2 CW-Complexes

In data analysis and visualization, spaces are rarely represented as smooth manifolds. Instead, discrete approximations are used to represent manifolds. This section describes some general concepts of CW-complexes, and introduces these kinds of meshes as a substitute for manifolds. An in-depth study of CW-complexes can be found in Lundell and Weingram [32]. The definitions in this section are primarily due to Forman [19].

Definition 2.2.1. **(d-Cell)** A **d-cell** α is a topological space which is homeomorphic to a **d-ball**, $\alpha \cong B^d = \{x \in \mathbb{E}^d | \ |x| \leq 1\}$. We say the **dimension** of a d-cell is d.

Cells are the basic building blocks of complexes. They are related to one another based on the behavior of their boundaries.

Lemma 2.2.2. *The boundary of a d-cell α, denoted $\dot{\alpha}$, is the unit $(d-1)$-sphere* $S^{(d-1)} = \{x \in \mathbb{E}^d | \ |x| = 1\}$.

Definition 2.2.3. (Cell) A **cell** is a topological space that is a d-cell for some d.

Attaching is the process by which cells can be glued together to form a complex.

Definition 2.2.4. (Attaching Map) Let $\mathbb{X}$ be a topological space, α a d-cell, and $f : \dot{\alpha} \rightarrow \mathbb{X}$ a continuous map. Let $\mathbb{X} \cup_f \alpha$ denote the disjoint union of $\mathbb{X}$ and α quotiented out by the equivalence relation that each point $s \in \dot{\alpha}$ is identified with $f(s) \in \mathbb{X}$. We say that $\mathbb{X} \cup_f \alpha$ is the result of **attaching** the cell α to $\mathbb{X}$. The map f is called the **attaching map**.

Therefore, an attaching map glues a cell to a space along its entire boundary, and the entire boundary must already exist in that space.

Definition 2.2.5. (Finite CW-Complex) A **finite CW-complex** is a topological space X such that there exists a finite nested sequence

$$\emptyset \subset X_0 \subset X_1 \subset \cdots \subset X_n = X$$

such that for each i = 0, 1, 2, ... , n, X_i is the result of attaching a cell to $X_{(i-1)}$.

This description, using attachment as a means to construct a CW-complex, is due to A. Hatcher [27]. An alternate construction using a decomposition is due to Cooke and Finney [9]. For practical applications, it is useful to restrict our attention to a subset of CW-complexes, where every cell of dimension less than d in a d-dimensional space lies on the boundary of at least one d-cell.

Definition 2.2.6. (Regular CW-Complex) A **regular CW-complex** is a finite CW-complex, where any two incident cells ρ and τ with $dim(\tau) = dim(\rho) - 2$, there are exactly two cells σ_1 and σ_2 such that $\tau < \sigma_1 < \rho$ and $\tau < \sigma_2 < \rho$.

A regular CW-complex has restrictions imposed on the attaching map, preventing some degenerate conditions. Simplicial complexes, gridded meshes, and adaptive multi-resolution meshes are all examples of finite regular CW-complexes. We say the **dimension** of a CW-complex is the maximum dimension of any of its cells.

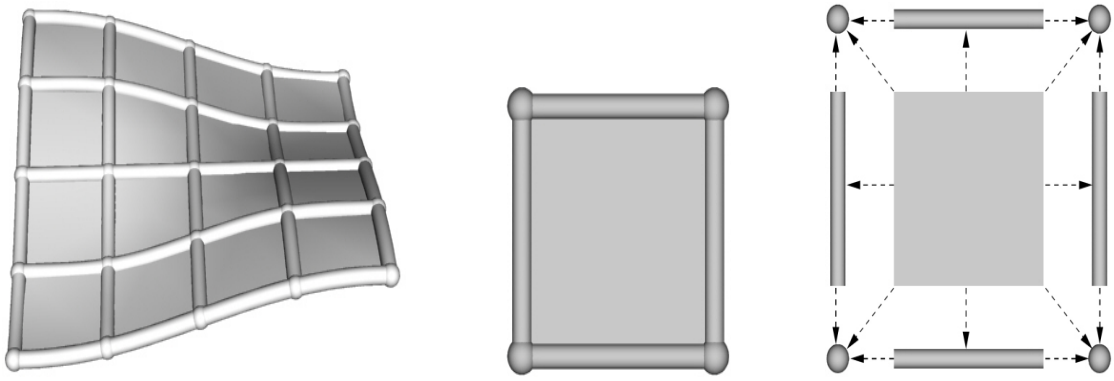
For practical applications and simplicity we will use "CW-complex" in this text to refer to a finite regular CW-complex, unless explicitly stated otherwise.

The following definitions relate the cells of a complex to the cells forming their boundaries in the attachment of that cell.

Definition 2.2.7. (Face, Facet) Let α be a d-cell in a CW-complex K, and $\dot{\alpha}$ be its boundary. The cells $\beta \in K \cap \dot{\alpha}$ are the **faces** of α, and $\beta < \alpha$ denotes that β is a face of α. If the dimension of β is exactly one less than the dimension of α, then we say β is a **facet** of α.

Definition 2.2.8. (Co-face, Co-facet) Let β be a face of α. We say that α is a **co-face** of β. If the dimension of α is exactly one more than the dimension of β, we say α is a **co-facet** of β.

Figure 2.2.1 shows the cells of a regular grid representing a 2-manifold. The faces of quads are edges and vertices, and the faces of edges are vertices.



(a) (b) (c)

Figure 2.2.1: A 2-manifold surface is represented by a mesh of quadrilaterals (a). Arrows point from a cell to its faces (c). The faces of the quad are four edges and four vertices. The faces of edges are two vertices.

Chapter 3

Morse Theory

Morse theory deals with the relationship between the structure of spaces, and functions defined on those spaces. In particular, it presents a class of functions for which certain structural properties hold, and investigates those properties. In the following, we first present Morse functions, a class of functions which have well-studied properties relating their critical points, gradient behavior, and structure of level sets. Next, we present contour trees and Reeb graphs, which make use of these properties of Morse functions to create an abstract representation of the topology of level sets for the purposes of data analysis and visualization. We survey other surface segmentation methods, and then present the definition of a Morse complex and Morse-Smale complex. We end the introduction to Morse theory with a discussion of topological features and simplification.

3.1 Morse Functions

Morse functions are a class of functions for which the behavior of critical points, the gradient, and the topology of level sets are related via simple rules. These functions are ideal for analysis of data: they are *dense* in the space of scalar functions, and they can be represented efficiently in abstract terms, such as with Reeb graphs, and with Morse-Smale complexes. The following definitions and theorems regarding Morse functions are well-known. The definitions and theorems here are due to Morse, Milnor, and Matsumoto [40, 39, 35].

Definition 3.1.1. (Gradient) Given a smooth function $f : \mathbb{R}^n \rightarrow \mathbb{R}$, the **gradient** of f, denoted ∇f, is

$$\nabla f = (\frac{\partial f}{\partial x_0}, \cdots, \frac{\partial f}{\partial x_{n-1}}).$$

The gradient of a scalar function is a vector field that points in the direction of steepest ascent. At each point of a smooth manifold there exists a chart (definition 2.1.22), which defines a local coordinate system in which the gradient can be computed.

Definition 3.1.2. (Critical Point) Let $\mathbb{M}^d$ be a d-manifold and $f : \mathbb{M}^d \rightarrow \mathbb{R}$ be a smooth function defined on it. A point p of $\mathbb{M}$ is a **critical point** of f if $\nabla f(p) = \mathbf{0}$. The point is called **regular** otherwise.

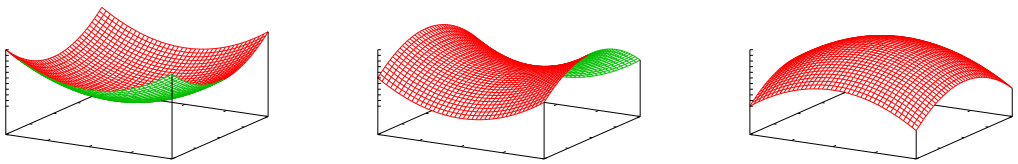


Figure 3.1.1: Local configurations of critical points (minimum, saddle, and maximum) for a function $f : \mathbb{M}^2 \to \mathbb{R}$. A minimum (left) has the standard form $x^2 + y^2$, a saddle (middle) has the form $x^2 - y^2$, and a maximum (right) has the form $-x^2 - y^2$.

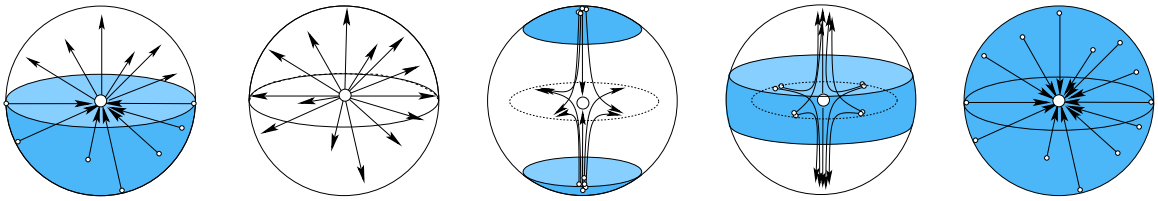


Figure 3.1.2: Local configurations of a regular point and the four types of critical points (minimum, 1-saddle, 2-saddle, and maximum) for a function $f : \mathbb{M}^3 \to \mathbb{R}$. Shaded regions indicate **oceans**, regions where the function value is lower than the critical point and white regions indicated **continents**, regions where the function value is higher than the critical point. Lines of steepest ascent are also drawn to show the gradient behavior in the neighborhood of the critical points.

Critical points of a scalar function occur where the gradient vanishes. They are classified using the second partial derivatives.

Definition 3.1.3. (Hessian Matrix) Let p be a critical point of $f : \mathbb{R}^d \to \mathbb{R}$. The **Hessian matrix** of the function f at p is the $d \times d$ matrix of second partial derivatives:

$$H_f(p) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2}(p) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_d}(p) \\ & \ddots & \\ \vdots & \frac{\partial^2 f}{\partial x_i \partial x_j}(p) & \vdots \\ & & \ddots \\ \frac{\partial^2 f}{\partial x_d \partial x_1}(p) & \cdots & \frac{\partial^2 f}{\partial x_d^2}(p) \end{pmatrix}$$

Definition 3.1.4. (Degenerate, Non-degenerate) A critical point p of a function $f : \mathbb{R}^n \to \mathbb{R}$ is a **degenerate** critical point of f if the Hessian matrix is singular, i.e., $det(H_f(p)) = 0$. It is a **non-degenerate** critical point otherwise.

Note that the determinant of the Hessian matrix of a point is zero if and only if it is zero in all coordinate representations around that point. Furthermore, non-degenerate critical points are *stable* under linear perturbation, while degenerate critical points are *unstable*, *i.e.*, they can be made into non-degenerate critical points by adding a linear perturbation to the function.

Definition 3.1.5. (Morse Function) A smooth function $f : \mathbb{M} \to \mathbb{R}$ is a **Morse function** if all of its critical points are non-degenerate and isolated.

A result of Morse theory is that the Morse function can be expressed around non-degenerate critical points in a *standard form*.

Theorem 3.1.6. ***(Morse Lemma)*** *Let p be a non-degenerate critical point of $f : \mathbb{M}^d \to \mathbb{R}$. Then there exists local coordinates $(X_1, X_2, ..., X_n)$ about p such that the coordinate representation of f with respect to these coordinates has the following **standard form**:*

$$f = -X_1^2 - X_2^2 - ... - X_\lambda^2 + X_{\lambda+1}^2 + ... + X_d^2 + c,$$

where p is the origin $(0, 0, ..., 0)$ and $c = f(p)$.

Figure 3.1.1 shows the standard form locally around the three kinds of critical points for a Morse function defined on a 2-manifold. This is a powerful theorem that states that the function has quadratic behavior in a local neighborhood around every critical point.

Definition 3.1.7. (Index) Let p be a critical point of $f : \mathbb{M}^d \to \mathbb{R}$ and $f = -X_1^2 - X_2^2 - ... - X_\lambda^2 + X_{\lambda+1}^2 + ... + X_d^2 + c$ the standard form at p. Then the number of minus signs, λ is the **index** of p.

The index of a critical point is the same regardless of the local coordinates in the standard form.

Definition 3.1.8. (Level Set) The **level set** of a function $f : \mathbb{M} \to \mathbb{R}$ with respect to a constant c in the range of f is the set $\{x \in \mathbb{M} | f(x) = c\}$.

Level sets are also called contours, iso-contours, and iso-surfaces. For a Morse function defined on d-manifold, the level sets are generally $(d-1)$-manifolds (unless collapsed to a point).

3.2 Handle Bodies

A major result of Morse theory is the ability to define the global shape of a manifold in terms of its critical points.

Theorem 3.2.1. *Let $\mathbb{M}$ be a closed manifold and $f : \mathbb{M} \to \mathbb{R}$ a Morse function on $\mathbb{M}$. Let $\mathbb{M}_t = \{p \in \mathbb{M} | f(p) \leq t\}$ for a value t of f. If f has no critical points in $[a, b] \in \mathbb{R}$, then $\mathbb{M}_a$ and $\mathbb{M}_b$ are diffeomorphic: $\mathbb{M}_a \cong \mathbb{M}_b$.*

This implies that all changes in the topology level sets of f on $\mathbb{M}$ occur at critical points of the function. Since there are a finite number of critical points, we can order them $p_0, p_1, ..., p_n$ in ascending order according to their function values, such that $c_0 < c_1 < ... < c_n$, where $c_i = f(p_i)$. Note that for a general Morse function two critical points may have the same function value, *i.e.*, $c_i = c_j$, however, we can always perturb f such that $f(p_j) = c_j + \epsilon$, to get a perturbed function f' arbitrarily close to f with the strict ordering property. Passing a critical value in this ordering while sweeping t across a critical value from $c_i - \epsilon$ to $c_i + \epsilon$ causes a well defined change in the topology of $\mathbb{M}_t$. The change in topology of $\mathbb{M}_t$ on sweeping through c_i is modeled by attaching a handle.

Definition 3.2.2. **(λ-Handle)** A **λ-handle** $D^d_{p_i}$ corresponding to a critical point p_i with index λ is a d-manifold homeomorphic to a d-ball that is attached through disjoint union to $\mathbb{M}_{c_i-\epsilon}$, so that $\mathbb{M}_{c_i-\epsilon} \sqcup D^d_{p_i}$ is diffeomorphic to $\mathbb{M}_{t+\epsilon}$.

A manifold can be constructed (up to homeomorphism) by a sequence of gluing handles.

Definition 3.2.3. **(Handle Decomposition)** The **handle decomposition** of an n-manifold M is a representation of that manifold as a sequence $M_0 \subset M_1 \subset ... \subset M$, where each M_i is obtained from M_{i-1} by attaching a λ-handle.

The handle decomposition is a sequence of gluing operations, where handles are glued to the manifold changing its structure. Note that any order of attaching handles in a handle decomposition will generate the same manifold. Figure 3.2.1 illustrates one handle decomposition of a torus with the height function defined on it, where the change in topology of each successive surface is obtained

by attaching a handle. The handle decomposition of this example would be given by the sequence of attaching: a 0-handle, a 1-handle, a 1-handle, and a 2-handle.

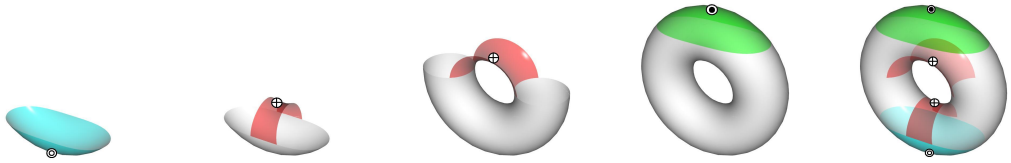


Figure 3.2.1: The handle decomposition of a torus with the height function defined. From left to right: attaching a 0-handle creates a disk; attaching 1-handle turns the disk into a tube; attaching a 1-handle creates a torus with a hole; attaching a 2-handle fills in the hole in the surface. Each successive surface is generated by gluing a 2-disk to the boundary of the previous surface.

When handles are attached in the ordering of critical points of increasing value, $p_0, p_1, ..., p_n$, the handle decomposition tells a story of the lifetime of features of the manifold. This will be an important concept when we look at topological features. In fact, the following theorem states that all interesting changes to the structure of the manifold occurs only during attaching a handle.

Theorem 3.2.4. *Let $f : \mathbb{M}^d \to \mathbb{R}$ be a Morse function, and p be an index λ critical point of f, and $c = f(p)$. Then $\mathbb{M}_{c+\epsilon}$ is diffeomorphic to $\mathbb{M}_{c-\epsilon} \cup D^\lambda \times D^{d-\lambda}$.*

For example, this theorem states that the grey regions in each step of the handle decomposition of figure 3.2.1 is diffeomorphic to the manifold in the previous step.

The dimension of $\mathbb{M}$ is the same as the dimension of the handles in its decomposition. For 3-manifolds, attaching a 0-handle creates a new 3-manifold component, attaching a 1-handle glues two components together or creates a "loop", attaching a 2-handle creates a bubble or fills in a loop, and attaching a 3-handle fills in a bubble. Figure 3.2.2 illustrates the 1- and 2-handles for 3-manifolds.

3.3 Reeb Graphs and Contour Trees

Reeb graphs and contour trees trace out the creation, destruction, merging, and splitting of level set components. They have been studied extensively, and several algorithms exist for computing Reeb Graphs and contour trees.

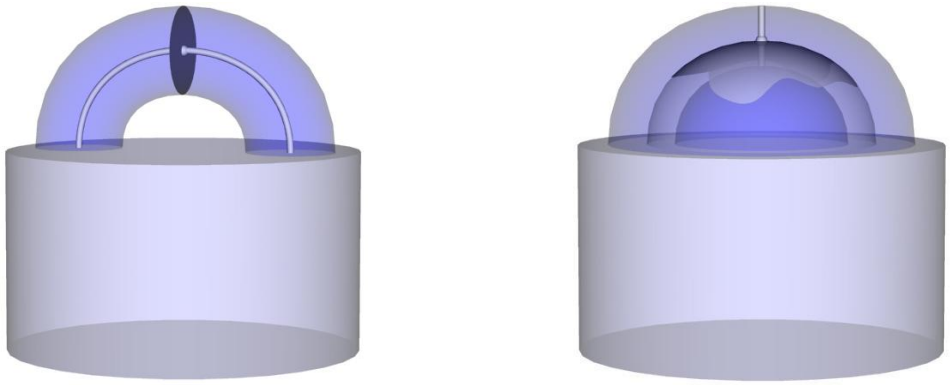


Figure 3.2.2: When constructing 3-manifolds, attaching a handle corresponding to a 1-saddle (left) creates a loop. Attaching a handle corresponding to a 2-saddle (right) fills in a loop.

Definition 3.3.1. (Reeb Graph) Let $f : \mathbb{M} \to \mathbb{R}$ be a scalar function defined on a compact manifold $\mathbb{M}$. The **Reeb graph** of f is the quotient space of the graph of f in $\mathbb{M} \times \mathbb{R}$ by the equivalence relation "$\sim$" given below:

$$(X_i, f(X_1)) \sim (X_2, f(X_2))$$

if and only if

$$f(X_1) = f(X_2)$$

and X_1, X_2 are in the same connected component of $f^{-1}(f(X_1))$.

This describes an equivalence relation between isocontours, where two isocontours at values c_i and c_j of $f : \mathbb{M} \to \mathbb{R}$ are related if they are in the same connected component of $M \subseteq \mathbb{M}$, where $M = \{x | c_i \leq f(x) \leq c_j\}$ and there is no critical point in M. The finite equivalence classes give the nodes of the Reeb graph, and the infinite classes give the arcs of the graph. The following is a more intuitive, yet less precise, definition: the Reeb graph is the result of contracting each isocontour to a point. The *contour tree* is a Reeb graph defined over a simply connected Euclidean space $\mathbb{E}^d$. Figure 3.3.1 shows the Reeb graph of a height function on a torus, and how the Reeb graph implies a segmentation of the domain.

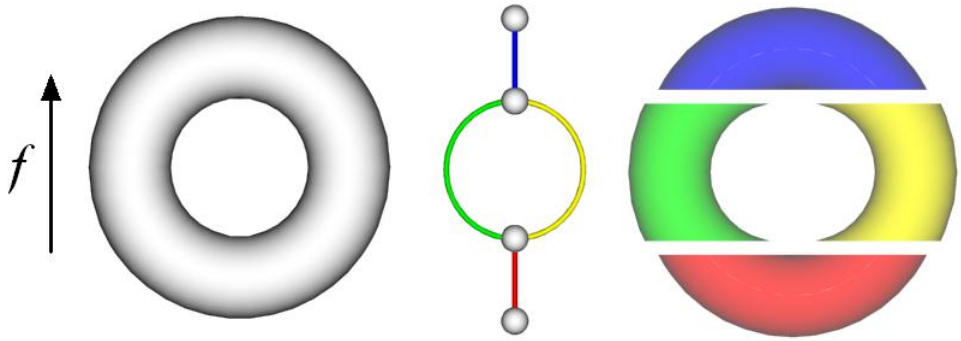


Figure 3.3.1: Let f be the height function defined on a torus (left). The Reeb graph captures the topology of the torus (middle), and the arcs of the Reeb graph subdivide the torus into sections where level sets have a single component (right). Note that the critical points of f are represented as nodes in the Reeb graph.

3.3.1 Previous Work in Reeb Graphs and Contour Trees

Contour trees were introduced by Boyell and Ruston [4], and were used by Freeman and Morse [20] to find terrain profiles in a topographic map. Several algorithms have been proposed to compute contour trees. The most successful algorithms for constructing a contour tree involve sweeps through the data, identifying where level sets are created, merge, split, and are destroyed. Tarasov and Vyalyi [56], basing their algorithm off Kreveld et al. [57] and Mark [33, 34], introduced using three sweeps, one to find the creation and joining of level sets, encoded in a *join tree*, one to find the splitting and destruction of components, encoded in a *split tree*, and one to join the two trees to find the complete contour tree. This $O(m \log m)$ construction, m being the number of simplices in the data, was an improvement over the $O(m^2)$ approach by Kreveld [57]. This was improved by Carr at al. [6] to $O(n \log n + m)$, and later by Pascucci and Cole-McLaughlin [44] to $O(n + t \log n)$ through a divide and conquer strategy, where n and t are the number of vertices and critical points respectively. These methods to construct a contour tree rely on the Union-Find data structure [10] to mark and query efficiently the level set component of a vertex.

G. Reeb [49] first introduced the Reeb graph, however, the formal modern description of a Reeb graph that we use is the one given by Shinagawa and Kunii [51]. Construction of Reeb graphs is more complex, and an explicit representation of the level sets is required as they sweep through the domain. Cole-McLaughlin et al. [8] and Shinagawa and Kunii [51] presented practical

algorithms for constructing Reeb graphs, however, their methods were limited to smaller datasets, global sorting, and the demand for the input to be a manifold surface. Of particular interest is the algorithm presented by Pascucci et al. [45] for constructing Reeb graphs, as it constructs the Reeb graph in a streaming manner, by maintaining and updating the Reeb graph for the geometry processed at a particular stage of the streaming. We will utilize a similar idea to design a divide-and-conquer algorithm for constructing Morse-Smale complexes (section DACA).

Reeb graphs and contour trees have been used as an aid in visualization and simplification. Kreveld et al. [57] used the contour tree to generate seed sets for isosurface traversal, eliminating the need to examine every cell of the data to extract an isosurface. Shinagawa and Kunii [52] devised an efficient surface coding based on Reeb graphs so that shapes encountered in medical imaging could be described accurately and efficiently. Reeb graphs, contour trees, and their variants have been used successfully to guide the removal of topological features [7, 21, 54, 55, 58].

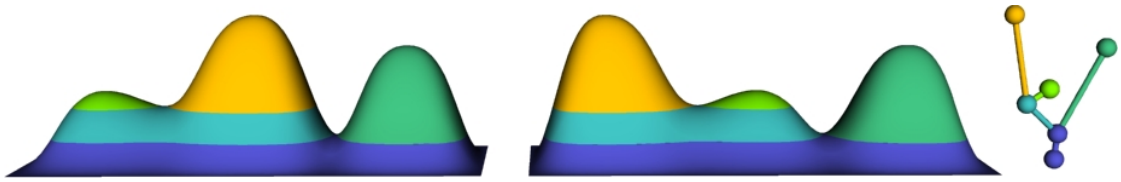


Figure 3.3.2: Let f be the height function defined on the two terrains (left and middle). The contour tree for both these terrains is the same (right).

The Reeb graph maps out the relationship between index-0 and index-1, and index-$(d-1)$ and index-d critical points in a d-dimensional space $\mathbb{M}^d$. One limitation of contour trees and Reeb graphs is that they do not encode the neighborhood information of contours. Figure 3.3.2 illustrates two terrains that are "structurally" different but have the same contour tree. The information that is lacking is the geometric connectivity of the features. In fact, this poses even more serious problems by allowing reconnection of a Reeb graph after performing a saddle-extremum cancellation in a manner that can not be realized geometrically. Additionally, for higher dimensional manifolds, the saddle-saddle connections are not represented in the Reeb graph. These saddle-saddle pairs describe the loops and holes in level set components. Pascucci et al. [44] addressed this problem in $\mathbb{R}^3$ with the **Augmented Contour Tree** by calculating the Betti numbers along each arc of the contour

tree. However, for Morse functions on manifolds of dimension ≥ 3, the connections between saddles are essential to enable a complete topology-based simplification of the function, limiting the applicability of Reeb graph based methods.

3.4 Surface Segmentation

There have been many proposed surface segmentation strategies to encode the structure of a function on a surface. Surface networks ideally segment terrain-type data into the cells of the two-dimensional Morse-Smale complex, *i.e.*, into regions of uniform gradient flow behavior. Such a segmentation of a surface would identify the features of a terrain such as peaks, saddles, dips, and the lines connecting them. In practice, such methods suffer from a lack of rigor. For example, Pfaltz [47, 46] extracted a surface network that he guaranteed was a valid network for some terrain, but could not prove that it was the surface network of the desired terrain. Rana and Morely [48] provide an excellent survey of the methods and limitations of surface networks.

Another strategy for segmenting surfaces often used in image processing is the watershed transform. Introduced by Digabel and Lantuéjoul [12], and later improved by Beucher and Lantuéjoul [3] the watershed transform segments an image(a scalar valued terrain) into regions of uniform local gradient behavior. The watershed is the set of ridge lines(as in [36]) that delimit the boundaries between basins in a terrain. Roedink and Meijster [50] provide a survey of the methods and limitations of the watershed transform. The watershed transform segments an image based on a number of heuristics. In general, the transform is computed on the morphological gradient of the image [38], however, this gradient is not strictly defined. Computing a distance field from the boundaries of some boundary detection on the image is a popular method for creating a function with a "nice" gradient [37, 42]. A pseudo-Morse complex is extracted, computing the watersheds and basins. This differs from a true Morse complex in that integral lines terminate at boundaries, and are not followed to their limits. The waterfall algorithm [2] merges cells by removing their shared boundary. The order of simplification is based on heuristics, such as closeness of texture or color within each cell. As with surface networks, the weakness of the watershed method for analysis is that there

is no consistent theory behind it, as different heuristics are developed for the particular needs of an application. Degeneracies are dealt with using *ad hoc* solutions, such that results cannot be guaranteed. Nevertheless, the driving motivation behind these techniques is so urgent that the watershed transform remains a popular method for image segmentation.

3.5 Morse Complex

Instead of partitioning a manifold according to the behavior of level sets, it is more general to partition the manifold based on the behavior of the gradient. Critical points (definition 3.1.2) are points where the gradient vanishes, and therefore will be represented in any technique involving analysis of the gradient. In fact, the behavior of level sets can be induced from the behavior of the gradient in such a manner that it is sufficient to explore the behavior of the gradient to understand the topology of level sets of a function on a manifold. The gradient of a function defines a smooth vector field on $\mathbb{M}$ with zeroes at critical points.

Definition 3.5.1. (Integral Line) A curve $l(t)$ is an **integral line** of f if $\frac{\partial}{\partial t}l(t) = \nabla f(l(t))$ for all $t \in \mathbb{R}$. In other words, an integral line is a path where the tangent of the path is parallel to the gradient at every point along the path.

Integral lines represent the flow along the gradient between critical points. At any point where the gradient is not zero, the integral line passing through that point can be found by tracing forwards and backwards along the gradient vector field. An integral line is an open 1-manifold, and since $\mathbb{M}$ is compact, the limits at both end exist. The following definition identifies the upper and lower limit points of integral lines.

Definition 3.5.2. (Origin, Destination) The limit $lim_{t\to-\infty}l(t)$ is called the **origin** of an integral line $l(t)$, and is denoted $org(l)$. The limit $lim_{t\to\infty}l(t)$ called the **destination** of $l(t)$, and is denoted $dest(l)$.

The integral line passing through a critical point is the critical point itself. The next theorem describes some properties of integral lines in smooth functions.

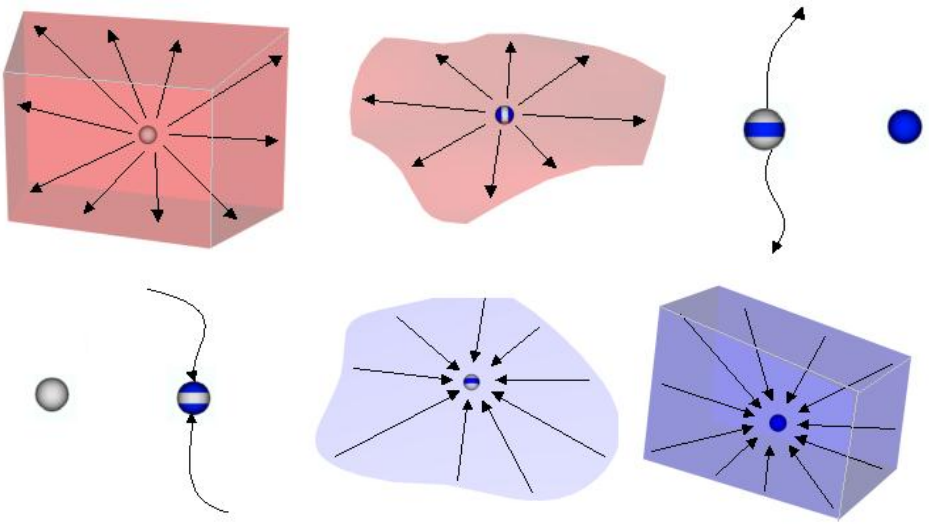


Figure 3.5.1: The dimension of ascending (red) and descending (blue) manifolds of $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ depends on the index of the corresponding critical points. The ascending manifolds of a minimum, 1-saddle, 2-saddle, and maximum are a volume, a surface, pair of lines, and a point respectively. Similarly, the descending manifolds are a point, a pair of lines, a surface, and a volume.

Theorem 3.5.3. *Integral lines have the following properties:*

1. *Two integral lines are either disjoint or the same.*
2. *Integral lines cover all of* $\mathbb{M}$.
3. *The origin and destination of an integral line are critical points of* f.

Integral lines are monotonic and therefore $org(l) \neq dest(l)$. These properties ensure that every point in the domain has exactly one integral line passing through it. All points on $\mathbb{M}$ can be classified based on the origin or destination of the integral line passing through that point.

Definition 3.5.4. (Ascending/Descending Manifolds) Let p be a critical point of $f : \mathbb{M} \rightarrow \mathbb{R}$. The **ascending manifold** of p is the set of points belonging to integral lines whose origin is p, $A(p) = \{p\} \cup \{x \in \mathbb{M} |\ x \in \text{im}(l),\ org(l) = p\}$. The **descending manifold** of p is the set of points belonging to integral lines whose destination is p, $D(p) = \{p\} \cup \{x \in \mathbb{M} |\ x \in \text{im}(l),\ dest(l) = p\}$.

Note that ascending and descending manifolds are also referred to as unstable and stable manifolds [40], lower and upper disks [35], and right-hand and left-hand disks [39]. Figure 3.5.1 illus-

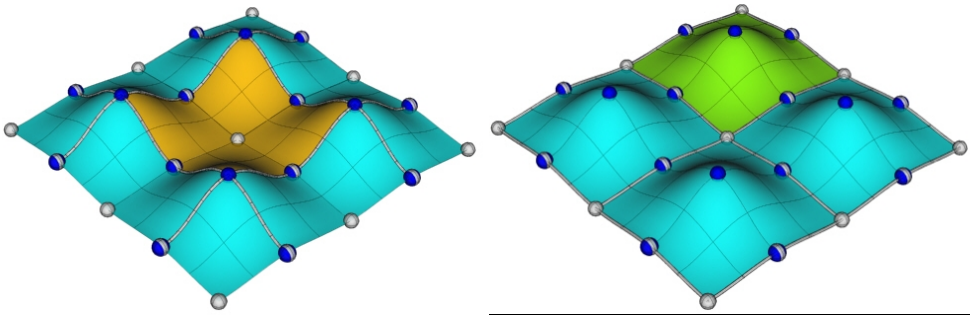


Figure 3.5.2: A terrain is decomposed into is ascending manifolds (left). This decomposition is the Morse complex of $-f$. The terrain can also be decomposed into its descending manifolds (right), the Morse complex of f. The orange region illustrates a 2-cells of the Morse complex of $-f$, and every point in this region is part of an integral line whose origin is the minimum in the center of the region. The green region symmetrically shows a 2-cell of the Morse complex of f, where every point in this region is part of an integral line whose destination is the maximum in the center of the region.

trates the ascending and descending manifolds of critical points of a function on $\mathbb{R}^3$. Viewing $\mathbb{M}$ as a terrain, the ascending manifolds correspond to "basins," and descending manifolds correspond to "mountains". Note that negating the function turns the ascending manifolds into descending manifolds, and vice versa. The following theorem formally states the relationship between a critical point and its ascending and descending manifolds.

Definition 3.5.5. (Open d-Cell) An **open d-cell** is a space homeomorphic to $\mathbb{R}^d$.

Theorem 3.5.6. *Let $f : \mathbb{M}^d \to \mathbb{R}$ be a Morse function, and p be an index λ critical point of f. The ascending manifold $A(p)$ of p is an open cell of dimension $d - \lambda$. The descending manifold $D(p)$ of p is an open cell of dimension λ.*

Definition 3.5.7. (Morse Complex) Let $f : \mathbb{M}^d \to \mathbb{R}$ be a Morse function. The complex of descending manifolds of f is called the **Morse complex**.

The descending manifolds decompose $\mathbb{M}^d$ into open cells. The cells form a complex, since the boundary of every cell is the union of its faces. Figure 3.5.2 illustrates the ascending and descending manifolds of a terrain: the height function defined on a 2-manifold. The cells of the Morse complex of this terrain are 2-manifold regions associated with minima, 1-manifolds associated with saddles, and 0-manifolds associated with maxima.

3.6 Morse-Smale Complex

The Morse complex classified each point according to the origin or destination of the associated integral line. It is possible to classify points according to both origin and destination, but to do this in a manner that consistently generates a complex, additional definitions are required.

Definition 3.6.1. (Transversal Intersection) Two submanifolds $\mathbb{M}^a$ and $\mathbb{M}^b$ of $\mathbb{M}^d$ are said to **intersect transversally** if the tangent space of $\mathbb{M}^a$ and the tangent space of $\mathbb{M}^b$ generate the tangent space of $\mathbb{M}^d$, or when the intersection $\mathbb{M}^a$ and $\mathbb{M}^b$ is empty. When $a + b = d$, the intersection is a single point.

Less formally, we can think of an intersection of two manifolds as transversal when they are not "parallel" at their intersection. When an ascending and descending manifold intersect transversally at exactly one point, that point must be critical, and the manifolds are the ascending and descending manifolds associated with that critical point.

Definition 3.6.2. (Morse-Smale Function) A Morse function is **Morse-Smale** if the ascending and descending manifolds intersect only transversally.

A direct result of this condition is that a pair of critical points that are the origin and destination of an integral line in a Morse-Smale function f cannot have the same index, and furthermore, the index of the critical point at the origin is less than the index of the critical point at the destination. In $\mathbb{R}^2$, this prohibits an integral line originating at a saddle from terminating at another saddle.

Definition 3.6.3. (Morse-Smale Complex) Given a Morse-Smale function f, the **Morse-Smale complex** of f is the complex formed by the intersection of the Morse complex of f and the Morse complex of $-f$.

This defines an equivalence relation over the points in the domain, where two points are related exactly when their associated integral lines share a common origin and destination, and there is a connected path of points between them where the associated integral line of every point on the path shares the same origin and destination. The equivalence classes form the cells of the Morse-Smale complex. Intuitively, cells of the Morse-Smale complex are formed by the collection of

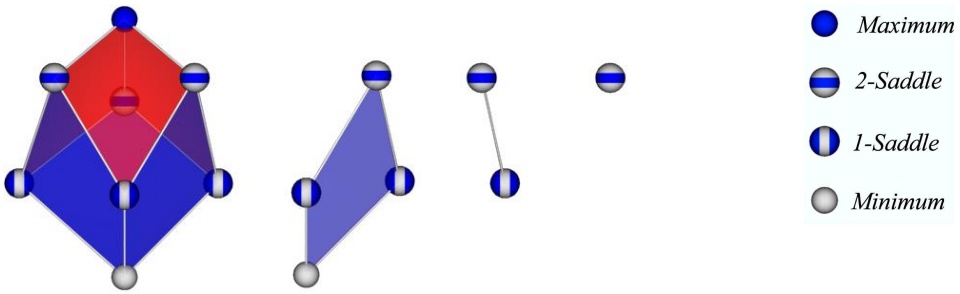


Figure 3.6.1: The cells of a complex for a three dimensional domain. From right to left, a **crystal** is a 3-cell that has one minimum, one maximum, and an alternating ring of 1-saddles and 2-saddles. The facets of a crystal are **quads**, quadrangular 2-cells. The facets of quads are the **arcs**, integral lines connecting critical points with indices differing by one. The facets of arcs are **nodes**, which are critical points of the function.

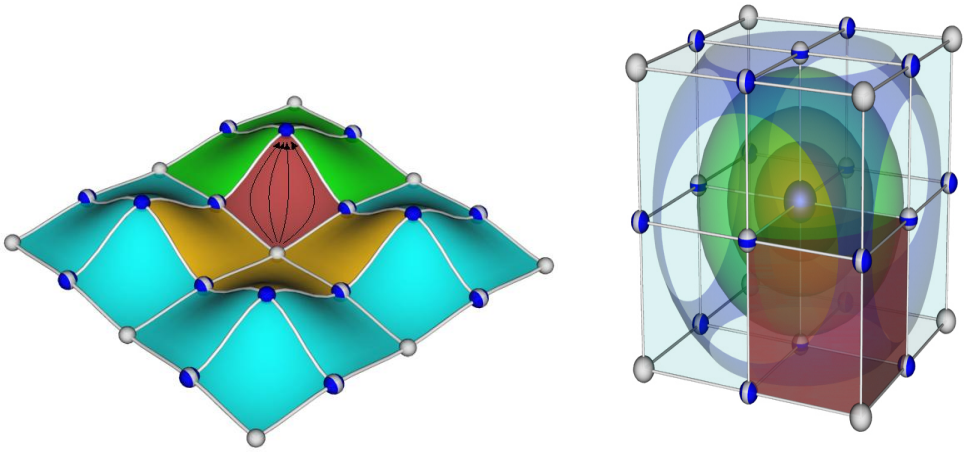


Figure 3.6.2: A two-dimensional Morse-Smale function (left) is partitioned by the cells of the Morse-Smale complex. Note that these cells are the result of intersecting the ascending and descending manifolds from Figure 3.5.2. A 3-manifold with a function that has a maximum in the middle is partitioned into cells of the three-dimensional Morse-Smale complex (right).

integral lines that share a common origin and destination. When f is defined on $\mathbb{M}^3$, we denote the cells of dimension 0, 1, 2, and 3 as **nodes**, **arcs**, **quads**, and **crystals**. Furthermore, these cells have consistent combinatorial structures. Figure 3.6.1 shows the cells of different dimension. Figure 3.6.2 shows and example of the complex in two dimensions and three dimensions.

3.7 Piecewise Linear Morse-Smale Complex

Scientific data is usually available as a finite number of discrete samples over some continuous space. A continuous function is constructed by assigning some triangulation K with vertices at the sample points, and interpolating f in the interior of cells. In general, the derived function is not smooth, *i.e.*, it is not necessarily first or second-order differentiable, and therefore the traditional notions from smooth Morse theory are undefined. Edelsbrunner et al. [15] adapted smooth Morse theory to be applicable to piecewise linear(PL) functions defined over $\mathbb{M}^2$ and $\mathbb{M}^3$ [14] by introducing the *quasi Morse complex*, a complex structurally identical to a Morse complex.

Let K be a triangulation (simplicial complex) of a compact 2- or 3-manifold $\mathbb{M}$ and let $f : \mathbb{M} \rightarrow \mathbb{R}$ be linear on all the cells of K. The function is defined by the scalar values at its vertices. Using *simulation of simplicity* [16], a symbolic perturbation of f, we can assume that $f(u) \neq f(v)$ for all $u \neq v$ in K. A critical point p_i of f with critical value c_i occurs where there is a change in the topology of the level set at $c_i - \epsilon$ and $c_i + \epsilon$. In Morse functions, this happens when the gradient vanishes. We can say that a vertex v of K is critical if the topology of the level sets below and above $f(v)$ are different. To detect this, we analyze the PL equivalent of a neighborhood of v, the *star* of v, $St\ v$.

Definition 3.7.1. (Star($St\ v$)) The **star** of a vertex $v \in K$ is $St\ v = \{\sigma \in K | v \leq \sigma\}$, where $v \leq \sigma$ means v is a face of σ.

Definition 3.7.2. (Lower Star($St^-\ v$)) The **lower star** of a vertex $v \in K$ is $St^-\ v = \{\sigma \in St\ v | f(u) \leq f(v), u \leq \sigma\}$.

Definition 3.7.3. (Upper Star($St^+\ v$)) The **upper star** of a vertex $v \in K$ is $St^+\ v = \{\sigma \in St\ v | f(u) \leq f(v), u \leq \sigma\}$.

Colloquially, the star of a vertex v is the set of all co-faces of v . The *link* of a vertex v is composed of the closure of the star.

Definition 3.7.4. (Link ($Lk\ v$)) The **link** of a vertex $v \in K$ is $Lk\ v = \{\tau \leq \sigma \in St\ v | v \not\leq \tau\}$, where $v \not\leq \tau$ means v is not a face of τ.

Definition 3.7.5. (Lower Link ($Lk^-\ v$)) The **lower link** of a vertex $v \in K$ is $Lk^-\ v = \{\sigma \in Lk\ v | f(u) < f(v), u \leq \sigma\}$.

Definition 3.7.6. (Upper Link ($Lk^+\ v$)) The **upper link** of a vertex $v \in K$ is $Lk^+\ v = \{\sigma \in Lk\ v | f(v) < f(u), u \leq \sigma\}$.

The connected components of the lower link determine whether or not a vertex is critical, and its index of criticality. More specifically, the reduced Betti number of the lower link distinguishes a regular point from a critical point, and also indicates the index of the critical point [41]. Intuitively, if the number of components of the lower link are not equal to one, or the number of components of the upper link are not equal to one, level sets passing through the vertex will change topology.

A piecewise-linear function has piecewise constant gradient, therefore integral lines may merge. Instead of computing a Morse-Smale function, we simply restrict the structure of the cells of the complex identified, so that it is structurally identical to the Morse-Smale complex for *some* Morse-Smale function. The *quasi Morse-Smale complex* is such a complex. For three-dimensional domains, it is a decomposition of space into crystals, quads, arcs, and nodes. Let N_0, N_1, N_2, and N_3 be the sets of minima, 1-saddles, 2-saddles, and maxima of f. Let A_{01}, A_{12}, and A_{23} be the sets of arcs that connect minima to 1-saddles, 1-saddles to 2-saddles, and 2-saddles to maxima respectively. Let Q_{012} and Q_{123} be the sets of quads with nodes from N_0, N_1, N_2, N_1 and N_1, N_2, N_3, N_2 in that order, respectively, around the boundary.

Definition 3.7.7. (Quasi Morse-Smale Complex) A **quasi Morse-Smale complex** is a segmentation of K into open cells that satisfies the following properties:

1. all nodes are from N_0, N_1, N_2, and N_3, all arcs are from A_{01}, A_{12}, and A_{23}, and all quads are from Q_{012} and Q_{123},
2. there are no critical points within the arcs, quads, or crystals, and
3. each arc in A_{12} is the boundary of four quads, which in a cyclic order alternate between Q_{012} and Q_{123}.

The quasi Morse-Smale complex is structurally indistinguishable from the Morse-Smale complex of some Morse-Smale function $f' : \mathbb{M} \to \mathbb{R}$. The gradient of a PL function is not continuous across simplices, therefore, any combinatorially correct complex can be viewed as the "correct" Morse-Smale complex given some consistent method for resolving the degeneracies in gradient flow. In this way, the complex of cells of the quasi Morse-Smale complex define a smooth Morse-Smale function over $\mathbb{M}$. In the following, when referring to a Morse-Smale complex of some PL function or discretely sampled function, we implicitly mean a "quasi Morse-Smale complex".

3.8 Topological Features and Simplification

The presence of critical points in a function implies some notion of complexity. Each critical point corresponds to the attachment of some handle to the manifold, which corresponds to changes in the topology of a manifold, and also affects the structure of level sets above and below the critical point. The handle decomposition correlates the function directly with the embedding, and can be used to analyze a specific Morse function. One can imagine that attaching a handle to a manifold adds some topological detail to it.

In a similar manner, critical points as represented in a Morse-Smale complex correspond to places where level set behavior of the function is changed. For example, in 2-manifolds, when sweeping levels sets from low function value to high function value, a minimum will create a new level set component, a saddle will merge or split level set components, and a maximum will destroy level set components. When a saddle merges two level set components, it *destroys* one of the components (by merging it with the other) that was created by some minimum. In this sense, we can pair the critical points of a function in a manner where the lower critical point corresponds to a change in the topology of the level sets that is undone by the upper critical point. This pairing is known as **persistence**.

Definition 3.8.1. (Persistence) Let p_a be the critical point creating a boundary component B and p_b the critical point destroying B, then the pair (p_a, p_b) is a **persistence pair**. The difference is function value $|f(p_a) - f(p_b)|$ is called the **persistence** of the topological feature (p_b, p_a).

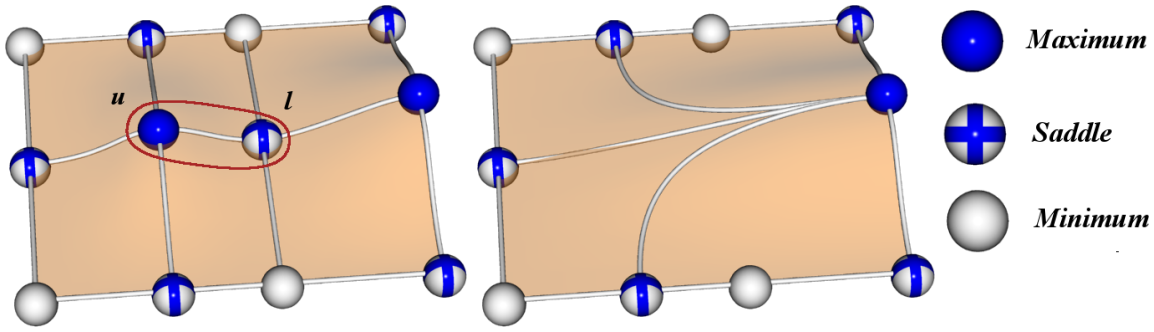
Intuitively, persistence describes how much a function would need to change to remove a feature. Edelsbrunner and Mücke [15] described how a *filtration* is a global persistence pairing of critical points that give a successive refinement of topological detail.

3.8.1 Simplification

Critical points can be paired in a manner where each pair corresponds to the life-cycle of a topological feature, and removing that pair of critical points simplifies a function by removing the topological feature.

Theorem 3.8.2. ***(Cancelling Handles)*** *Let p_i and p_j be critical points of index λ and $\lambda + 1$ respectively of $f : \mathbb{M}^d \to \mathbb{R}$, with critical values c_i and c_j. If the ascending manifold of p_i and the descending manifold of p_j intersect transversally at a unique point for each level set for function values in the range $[c_i, c_j]$, then f can be perturbed in such a way that p_i and p_j are removed with only local change to the function.*

Figure 3.8.1 illustrates a *cancellation*, where a pair of critical points corresponding to the creation (saddle) and destruction (maximum) of a level set component are removed from the function, in effect "smoothing" it out.



(a) (b)

Figure 3.8.1: The circled arc connects a saddle l to a maximum u (a). Cancellation of (l,u) removes a topological feature and smooths out the function.

Successive cancellations simplify a function. When cancellations are performed in increasing order of persistence, a sequence of systematically simpler functions are generated, enabling a multi-

resolution representation of a function, where successive resolutions correspond to a simplification of the function. This is a powerful concept that allows us to use simplification to identify topologically relevant features and remove noise. Simplification and filtering will be discussed extensively in section 5.1.

Chapter 4

Computing the Morse-Smale Complex through Simplification

4.1 Previous Work

The predecessors of my own work are the algorithms for computing the Morse-Smale complex in two dimensions, and the proposed algorithms for computing the Morse-Smale complex in three dimensions. Edelsbrunner et al. [15, 14] developed the theoretical extensions of Morse theory to PL functions that is essential for a robust solution. Bremer et al. [5] provided an efficient algorithm for computing the Morse-Smale complex for two-dimensional domains. These will be discussed in more detail in the following section.

4.1.1 Morse-Smale Complex in Two Dimensions

The initial paper for computing the Morse-Smale complex for triangulated 2-manifolds was presented by Edelsbrunner et al. [15]. This paper described the extension of smooth Morse theory to PL functions, as described in section 3.7. The algorithm presented in this paper traces lines of steepest ascent and descent from saddles in an ascending and descending sweep. The lines are restricted to edges of the triangulation, therefore much of the discussion of the paper describes how to maintain a system to keep track of the splits and merges of integral lines. In the theory of smooth Morse functions, gradient lines are disjoint, however, in the theory of PL functions they can merge, adding an additional level of complexity. For example, when an ascending and descending line occupy the same edge, there are restrictions on the direction each can take when they are split again, to ensure that they do not cross, thus maintaining the transversal property of manifolds in a Morse-Smale function. The ability to order parallel edges on a 2-manifold makes this kind of bookkeeping possible at all.

The output of the arc tracing in the first stage of the algorithm is a quasi Morse-Smale complex: it is combinatorially valid, but may not represent the connections between critical points that a true gradient following method would produce. To solve this problem, they introduce the *handle slide* operation on the quasi Morse-Smale complex that redirects the interior arcs of three adjacent cells. This operation has only a local impact, and "moves" the arcs of the complex closer to the actual gradient lines. After resolving multi-saddles by splitting saddle points, the Morse-Smale complex

is computed.

This paper also presents a hierarchy for the complex by means of cancellation of persistence pairs. The notion of topological persistence [16] is used to order the critical point pairs for cancellation. Persistence deals with the life and death of a feature associated with critical points. Critical points that create features are paired with the critical points that destroy those features. The persistence of such a pair is the absolute difference in function value. The *adjacency lemma* [15] states that the i-th pair of critical points ordered by persistence forms an arc in the Morse-Smale complex obtained by canceling the first $i - 1$ pairs ordered by absolute difference of function value of its endpoints. This allows us to find the persistence pairing on-the-fly from the Morse-Smale complex without having to compute the pairs explicitly, by always canceling the pair of critical points connected by the arc of the complex that have the lowest absolute difference in function value. By canceling critical points in such a way, it is possible to create a hierarchy of complexes that successively remove small features such as noise.

Bremer et al. [5] presented another algorithm for computing the Morse-Smale complex on triangulated 2-manifold surfaces which resolved much of the complexity of the initial algorithm. The main difficulties in Edelsbrunner's algorithm arose from the fact that ascending and descending integral lines were allowed to merge, and the fact that an integral line was restricted to edges of the triangulation. Bremer et al. traced integral lines along the true gradient, splitting triangles to insert edges where necessary. Furthermore, while ascending arcs were allowed to merge with other ascending arcs, and descending arcs were allowed to merge with descending arcs, a physical separation was always maintained between the descending and ascending arcs. The resulting algorithm is much simpler, and leads to a more efficient implementation.

The algorithm developed by Bremer et al. [5] for computing the two-dimensional Morse-Smale complex was used in quadrangular remeshing of a 2-manifold surface, by Dong et al. [13], and then in tracking the formation of bubbles in the turbulent mixing of fluids, by Laney et al. [29]. In both cases, the extraction of the two-dimensional Morse-Smale complex was the kernel of the methods.

4.1.2 Morse-Smale Complex in Three Dimensions

Edelsbrunner et al. [14] introduced the Morse-Smale complex as a combinatorial analog of the CW-complexes of the ascending and ascending manifolds of Morse functions. They proposed an algorithm for computing the Morse-Smale complex for volumetric datasets using this idea. The algorithm first computes the descending manifolds in a pass over the data, and then uses the cells of the Morse complex to guide the construction of ascending manifolds. The algorithm is proven to be correct for simplicial meshes, however, a major drawback is the immense complexity. In particular, maintaining disk invariants, resolving multi-saddles, and ensuring transversal intersection of descending and ascending manifolds requires such complicated bookkeeping that no implementation of this algorithm has been achieved.

In fact, the approach taken in this paper has some severe drawbacks. First, it is defined on a simplicial complex, *i.e.*, it is defined for tetrahedralized meshes, and requires storage at all simplices of the mesh. While it is possible to tetrahedralize gridded datasets, for any tetrahedralization except an implicit one the overhead of storing all the simplices is usually prohibitive. Furthermore, explicit storage of ascending and descending disks adds another huge overhead. In a typical dataset, the initial complex computed has a very large number of low persistence critical points. Storage of an ascending and descending disk at every 1-saddle and 2-saddle is unpractical. The most critical drawback is the high level of complexity, because of which there has been no implementation of this algorithm.

4.2 Algorithm Overview

The primary focus of my research has been extending Bremer et al. [5] to volumetric domains. Several factor have made this extension non-trivial: the combinatorial structure of the Morse-Smale complex is much harder to guarantee; the cancellation of 1-saddles and 2-saddles has no lower-dimensional equivalent; data structures cannot rely on the ability to order parallel lines; and there is larger overhead for mesh manipulation, such as splitting a vertex to resolve a multi-saddle. Given a trivariate scalar function with discrete samples over a grid, there are several options on selecting

an interpolant, and different interpolants often introduce complications that cannot be resolved. For example, using a trilinear interpolant a hexahedral grid of sample locations permits saddle locations in the interior and boundary of voxels. Even detecting such critical points can be numerically unstable, and connecting them as arcs of the complex is almost impossible. In fact, most of our methods have arisen out of the need to simplify the problem in general.

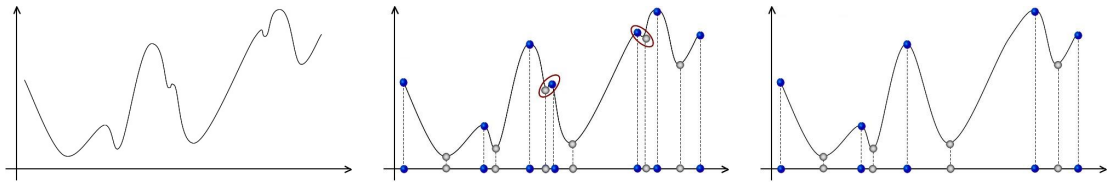
We describe a new method for constructing the Morse-Smale complex and explicit removal of topological features of a given trivariate scalar function f with the goal of constructing a hierarchical representation. Specifically, we

- introduce two atomic operations that destroy target topological features without affecting the function globally, *i.e.*, a characterization of the cancellation of critical points for 3-manifolds,
- introduce the *augmented function*, a function "close" to f for which the Morse-Smale complex is known *a priori*,
- describe a combinatorial algorithm that selectively removes non-significant topological features by repeated application of the two atomic operations,
- present an improved version of this algorithm,
- describe a data structure that stores the Morse-Smale complex and allows optimal execution of the atomic operations,
- and we show an example of how the Morse-Smale complex can be used to aid in data analysis and visualization.

Figure 4.2.1 shows how a controlled removal of topological features can help in multi-scale analysis of a function.

4.3 Cancellations and Multi-scale Analysis

A minimal Morse function is generated from f by repeated cancellation of pairs of critical points. This operation is legal (*i.e.*, it can be realized by a local perturbation of the gradient vector field)



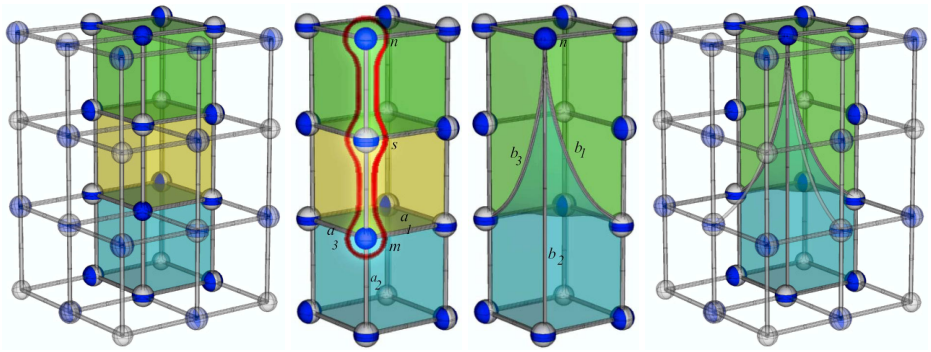
(a) (b) (c)

Figure 4.2.1: Multi-scale analysis of a univariate function. (a) Visualization of the function. (b) Critical points of the function partition the domain into monotonic regions. Pairs of critical points identify features, whose sizes are equal to the difference in function value of the critical points. (c) Small-sized monotonic regions are explicitly identified and removed, leaving behind the "significant" features.

if the indices of the two critical points differ by one and they are connected by a unique common arc in the Morse-Smale complex. For Morse functions defined on three-dimensional domains, we have three types of legal cancellations: minimum and 1-saddle; 1-saddle and 2-saddle; and 2-saddle and maximum. Cancellations play a crucial role in Morse theory for proving important results, including the generalized Poincaré conjecture for higher dimensions [53]. We use cancellations to reduce the number of critical points and hence remove topological features. The local change in the Morse-Smale complex indicates a smoothing of the gradient vector field and therefore a smoothing of the function f. The ordering of critical point pairs is specified by their persistence value, which quantifies the importance of the topological feature associated with a pair. The persistence of a critical point pair is defined as the absolute difference in value of f between the two points.

We simplify the Morse-Smale complex of a Morse function f by performing a series of critical point pair cancellations. A cancellation "simulates" a smoothing operation applied to f by modifying gradient flow in the neighborhood of two critical points. Arcs connecting critical points are lines of steepest descent or ascent, and changing them affects the gradient flow behavior of f. Rules that apply to gradient flow must be adhered to in the simplification process. For example, integral lines must remain disjoint.

Critical point pairs that we consider are end points of an arc in the Morse-Smale complex and therefore have consecutive indices. We group the pairs into two types: saddle-extremum (indices 1 and 0 or indices 2 and 3) and saddle-saddle (indices 1 and 2). The two types of cancellations are distinct in the way they modify gradient flow behavior. The cancellation procedure is similar



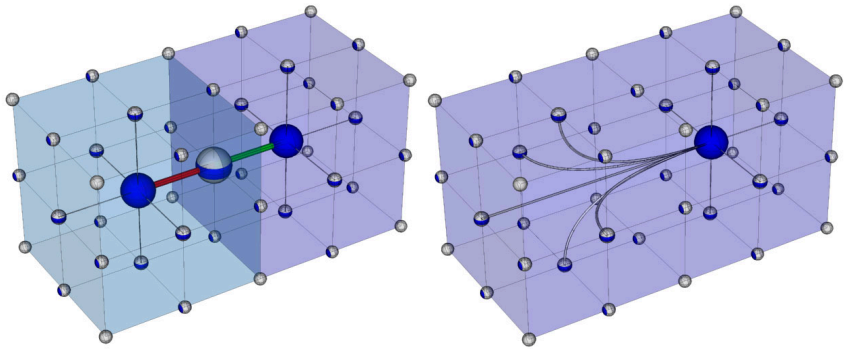
(a) (b) (c) (d)

Figure 4.3.1: Snapshot of a Morse-Smale complex before and after a saddle-maximum cancellation. (a) Three of the twelve crystals affected by the cancellation are colored. (b) Close-up view of the three crystals and the maximum-saddle-maximum triple that is merged into a single maximum. (c) After cancellation, all ascending arcs and disks that originally flow into the maximum lying below the saddle, flow into the maximum lying above the saddle. One of the three crystals is deleted in this process. (d) Ascending arcs and disks in the other nine crystals are similarly re-routed, resulting in a removal of three more crystals.

to vertex removal used in mesh simplification methods, with a pair of critical points being removed instead of a single vertex, and the reconnection of the complex governed by rules of Morse theory rather than mesh geometry. For reasons of clarity, we illustrate the two types of cancellations using prototypical figures of Morse-Smale complexes. The description, however, holds for all possible configurations. Recall that each critical point is the origin of an ascending manifold, and the destination of a descending manifold. The most intuitive way to understand the cancellations is to identify the changes to the ascending and descending manifolds involved.

4.3.1 Saddle-Extremum Cancellation

The saddle-extremum cancellation removes either a 2-saddle-maximum pair or a 1-saddle-minimum pair. The two pairs are dual to each other as can be seen by negating the function: maxima become minima, 2-saddles become 1-saddles and vice-versa. We restrict our discussion to 2-saddles and maxima. A 2-saddle, by definition, is connected by ascending arcs to exactly two maxima. When one of these maxima is removed in a saddle-maximum cancellation, integral lines ending at this maximum flow toward the second maximum. One can think of a saddle-maximum cancellation



(a) (b)

Figure 4.3.2: The configuration before the cancellation of the red arcs(a). The light blue and darker blue regions are the descending 3-manifolds of the two maxima. After the cancellation (b), these regions are merged, as are the ascending 1-manifolds terminating at the cancelled maximum.

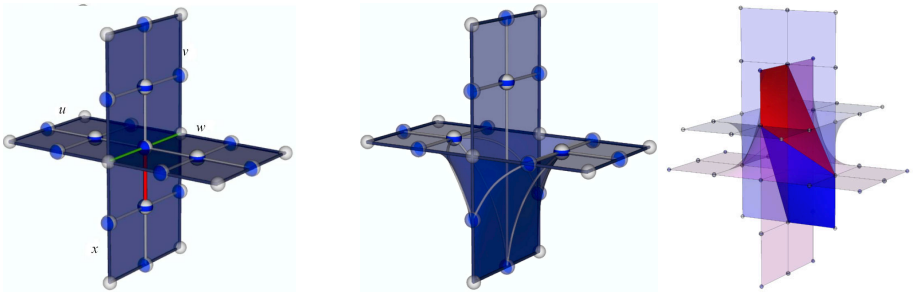
as merging three critical points into one maximum. Applying the saddle-maximum cancellation simplifies the function by removing a "bump." Figure 4.3.1 shows how the integral lines terminating at the two maxima flow into the remaining maximum after cancellation.

In terms of ascending and descending manifolds, the 2-saddle-maximum cancellation merges the descending 3-manifolds of the two maxima along their co-boundary, the descending disk of the 2-saddle. The ascending 1-manifolds that terminated at the cancelled maximum are merged with the ascending 1-manifold of the 2-saddle - the arcs connecting the cancelled maximum, the 2-saddle, and the other maximum. Figure 4.3.2 shows the saddle-extremum cancellation in terms of the ascending and descending manifolds.

A saddle-maximum cancellation is legal only if the 2-saddle is connected to two distinct maxima. If this condition is not met, then we recognize that the cancellation causes a *strangulation* of the descending disk that originates at the 2-saddle. In fact, it is not possible to route the integral lines terminating at the 2-saddle if we cancel such a saddle-maximum pair. Figure 4.3.5 shows this configuration.

4.3.2 Saddle-Saddle Cancellation

The saddle-saddle cancellation removes a 1-saddle-2-saddle pair, see Figures 4.3.3 and 4.3.4. This cancellation does not have an analog in lower dimensions. A 1-saddle descends to exactly two



(a) (b) (c)

Figure 4.3.3: A saddle-saddle cancellation. (a) Descending disks affected by the cancellation. The red arc connects the pair to be cancelled. All four disks (u, v, w, and x) have two common descending arcs (shown in green) on their boundary, both originating from the 1-saddle to be removed. (b) Descending disks that remain after cancellation. The green descending arcs are deleted from the boundary of the three surviving disks, which now extend to inherit the boundary of x. Ascending manifolds are dual to the descending manifolds and hence modified in a similar way. The new intersections of the modified ascending and descending disks (c) gives rise to additional cells in the Morse-Smale complex.

minima, and a 2-saddle ascends to exactly two maxima. After canceling this saddle pair, we need to ensure that the two pairs of extrema originally separated by these saddles remain that way. This constraint necessitates the introduction of new cells to fill in space between the two pairs of extrema. One can think of this cancellation by considering what happens to the descending disk originating from the 2-saddle and the ascending disk originating from the 1-saddle. After cancellation, these two disks disappear, and neighboring disks stretch out and share their boundary. We can no longer consider the cancellation as merging three critical points as we can for the saddle-maximum cancellation. Consider the descending disk that is removed by the cancellation: The boundary of this disk consists of alternating 1-saddles and minima. Arcs lying within the disk connect the 1-saddles on the boundary to the 2-saddle where the disk originates. One of these 1-saddles is involved in the cancellation. This 1-saddle and its two descending arcs are deleted as a result of the cancellation. Descending disks that contain these descending arcs in their boundary expand to share the boundary of the removed disk. Similarly, one ascending disk is removed and its boundary is shared by the neighboring ascending disks. Figure 4.3.3 illustrates the operation by showing the descending disks before and after cancellation.

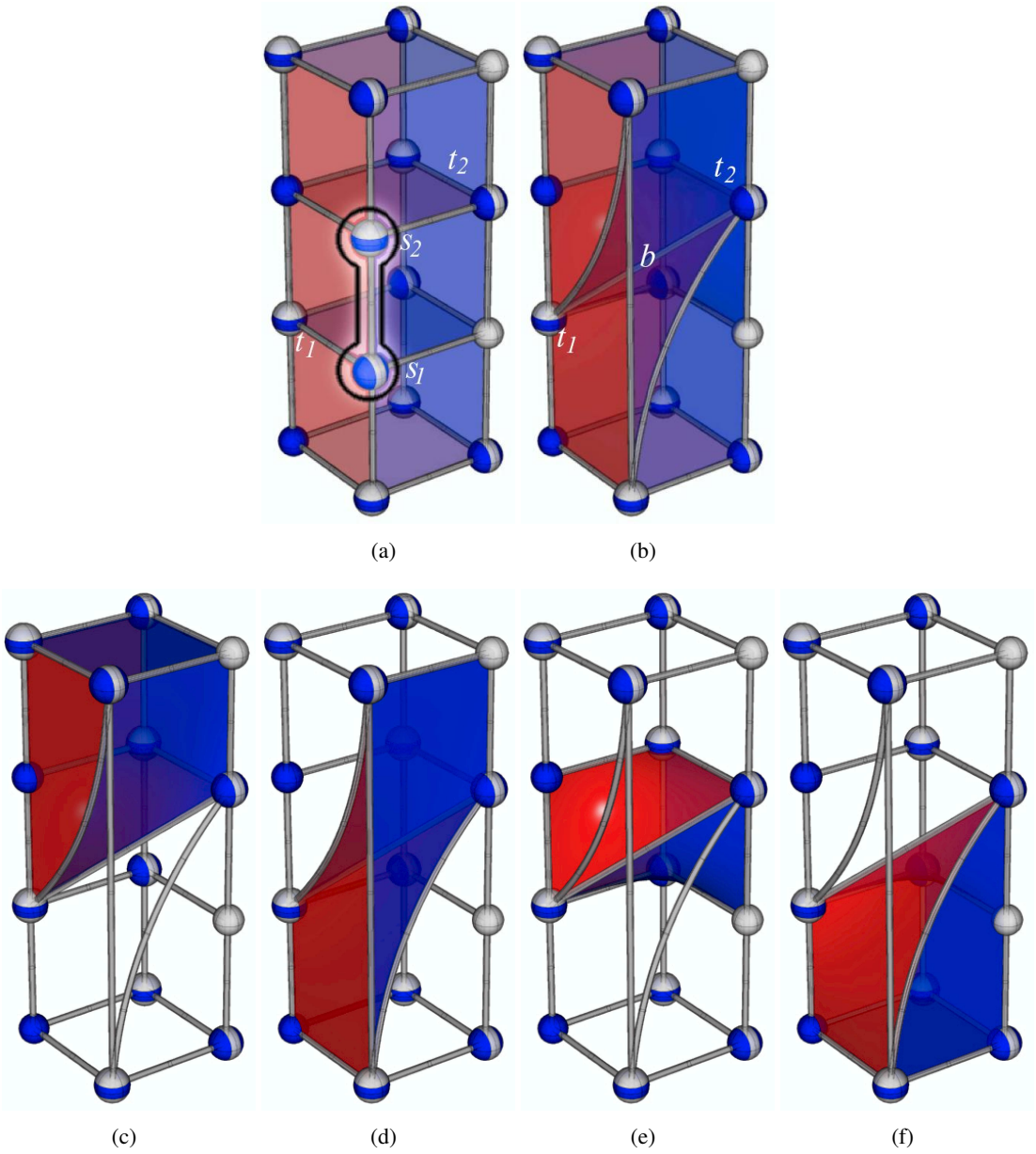


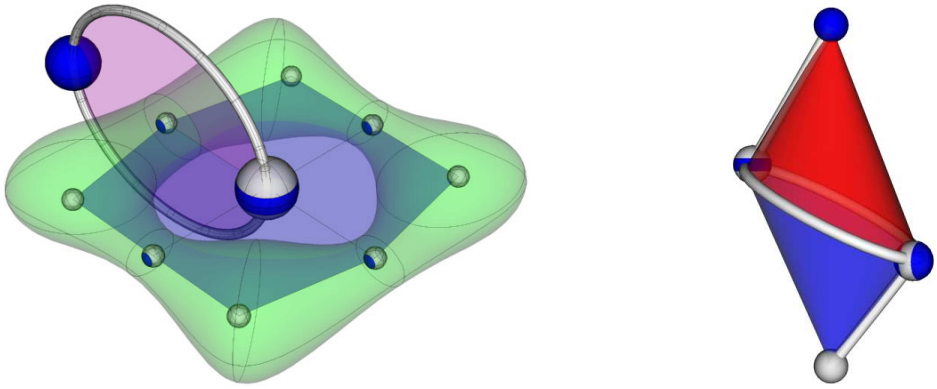
Figure 4.3.4: Cells destroyed and created by a saddle-saddle cancellation. The neighborhood of the saddle-saddle pair is divided into four regions of three crystals each. Only one region is shown for reasons of clarity. Cells within the other regions are modified in a similar manner. (a) Three crystals before cancellation of the highlighted pair of saddles. (b) Four crystals after cancellation. (c) Top crystal stretching down to the lower 2-saddle. (d) A crystal defined by a new minimum-maximum pair stretching from lower-left to upper-right corner of the region. (e) Middle crystal shrinking after losing the saddle-saddle pair. (f) Bottom crystal stretching up to the 1-saddle in the upper right.

A good way to think about reconnecting the complex after a saddle-saddle cancellation is in terms of ascending and descending disks: All surviving descending disks expand to share the boundary of the deleted disk, thereby creating connections between surviving 2-saddles and 1-saddles on the newly inserted boundary. Similarly, "surviving" 1-saddles connect to 2-saddles on the newly inserted boundary of their ascending disks, which guarantees full re-connectivity of the complex after a cancellation.

The Morse-Smale complex actually increases in terms of cells after a saddle-saddle cancellation since re-routing ascending and descending disks creates new intersections between them, thus adding new cells to the complex. Figure 4.3.4 shows the cells destroyed and created by this operation. For simplicity, these figures show only three of the twelve crystals changed by the cancellation and the cells that reconnect this complex within this region. Introducing new cells is counter-intuitive. Although the Morse-Smale complex grows in size, the function is smoothed by the removal of saddle pairs. Also, the cells created by the saddle-saddle cancellation are introduced into rings around saddle-extremum pairs. A future saddle-extremum cancellation will remove all these cells, leading to a smoother Morse function and, ultimately, a smaller Morse-Smale complex. A simple proof that the algorithm terminates, despite the increase in number of cells after a saddle-saddle cancellation, follows from the fact that every cancellation results in the removal of a pair of nodes from the complex.

In theory, the maximum number of cancellations is bounded by half the number of critical points. In practice, however, we stop earlier to preserve more persistent features. The saddle-saddle cancellation introduces significant complexity that does not occur in lower dimensions. Furthermore, the complexity of this cancellation indicates that extensions to higher dimensions are likely to be rather complicated. The intuition, however, that the ascending manifolds and descending manifolds of the two critical points involved merge to share their boundary, is an important starting point for such investigation.

An arc connecting two saddles does not guarantee a valid saddle-saddle cancellation. Performing a cancellation could lead to the formation of a *pouch*, see Figure 4.3.5. This situation occurs when a crystal incident on the arc has exactly two quads, one connecting the saddle pair to a min-



(a) (b)

Figure 4.3.5: A saddle-maximum pair that cannot be cancelled (a). Two integral lines beginning at a 2-saddle flow to the same maximum. Canceling the saddle-maximum pair causes a strangulation of the blue descending disk since integral lines terminating at the 2-saddle are left without a destination. A saddle-saddle pair that cannot be cancelled (b). Canceling the pair would create a crystal, called a pouch, that contains exactly one minimum and one maximum as boundary nodes. The boundary of such a pouch cannot be represented as a collection of quads.

imum and the other connecting them to a maximum. Removing the 1-saddle and 2-saddle creates a crystal with zero saddles and zero quads, corresponding to a possibly valid Morse function but resulting in an invalid combinatorial structure for the Morse-Smale complex. Pouches and strangulations (in the case of saddle-extremum cancellation) do occur in practice, which implies that there exists a minimal Morse-Smale complex that cannot be further simplified using legal cancellations.

The combinatorial validity of a Morse-Smale compelx is preserved by cancellations. No illegal intersection of ascending and descending manifold is introduced, and the non-cancelled critical points retain their index.

4.4 Augmented Morse Function

Let f be a function sampled at discrete locations, which are the vertices in a regular CW-complex K. We define the *augmented Morse function* f^α as a function on a refined version of K. Let K^α be the regular CW-complex attained from inserting a vertex at every cell $\sigma \in K$, with a connecting edge between $v_i^\alpha \in K^\alpha$ and $v_j^\alpha \in K^\alpha$ if and only if σ_i is a face or co-face of σ_j in K. Then define f^α as the following: the function value of a vertex $v_i \in K^\alpha$ is the maximum of the function

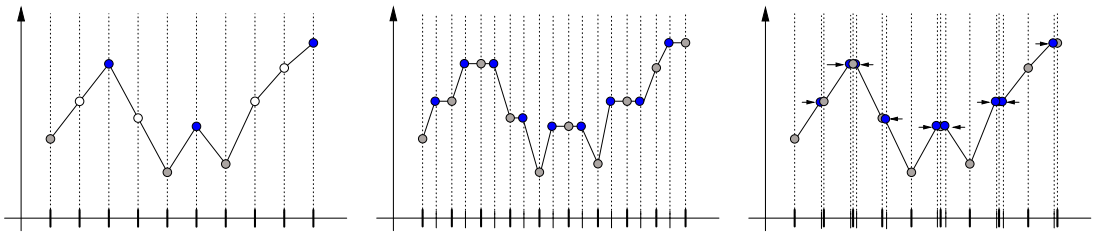
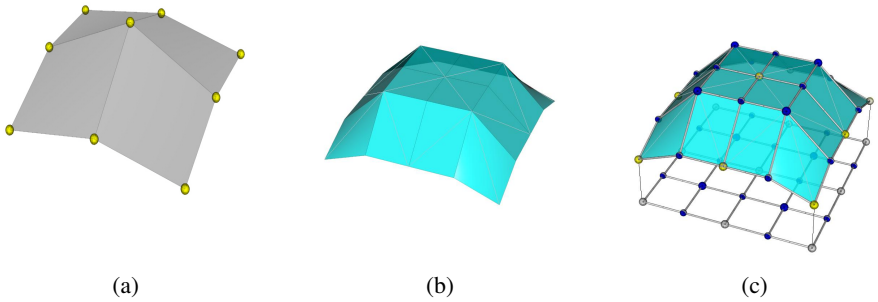


Figure 4.4.1: The function shown in (a) has finite sample points with a PL interpolant on edges. Maxima are shown in blue, and minima in grey. The augmented version of this function (b) inserts new vertices between every sample point, and also has a piecewise linear interpolant on edges. All vertices of the augmented function are all critical, and alternate between minima and maxima (note that the original vertices are now all minima). We can reposition the introduced vertices (c) of the new function so that the augmented PL function is ϵ-close to the original PL function.

values plus some ϵ of vertices that are faces of σ_i in K. This creates a triangulation of K with some special properties. Every vertex is a critical point in K^α. It is possible to reposition the newly added vertices such that a PL interpolation on the simplices of K^α is ϵ-close (maximum error between the two is bounded by some ϵ) to the PL interpolation of some triangulation of K. Figure 4.4.1 illustrates how this augmentation works for a univariate function.

Vertices of the augmented Morse function are all critical, and the index of criticality of $v_i^\alpha \in K^\alpha$ is identified by the dimension of the cell $\sigma_i \in K$ from which it was created. For example, when K is defined over a 3-manifold, the vertices that correspond to 3-cells are index-3 critical points, maxima, the vertices that correspond to 2-cells are index-2 critical points, 2-saddles, the vertices that correspond to 1-cells are index-1 critical points, 1-saddles, and the vertices that correspond to 0-cells are index-0 critical points, minima. The Morse-Smale complex of the augmented function is also trivially extracted. $v_i^\alpha \in K^\alpha$ and $v_j^\alpha \in K^\alpha$ are connected by an arc of the Morse-Smale complex if and only if σ_i is a facet or co-facet of σ_j in K. Furthermore, the descending manifold of $v_i^\alpha \in K^\alpha$ is simply its lower star. The ascending manifold of $v_i^\alpha \in K^\alpha$ is its upper star. The regularity of the critical points guarantees the combinatorial validity of cells of the Morse-Smale complex. Figure 4.4.2 shows how the augmented function has a predictable Morse-Smale complex for a two-dimensional scalar function, and Figure 4.4.3 illustrates how we subdivide a tetrahedron and a cube for three-dimensional scalar functions.



(a) (b) (c)

Figure 4.4.2: The function shown in (a) has finite sample points with a PL interpolant. The augmented version of this function (b) inserts new vertices between every sample point, and assigns a triangulation. Vertices of the augmented function are all critical, and alternate between minima, saddles, and maxima (note that the original vertices are now all minima). The Morse-Smale complex of the augmented function has a regular form (c).

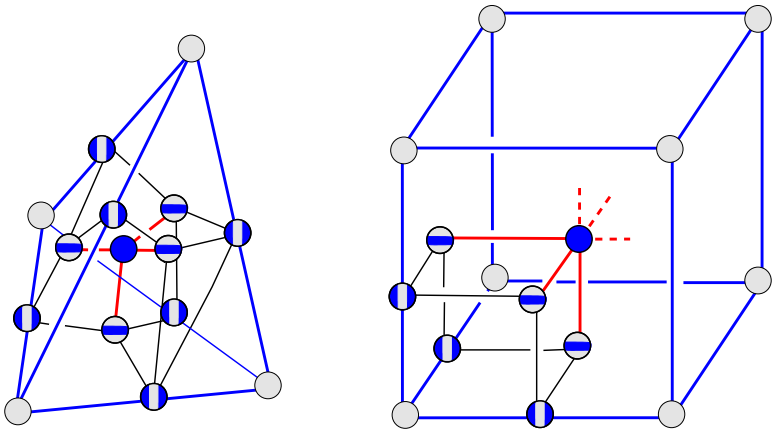


Figure 4.4.3: Creation of an artificial Morse-Smale complex by subdivision: Dummy critical points are introduced at barycenters of all cells thereby turning original data points into local minima. The function value at a dummy critical point is ϵ (an infinitesimally small positive value) greater than the largest value of the vertices of the cell in which it is located. Therefore, the dummy critical point has infinitesimally small persistence.

One of the results of constructing the augmented function is that we can identify the critical points and indices of criticality of this function without analyzing the neighborhood of a point, which makes it particularly attractive for possible extensions to higher dimensions. Furthermore, there are no degenerate critical points. It resolves multi-saddles in the original function by splitting them and moving the split saddles out to the edges and faces of K, and resolves flat regions through perturbation. Of course, in itself, the Morse-Smale complex of this function is useless: it conveys

no meaningful information. However, we solve this problem by applying multi-scale analysis: we cancel out the critical points that we introduced that are not significant.

4.5 Computing the Morse-Smale Complex

The following algorithm computes the Morse-Smale complex for three-dimensional scalar data:

1. Given a sampled function f defined on a finite embedable CW-complex K with convex cells, construct the augmented function f^α on K^α.

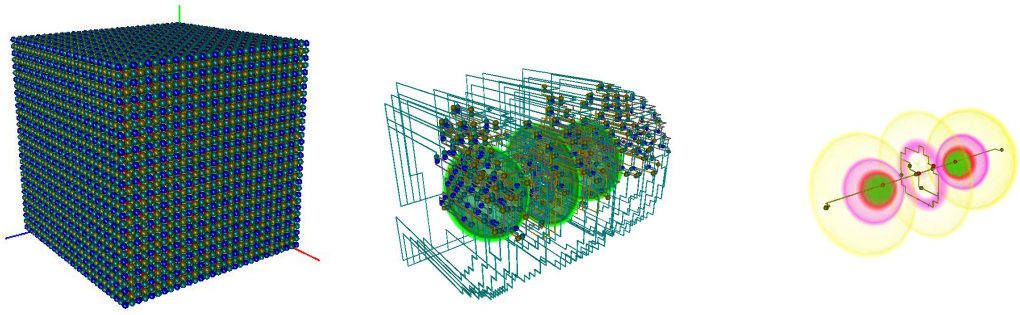
2. Compute the Morse-Smale complex of f^α on K^α.

3. While the lowest persistence arc in the Morse-Smale complex is less than some ϵ, cancel the pair of critical points connected by the lowest persistence arc in the complex.

We call the Morse-Smale complex of f^α on K^α the *artificial complex*. A disadvantage of starting with this artificial complex is that the number of nodes is equal to the total number of cells in the input mesh, thereby limiting the size of the data set that can be efficiently processed. However, low persistence pairs are removed and therefore exploration of the data can still be done interactively. Further simplification removes small features from the data, and can be done interactively. Figure 4.5.1 shows the steps in this algorithm, as well as the features extracted through further simplification.

4.5.1 Sweep Plane Approach

Representing the Morse-Smale complex of f^α on K^α is costly. Seven additional nodes (for cube lattices) are created in the complex for each data point. Thus, our earlier experiments [24] were restricted to data sizes up to $64 \times 64 \times 64$. We have overcome this restriction by utilizing the regularity of the artificial complex to pre-process the input in a streaming fashion. After removal of ϵ-persistence pairs, the size of the complex is proportional to the number of critical points in the original data. We create the complex incrementally: We start by adding a couple of slices of the artificial complex, and then cancel all ϵ-persistence pairs before adding another slice. We call this



(a) (b) (c)

Figure 4.5.1: The artificial complex of f^α on K^α (a) is simplified to remove all ϵ-persistence pairs (b). Further simplification of the complex reveals the features in the hydrogen atom dataset (c).

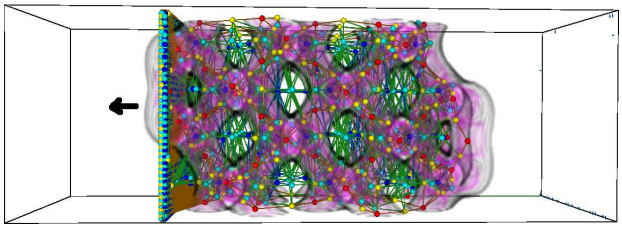


Figure 4.5.2: A snapshot of the artificial complex being constructed one-slice-at-a-time. The unprocessed region is to the left of the slice.

the "sweep plane," as it progresses through the data, adding one sheet of critical points at a time. We do not permit cancellations that affect the slice in order to maintain the regularity of the artificial complex. However, critical point pairs whose neighborhoods are disjoint from an incoming slice are valid pairs for cancellation. Figure 4.5.2 shows a snapshot of a complex constructed in this incremental fashion.

The streaming approach has several advantages. Most importantly, the size of the memory footprint is $O(n^{2/3}+k)$ instead of $O(n)$, where n is the number of vertices in the artificial complex of the data set and k is the size of the final Morse-Smale complex. We only need to store two full slices in addition to the complex at any time, as opposed to storing the entire artificial complex. Computation of the complex for larger data sets becomes possible via such a streaming method. We still pay the $O(n)$ time penalty however, since all "dummy" critical points must be removed. Therefore, datasets with a larger number of critical points will be pre-processed faster. The other

factor influencing pre-processing time is the maximum number m of arcs incident on any node. Since this is stored as a linked list (we will cover the data structures in section 4.6), search and deletion of arcs in this list pays an $O(m)$ penalty, and therefore can pose serious overhead during pre-processing. Although the streaming algorithm must perform the same number of cancellations as the non-streaming algorithm, the improved memory use and ordering of cancellations leads to an 8X speedup.

4.5.2 Boundary

Our input data is defined over a volumetric domain that has a non-empty boundary. We use a standard technique from point set topology, called *one-point compactification*, to convert the domain into a 3-manifold without boundary. The compactification involves addition of a *vertex at infinity* that connects to all boundary vertices. This extension of the triangulation is a simple and efficient way to handle the domain boundary [15, 43, 55]. Instead of explicitly adding the vertex at infinity and simplices connecting it to the boundary, we create a layer around the domain consisting of dummy critical points for each simplex that contains the vertex at infinity. These dummy critical points become nodes of the artificial complex and are removed when we cancel ϵ-persistence pairs. We restrict all cancellations to pairs that lie completely in the interior of the domain or within the boundary to ensure that we do not change the topology of the domain.

4.6 Efficiency and Data Structures

We have developed a new data structure for storing the Morse-Smale complex. The design of the data structure was governed by three major objectives: efficient execution of all simplification operations, minimal memory overhead, and rapid creation and deletion of nodes and arcs. The data structure stores two types of information: connectivity of the complex and geometry of each cell within the complex. The combinatorial structure of the complex is determined by the connectivity of nodes via arcs, and the geometric structure of the complex is given by the location of nodes, arcs, and all ascending/descending 2- and 3-manifolds. We store the connectivity of the complex as a

graph, and augment the graph with geometric components.

4.6.1 Connectivity of the Complex

We create a list of nodes and a list of arcs to store the connectivity of the Morse-Smale complex. Each node contains its index of criticality, a list of arcs that originate or terminate at this node, the function value at this point, and relevant geometric information. A design goal was to have fixed-size elements in the lists. Instead of storing a list of arcs incident at a node, we store a reference to exactly one such arc. All arcs have a reference each to the next arc that shares its end points. The list of arcs incident at a specific node is obtained by traversing these references. Besides these two references, each arc also has references to its two end point nodes and geometry. Figure 4.6.1 shows our data structure. All nodes of the complex are stored in an array, each element containing multiple fields. G0 refers to the geometric location of a node. The TAG field stores the index of criticality as well as internal flags to represent boundary conditions. If the node is a saddle, then the field G2 stores a reference to the geometry of the 2-manifold originating from that saddle; otherwise, the critical point is an extremum, in which case the field G3 stores volumetric information of the 3-manifold originating from the extremum. The field A stores a reference to the first arc in the list of arcs incident to this node. Arcs are stored in an array as well, each entry again containing multiple fields. G1 is a reference to the geometry of the arcs (i.e., the integral line), CP1 and CP2 are the two end points of the arc, and A1 and A2 store the next element in the list of arcs incident at CP1 and CP2, respectively.

4.6.2 Geometry of the Complex

We have designed the geometry components to minimize complexity of cancellations. These components are:

- G0 is a reference to the location of a node, identified by an index.
- G1 is a reference to the path of the arcs, a set of line segments.
- G2 is a reference to the geometry of the ascending/descending 2-manifolds.

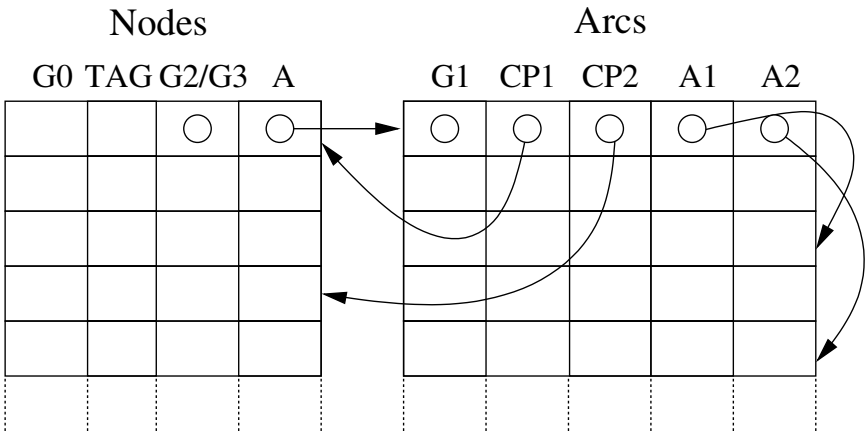


Figure 4.6.1: Data structure for connectivity of nodes and arcs in the Morse-Smale complex. Nodes and arcs are stored in two arrays. Each entry in these two arrays contains multiple fields that together store the connectivity of the Morse-Smale complex.

- G3 is a reference to the geometry of the ascending/descending 3-manifolds.

G0 is an index referring to an array of input vertices that determines the geometry of a critical point. Storing the geometry of arcs and ascending/descending 2-manifolds is more involved since we want to minimize storage. The key observation is that, upon simplification, arcs and 2-manifolds often merge, and the same line segment or face becomes a part of multiple arcs. This is a common behavior in the pre-processing stage of our simplification algorithm. After a cancellation, several arcs and surfaces are re-routed to pass through the same geometry. Our data structure takes advantage of this redundancy. G1 and G2 are stored as directed acyclic graphs (DAGs), as shown in Figure 4.6.2. All leaves of the DAG reachable from a given element in the DAG together constitute the geometry associated with that element. An arc references exactly one element in the DAG to recover the complete geometry of the constituent line segments. The leaves of this DAG are the geometric primitives of the line segments. Similarly, the leaves of the DAG storing the geometry of 2-manifolds contain the faces that compose the 2-manifolds. Each element in the DAG also stores the number of elements that directly reference it. The element is deleted when this counter is zero.

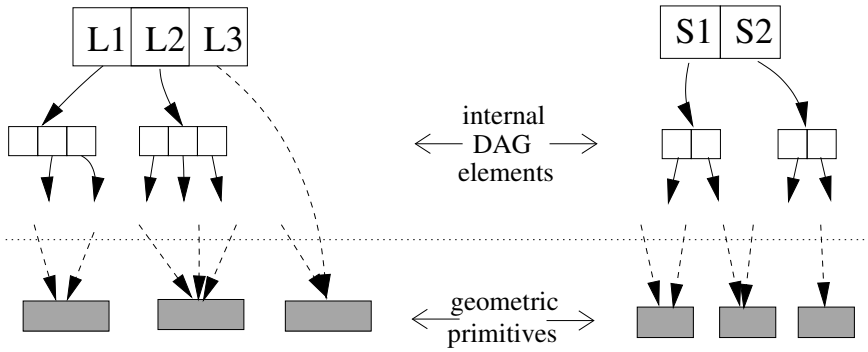


Figure 4.6.2: Geometry of arcs and ascending/descending disks are stored as DAGs to minimize storage. *Left*: our simplification algorithm creates new arcs by merging exactly three other arcs. Each internal element stores references to three children. *Right*: after a cancellation, two disks merge into one. So, each internal element stores references to two children.

4.6.3 Cancellations on the Complex

We review the cancellations in terms of these datastructures. The saddle-maximum cancellation is similar to its two-dimensional analog, which can also be interpreted as merging three critical points. We merge neighboring cells in the ring around the saddle-maximum arc. Therefore, besides removing two critical points, this cancellation also removes several crystals, quads, and arcs from the complex. Let s and m be the saddle and extremum to be cancelled and a be the arc connecting them. We implement the cancellation by performing the following sequence of connectivity-modifying operations:

1. Let n be the end point of the arc originating from s and not equal to m. Let b be this arc that connects s to n.

2. Replace all arcs a_i that terminate at m with new arcs b_i that share their origin with a_i and terminate at n. Add b_i to the arc list of its origin and the arc list of n.

3. Delete nodes s and m and all arcs incident to them. Delete all references of these arcs from the arc lists of their endpoints.

The geometry of arc b_i is obtained as the concatenation of arcs a_i, a, and b. References to the geometry of these arcs are added from a new element added to the DAG. All arcs b_i share the

geometry of a and b, which makes the DAG an efficient representation of the geometry. The single 2-manifold corresponding to s is deleted from the complex along with the saddle. Re-direction of arcs as described above changes boundaries of the 2-manifolds that it touches. However, it does not change their surface geometry, and the interior of the ascending/descending 2-manifolds remain simply connected after the cancellation. Therefore, we make no changes to the representation of the surface geometry of these 2-manifolds.

The saddle-saddle cancellation creates new cells in the complex, however, this is recorded implicitly through the merging of elements in our data structure. Let s_1 and s_2 be a 1-saddle and 2-saddle pair connected by an arc a. We have implemented the cancellation as a sequence of operations on our data structure:

1. For each pair of 1- and 2-saddles (t_1, t_2) different from (s_1, s_2), where the pair (t_1, s_2) is connected by arc a_1, and the pair (t_2, s_1) is connected by arc a_2, do the following:

   1.1) Create a new arc b from t_1 to t_2.

   1.2) Insert b into the arc lists of its endpoints.

2. Delete all arcs in the arc lists of s_1 and s_2 and delete the two nodes.

New arcs b are created by concatenating the geometry of arcs a_1, a, and a_2. The 2-manifolds of all 1-saddles t_1 merge with the 2-manifold of s_1, and the 2-manifolds of all 2-saddles t_2 merge with the 2-manifold of s_2. Deleting s_1 and s_2 removes a reference to their 2-manifolds. However no geometry needs to be removed since this surface geometry has become part of many ascending/descending 2-manifolds. When two surfaces merge, we create a new element in the DAG, add references to the merged surfaces, and include a reference to the new element from the saddle.

4.7 Examples and Results

We pre-process the input data by first creating an artificial complex and then removing several dummy critical points by canceling ϵ-persistence pairs. This process loads one slice at a time to

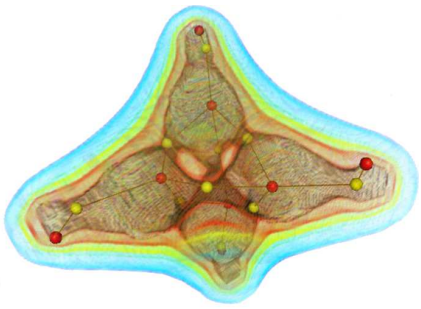


Figure 4.7.1: Atoms and bonds in the C_4H_4 molecule are identified by high-persistence critical points and ascending arcs in the simplified Morse-Smale complex.

construct the Morse-Smale complex incrementally. We perform further simplification of the Morse-Smale complex in an interactive process to identify features. Once the dummy critical points are removed, the complex provides an efficient representation of features in the original data. We identify important features as regions associated with persistent critical points. Similarly to Takahashi et al [55], we automatically design a transfer function to enhance critical values that correspond to these features. Our simplification allows us to limit the number of critical values affecting the transfer function to exactly those representing important features.

4.7.1 Feature Identification

We show that our simplification technique extracts known features and removes noise in well-studied data sets. Table 4.1 lists the data sets used, their sizes, and the time needed for pre-processing. All experiments were performed on a desktop PC (3GHz Pentium 4, 1GB RAM). The reduction in memory required for the pre-processing stage allows us to handle large data sets when compared to prior results [24]. After pre-processing, the remaining critical pairs can be removed interactively. The first data set represents an electron density distribution in a C_4H_4 molecule. Once ϵ-persistence critical point pairs are removed, the complex correctly outlines the bond structure of the C_4H_4 molecule. High-persistent maxima correspond to locations of atoms in the molecule, and ascending arcs that connect 2-saddles with these maxima correspond to bonds between atoms, see Figure 4.7.1. This correspondence is a visual depiction of the topological approach to identifying atoms in molecules as proposed by the AIM theory [1].

Table 4.1: Data sets used in experiments. Data set sizes, and timing results for pre-processing (t_{pp}).

Data set	Size	t_{pp}
C_4H_4	$33 \times 33 \times 33$	5s
Silicium	$34 \times 34 \times 98$	15s
Neghip	$64 \times 64 \times 64$	2m 35s
Aneurysm	$128 \times 128 \times 128$	30m
Hydrogen	$128 \times 128 \times 128$	45m

The other data sets listed in Table 4.1 were obtained either from simulations or from x-ray scans. Visualization of our results for these data sets are shown in Figures 4.7.2 and 4.7.3. The hydrogen atom data set describes the spatial electron density in a hydrogen atom subjected to a large magnetic field. The data set exhibits high density around the nucleus, two regions of high density on either side, and a torus of high density around the nucleus. We correctly identify these features. After the ϵ-persistence critical points are removed, there are still two disks of saddles separating the maxima in the data set, which correspond to noise. Initial cancellations remove the spurious 1-saddles and 2-saddles, leaving behind four maxima, which represent the four regions of high electron density.

Distinctive features in the silicium, neghip, and aneurysm data sets are revealed using a low threshold. We observed that a threshold value of 10% of the maximum persistence suffices to detect and remove all "insignificant" features. This can clearly be seen in the neghip data set, where we are able to isolate the different clusters of atoms automatically. We do not, however, expect that the same threshold should be used for all types of data sets. Rather, we view the simplification as an interactive filter that will allow scientists to explore the data at different scales.

4.7.2 Noise Removal

We have used synthetic data to illustrate how the cancellation of critical point pairs removes noise in a natural manner and hence leads to a robust identification of features. Our input is a simple sum of Gaussian distribution-like functions that decrease radially from seed points. Each radial function contributes to a local maximum at its seed point. The local maximum at the center of the domain is the largest and those near the domain boundary are the smallest. Figure 4.7.4 shows how the function is successively simplified by removing critical points based on persistence. The smaller

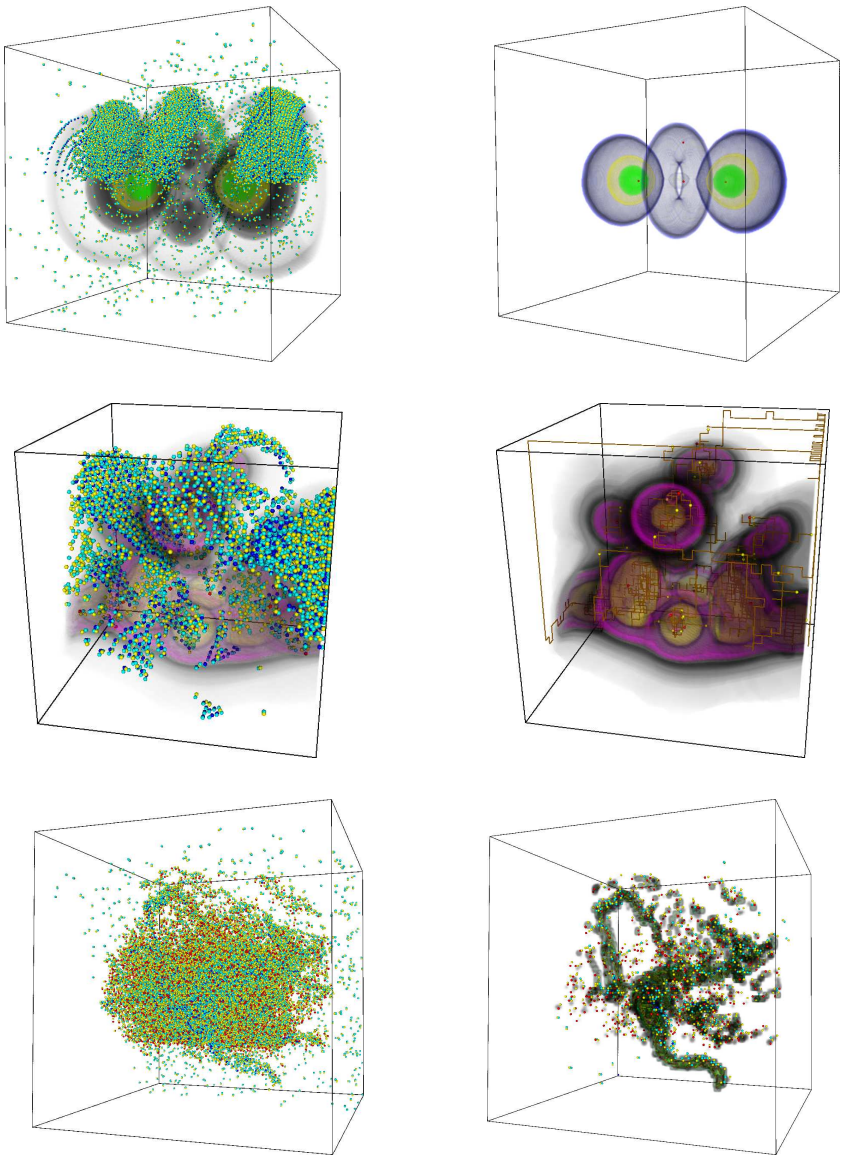


Figure 4.7.2: Topological simplification applied to various data sets (top to bottom: hydrogen, neghip, and aneurysm). The input (left) has a large number of critical points, several of which are identified as being insignificant and removed by repeated application of two atomic operations. Features are identified by the surviving critical points and enhanced in a volume-rendered image, using an automatically designed transfer function (right).

maxima represent small perturbations in the data and hence represent noise. The corresponding maxima have low persistence and are removed early during the simplification process. Negligible local maxima are removed first followed by the next tier of maxima with small but considerable

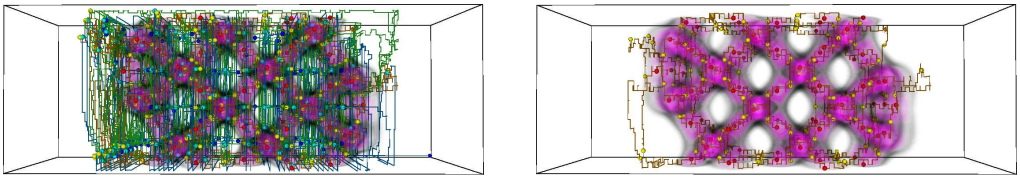


Figure 4.7.3: Topological simplification of the silicium data set. Cancellation of low-persistence critical pairs reveals the shape of the silicium lattice structure.

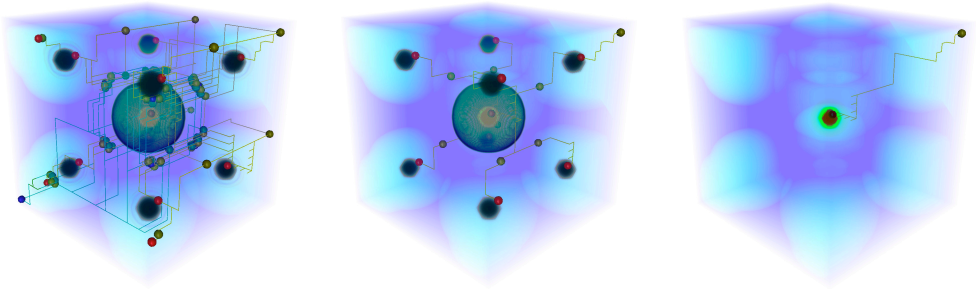


Figure 4.7.4: Noise in a synthetic function introduces features with negligible persistence. Left: The function consists of various spikes with the central one being the largest. Each spike is visualized as a sphere in the volume-rendered image. Middle: All nine spikes are clearly visible after removing noise that created the thin shells surrounding the spheres. Right: Further simplification destroys all maxima except the one representing the crucial features.

persistence, leaving behind the primary feature/maximum at the center.

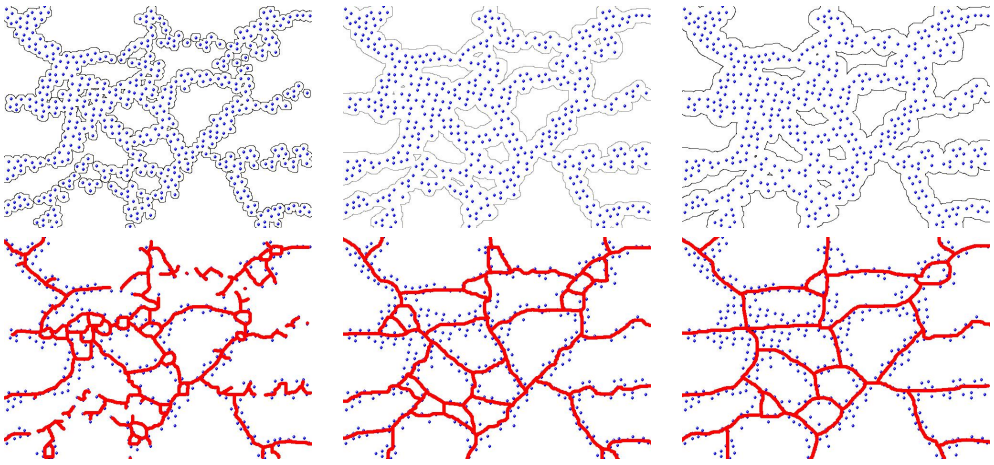
Chapter 5

Case Study: Finding Core Structures in Porous Media

Analysis of the results obtained from material simulations is important in the physical sciences. Our research was motivated by the need to investigate the properties of a simulated porous solid as it is hit by a projectile. The material is available as a distance field that is computed using a standard approach to an interface surface, and simplifies this field using ideas from Morse theory. The interface surface marks the boundary of "inside" and "outside" the material, and is embedded in a volumetric dataset. We present a procedure for identifying and extracting a feature set through analysis of the Morse-Smale complex, and apply it to find the filament structure of the porous solid [23]. This method produce a curved skeleton representation of the filaments that helps material scientists to perform a detailed qualitative and quantitative analysis of pores, and hence infer important material properties. Furthermore, we provide a set of criteria for finding the "difference" between two skeletal structures, and use this to examine how the structure of the porous solid changes over several timesteps in the simulation of the particle impact.

Several factors make straightforward analysis of the filament structure difficult. First, the scale of the structure we wish to identify is not known. While scientists may have a general idea of the size and distribution of the structure, any analysis based on such guesses would be skewed. In addition, analysis tools should be valid for any size and distribution of the structure. Second, there is inherent instability in any choice of criteria for identification of such a structure. There are several classes of methods for identifying curved skeletal structures [11], however, they assume a surface representation is given, and construct a skeleton based on that surface. The interface surface of the distance field representing the porous material ideally can be extracted as surface for isovalue zero. However, any choice of isovalue to select a "base surface" is unstable; Figure 5.0.1 shows how small changes in isovalue can produce profound differences between the resulting structures being the basis for analysis. Furthermore, the distance field itself has noise, artifacts from computation, and artifacts from discretizing a continuous function onto a grid. As a result, straightforward analysis of the distance field yields an excess of critical points that do not represent physically meaningful and relevant features of the function.

We overcome these difficulties by constructing the Morse-Smale complex for the distance field, and performing an analysis of the arcs to find "stable" thresholds from which to construct a filament



(a) (b) (c)

Figure 5.0.1: Dependency of extracted core structure and the radius assigned to atoms. Atom locations are given along with a radius for each atom (top). This radius determines an interface surface for what is considered "inside" and "outside" the solid material. The extracted core structures (bottom) can vary to a large degree with only a slight change in the radius of each atom. As the radius increases, the coarser structure represents a larger aggregation of atoms, however, finer details are lost. No single radius can be considered as "correct," since the core structure of any radius is not stable under small changes to the radius.

representation of the porous solid.

5.1 Filtering Arcs of the Complex

Meaningful and important features of a given function are not always captured by the notion of persistence. For example, a scientist may be interested in the function's behavior within a region enclosed by certain isosurfaces. In this case, simplification should ideally preserve the topological structure of the isosurface components while removing noise in the volumetric region inside and outside. Extrema with function value within a given range may correspond to relevant features, and in this case simplification should leave these extrema unaffected. Both cases arise for the distance fields that we study. In fact, features may arise in locations not initially predicted. The Morse-Smale complex is a useful tool in identifying such features since it provides a full characterization of the gradient flow behavior (when viewing the function's gradient as a flow field). Therefore, analysis of the critical point pairs and arcs of the complex can lead to better understanding of the actual

locations of the features, and where to apply topological simplification.

The porous solid dataset suffers from the fact that the distance field was created from an interface surface that is unstable. A small change in the selection of this surface could lead to a profound difference in the topology. However, by having relaxed notions of the exact location of this interface, we can overcome the instability and produce a result that is invariant under small changes in selection of the interface surface.

We use several filters to direct our simplification process to preserve relevant features in the data (relevance understood here as user-specified features for a particular application). A filter specifies the arcs of the Morse-Smale complex that are to be removed from the list of candidates for cancellation. Any filtering requirements can be met with the following three conditions:

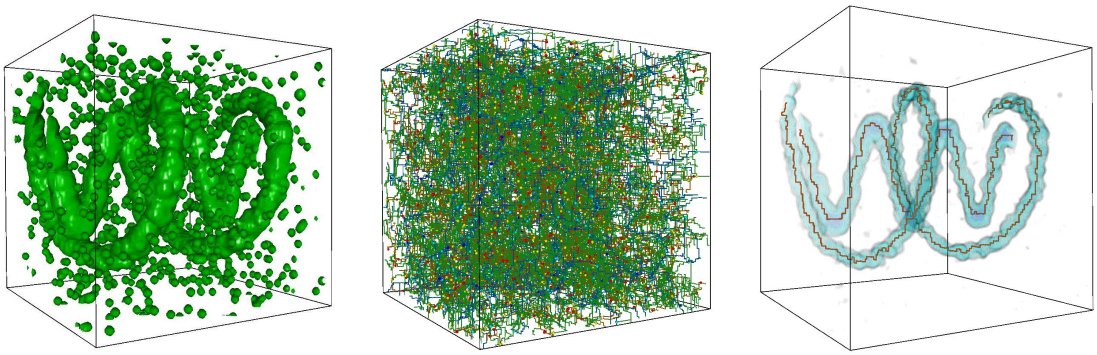
i Arcs that have their lower, upper, both, or neither end points in a given range of function values.

ii Arcs that cross a given isovalue.

iii Arcs whose lengths lie within a given range.

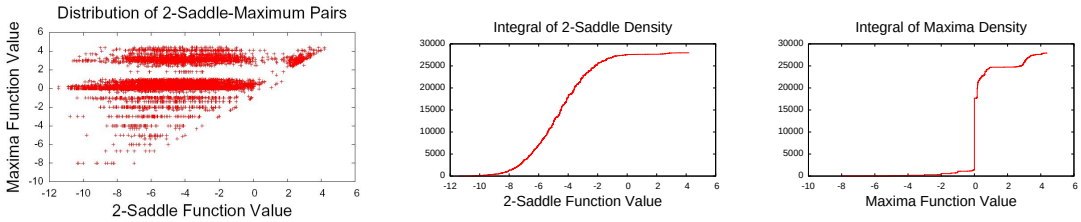
These criteria, or combinations thereof, designate a wide range of features.

We show an example where the distribution of critical point pairs help distinguish between actual features and artifacts in Figure 5.1.1. In this example, we created a distance field using a standard approach as the signed distance from the shells of a set of atoms distributed along a spiral and a sinusoidal curve. The atoms are placed at random along these curves, and additional "noise" atoms are added throughout the data. We compute the Morse-Smale complex for this distance field. We wish to extract the curved skeleton from this distance field, without knowing *a priori* the details about how the distance field was constructed. Intuitively, we can guess that the features will be represented by 2-saddle - maximum pairs where the maximum has large positive value and the 2-saddle has large negative value. The distribution of critical point pairs, illustrated in Figure 5.1.2, suggests certain stable thresholds for "important" maxima and 2-saddles. In particular, these pairs are those in the upper-left corner of the scatter plot. Flat regions in the integral along each axis reveal that the stable threshold for maxima is two, and the stable threshold for 2-saddles is -1, where this curve first starts to flatten out. By canceling all arcs that do not entirely cross the range [-1,2],



(a) (b) (c)

Figure 5.1.1: The isosurface for isovalue zero of the initial distance field (a). We compute the Morse-Smale complex of this field (b), and apply filtering to extract the stable core structure (c).



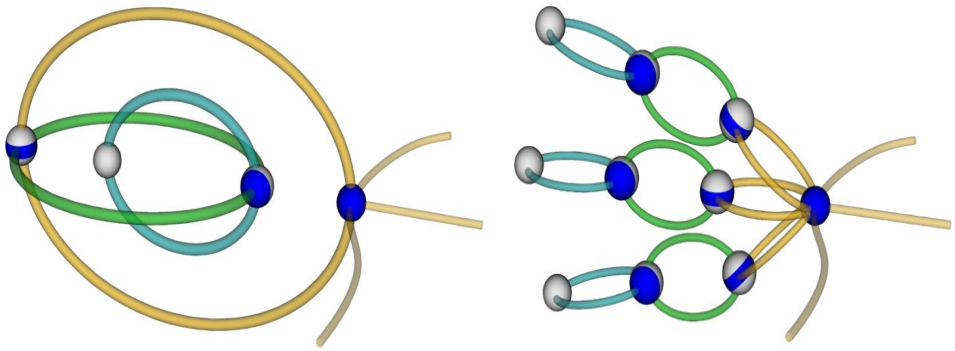
(a) (b) (c)

Figure 5.1.2: The distribution of 2-saddle - maximum pairs (a). Each pair is plotted as a point whose coordinates are the function values of the 2-saddle and the maximum. We integrate along the x-axis (b) and y-axis (c) to see the density distribution of 2-saddles and maxima.

we remove the artifacts and noise. The core structure is extracted as the 2-saddle - maximum arcs that remain and are entirely contained in the isosurface for isovalue zero.

5.2 Fingers and Reordering

Persistence-based simplification of the Morse-Smale complex can result in configurations that cannot be reduced. Such a situation arises when a cell of the complex contains three pairs of critical pairs, none of whom can be canceled because of an obstruction, as shown in Figure 5.2.1(a). Cancellation of any of the three critical pairs results in an invalid Morse-Smale complex as we show in section 4.3 Canceling the saddle-extremum pair leads to a *strangulation* of the cell whereas canceling the saddle-saddle pair results in the creation of a *pouch*. The cell cannot be removed by any



(a) (b)

Figure 5.2.1: Certain orders of cancellation result in finger-like configurations. (a) A single finger contains three critical pairs none, of which can be removed by a valid cancellation, even if the corresponding arcs have low persistence. (b) Fingers could accumulate leading to visual artifacts.

sequence of cancellations. These configurations, called *fingers*, are artifacts that result from our choice of order of cancellations in flat regions of the function. We introduce flat regions throughout the function to create the evenly spaced artificial complex. Therefore, these fingers can accumulate in large numbers as shown in Figure 5.2.1(b). In fact, the maximum persistence within such a finger may be much smaller than the persistence filter and yet the small feature cannot be removed. While the structure of the complex remains combinatorially sound, the fingers add to visual clutter, and should be removed. This is especially important in analysis of the porous solid, since fingers might erroneously indicate structural components of the distance field.

We use the sweep plane algorithm for construction of the Morse-Smale complex with slight modifications. We prevent the creation of fingers by reordering the cancellations in the pre- processing stage of this algorithm. All saddle-extremum cancellations are scheduled before the saddle-saddle cancellations to ensure that none of the ϵ-persistent saddle-extremum pairs remain. While obstructions can still develop, they are resolved by future saddle-saddle cancellations. Additionally, we perform all possible cancellations within a single slice of the data before canceling critical pairs that span multiple slices. In particular, this removes all minimum-1-saddle pairs within the slice. Since 2-saddle-maximum pairs span multiple slices, this reordering prevents the simultaneous existence of un-cancellable minimum-1-saddle and 2-saddle-maximum pairs.

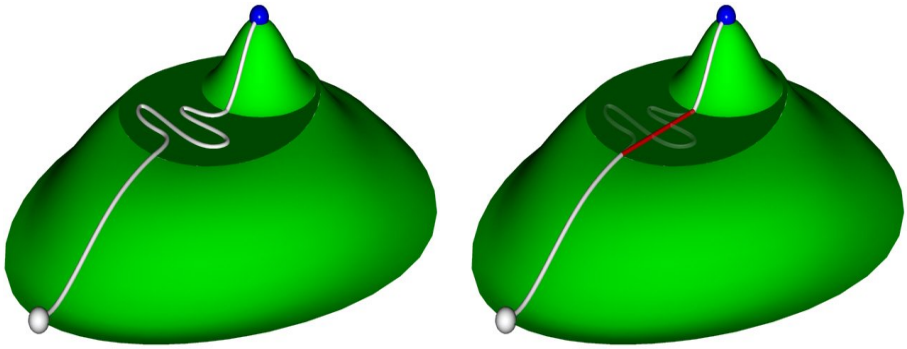
5.3 Arc Smoothing

The algorithm described in section 4.5 for constructing the Morse-Smale complex from the artificial complex results in arcs that contain geometric artifacts: the sequence of line segments representing the arc may differ slightly from the location of the corresponding integral line. This discrepancy occurs because the artificial complex introduces several small regions of constant value, "flat regions," to the function. Integral lines are not uniquely defined in these flat regions, and therefore the arcs we produce may wander before resuming the path of steepest ascent, see Figure 5.3.1. Figure 5.3.2 shows how these flat regions can introduce sharp spikes in the arcs. In fact, while the combinatorial structure of the complex is correct, the geometry of the arcs may be off from the integral lines by one voxel in any direction. For use in the analysis of the distance field, we want the arcs to behave like simple curves that can be represented as a sequence of line segments.

The spikes we introduce are the result of cancellations in flat regions. Previous approaches [5] perturb the function to avoid flat regions. However, the necessity to reorder cancellations to avoid creation of fingers in our approach prohibits such a solution. Instead, we shift the arcs towards the integral line using shortcuts. The spikes in the arcs have a unique property that they occur entirely within a single hexahedral cell, being a unit cube (or voxel) in our case, around the arc. These unit cubes have the property that they have unique points where the arc enters p_{entry} and where the arc exits p_{exit}. We consider the slope of any line with end points x_0 and x_1 in the cube as $(f(x_1) - f(x_0))/\|x_1 - x_0\|$ and therefore, given p_{entry} and p_{exit}, the maximum slope is in the direction that minimizes the distance. Hence, we use a shortest-path algorithm within unit cubes to shortcut the spikes and yield a smoother arc.

5.4 Core Structure of a Porous Solid

We compute the Morse-Smale complex using the sweep plane algorithm with the finger removing and arc smoothing modifications. Then, similar to our previous example shown in Figure 5.1.1, we analyze the critical point pairs, and filter the arcs to extract the core structure. The full complex shown in Figure 5.4.1, is used to plot the distribution of 2-saddle - maximum pairs, shown in Figure



(a) (b)

Figure 5.3.1: The geometric location of an integral line in a flat region is arbitrary. Therefore we shortcut the path, and find the shortest path through the flat region.

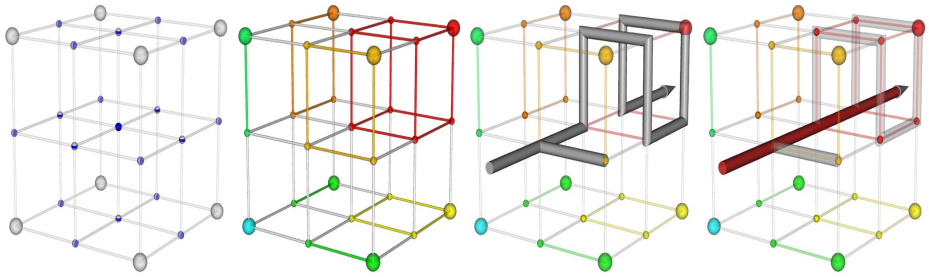
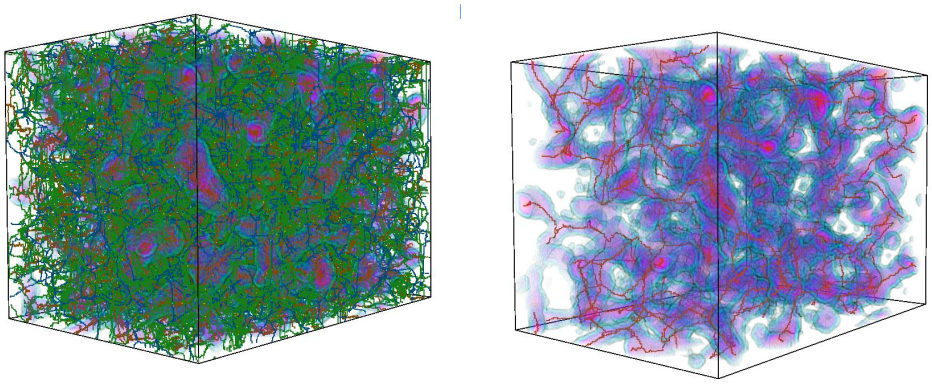


Figure 5.3.2: Artifacts from the construction of the complex include flat regions inside every voxel. Therefore, an integral line can take any valid path from the point of entry in a voxel to the point of exit. We again remove kinks in the integral line by shortcutting the path.

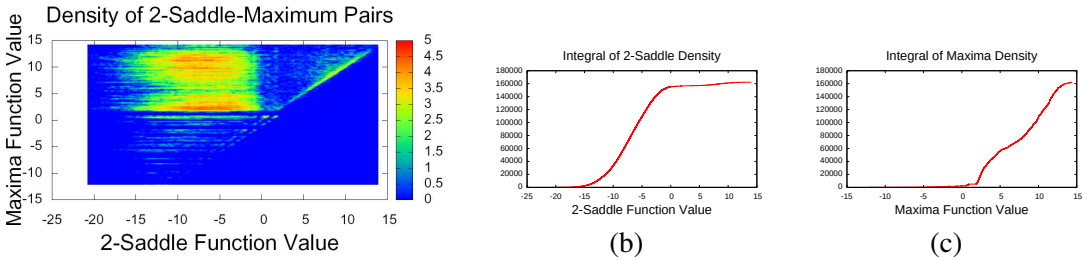
5.4.2. The features we are interested in are the arcs that connect a low 2-saddle to a high maximum; these critical point pairs are in the top left of the distribution. Flat regions in the integral along each axis reveal that the stable threshold for maxima is 1.5, and the stable threshold for 2-saddles is $-.8$. By canceling all arcs that do not entirely cross the range $[-0.8, 1.5]$, we remove the artifacts and noise. The core structure is extracted as the 2-saddle - maximum arcs that remain after simplification and are entirely contained in the isosurface for isovalue zero. For this particular application, we are interested in the connectivity of the porous solid, therefore we omit arcs that are connected to the structure at only one endpoint from the final core structure.

The results were generated using an off-the-shelf personal computer, a 3.4GHz Pentium 4, with 2 Gb of memory. The porous solid was represented as real-valued samples on a $230 \times 230 \times 375$ reg-



(a) (b)

Figure 5.4.1: The initial computation of the Morse-Smale complex for the full dataset (a) is simplified revealing the graph structure (b) of the porous solid.



(a) (b) (c)

Figure 5.4.2: The distribution of the 2-Saddle-Maximum pairs (a). The color scale shows the number of occurrences of 2-Saddle-Maximum pairs where the 2-Saddle has function value along the x-axis, and the Maximum has function value along the y-axis. We integrate along the x-axis (b) and along the y-axis (c) to find stable threshold values.

ular grid. The total time required for computation of the initial Morse-Smale complex was 6 hours 32 minutes and 45 seconds. Additional processing to attain the graph structure took 32 seconds. After the initial computation, exploration and further simplification can be done interactively. For larger datasets, computation of the full Morse-Smale complex is not possible using the sweep plane algorithm; however, we can still perform analysis on a subset of the data to attain the distribution of critical point pairs, assuming features are distributed nearly uniformly in the dataset.

5.5 Comparison of Core Structures

We compare two core structures using a number of heuristics:

i Hausdorff distance - the maximum geometric distance between points on one graph and their closest neighbor on the other. To obtain a symmetric measure, the geometric distance is computed in both directions and the larger value chosen.

ii Average distance between closest pairs on the two graphs.

iii Number of simple cycles in each graph, to estimate connectivity.

iv Total length of edges in each graph.

We computed the core structure using the Morse-Smale complex for each timestep. Using these distance measures, we can compare qualitatively how the structure of the material changes between timesteps. Figure 5.5.1 is a visual depiction of the displacement of filament segments at different times after impact. Note the large displacements near the crater, and the nearly zero displacement well below the crater.

It can be seen from a visual inspection that the density of the material barely changes in the bottom one-third of our sample. This is partly due to the fact that the foam is extremely efficient at absorbing the impact shock wave. Key statistics of the core structure for each time steps are summarized in Table 5.1.

Metric	t=500	t=12750	t=25500	t=51000
# Cycles	762	340	372	256
Total Length	34756	24316	23798	18912

Table 5.1: Statistics for each timestep.

The ratio of cycle counts before and after the impact supports this observation, as approximately two-thirds of the cycles are destroyed. The ratio of the total length of the filaments before and after the particle impact implies that volume of material displaced by crater is approximately one-half the volume of the rest of the material. Since this ratio is fairly close to the ratio of the cycle counts, we can say that the majority of the filaments that were broken happened to be in the interior of the crater. The sum of the Hausdorff distances between the timesteps is 98.6, giving the maximum distance that any element of the material travelled during the impact. This number is surprisingly

high, corresponding to the entire depth of the crater; it indicates that the material of the filaments first hit by the particle was displaced along the trajectory of the particle. The average distance between closest pairs in the graphs of the consecutive timesteps was less than 5.0, indicating that the displacement did not propagate into the material, outside the direct path of the particle.

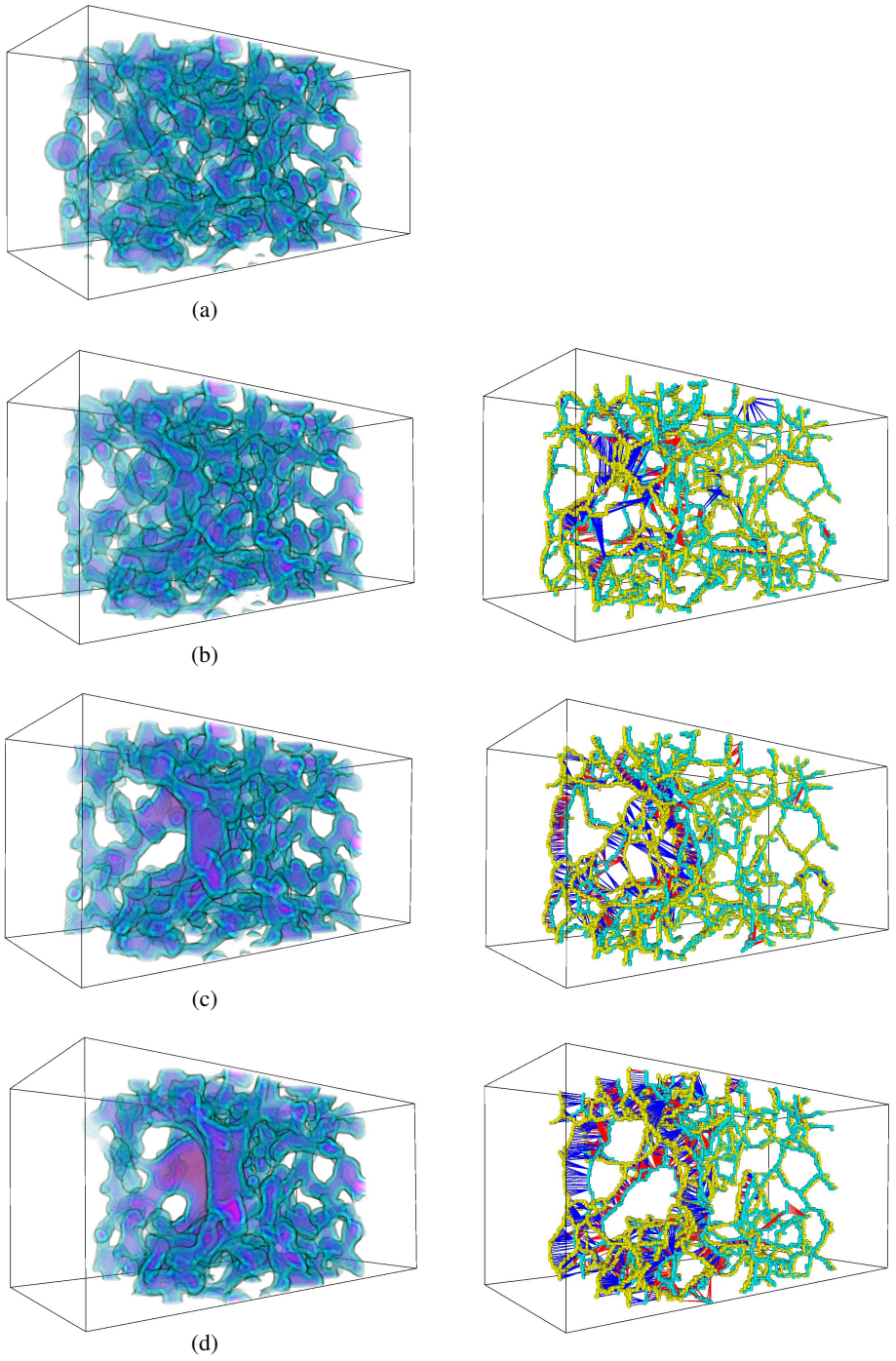


Figure 5.5.1: *left:* A volume rendering of the impact of the ball entering the porous solid from the left at time step 500 (a), time step 12750 (b), time step 25500 (c), and time step 51000 (d). *Right:* We compare the core structures of consecutive time steps. The yellow dots represent the core structure of the initial time step, and the teal dots represent the core structure of the next time. The closest arcs between in the core structures at the different time steps are connected via blue and red line segments. The length of these segments corresponds to the displacement of the arc.

Chapter 6

Computing Discrete Morse-Smale Complexes

In this chapter, we present a new algorithm for constructing a consistent Morse-Smale complex, and a framework which utilizes a divide-and-conquer strategy for dealing with large scale data in a variety of data formats and of any dimension. The kernel of our algorithm computes the discrete gradient on a *parcel* of the input data, generates a Morse-Smale complex from the gradient on the parcel, and then merges the Morse-Smale complexes together across the boundaries of the parcels. We use the discrete formulation of Morse theory as opposed to the continuous formulation for two reasons: it is simpler to implement because there are no special cases when dealing with higher dimensional components of the Morse-Smale complex; and it makes it possible to constrain the gradient flow on the boundary of parcels to enable a stratified approach. The foundations of discrete Morse theory are presented in section 6.1. We extend the definition of cancellations in a Morse-Smale complex by fully characterizing the effects of a cancellation operation in any dimensional complex in section 6.2. Finally, section 6.3 surveys previous work in identifying flow and Morse complexes in the discrete setting.

The kernel of this new approach is the algorithm from generating a discrete gradient field from a scalar function, where only the vertices of a mesh initially are assigned scalar values. We present this algorithm in section 6.4. Many valid discrete gradient fields can be computed for the same set of sample values, and we discusses the quality of the gradient found by this algorithm. Finally, a Morse-Smale complex is computed for the discrete gradient field through a systematic sweep of the ascending and descending manifolds, as presented in section 6.4.4.

We achieve a divide-and-conquer algorithm for computing the discrete gradient by dividing the data into parcels, processing them independently, and merging them. The discrete gradient and Morse-Smale complex are computed independently on each parcel, and only the Morse-Smale complex and gradient on the boundary of the parcel are necessary to merge parcels. We can control the size of the parcels and size of the Morse-Smale complex through simplification to obtain a memory-efficient algorithm. We discuss the steps in this algorithm in section 6.5.

Finally, we present the algorithm in a framework in section 6.6 that makes it possible to implement multiple data formats by means of simple query functions, and also permits format-specific optimizations, for example, for regular grids. We show that this approach is comparable in perfor-

mance to the fastest previous algorithm, but applicable to significantly larger data sets.

6.1 Foundations of Discrete Morse Theory

While the notions of discrete Morse theory have intuitive correlation to the notions of "smooth" Morse theory, similar results can be arrived at completely independently. This suggests that while discrete Morse theory is a parallel theory, it is not necessary to relate all aspects to the smooth setting. The following definitions are mainly due to Forman [18].

Definition 6.1.1. (Discrete Function) A **discrete function** $f : K \to \mathbb{R}$ is a function that assigns a scalar value to every cell in K.

Definition 6.1.2. (Discrete Morse Function) A function $f : K \to \mathbb{R}$ is a *discrete Morse function* if for every $\alpha^{(d)} \in K$, its number of co-facets

$$\#\{\beta^{(d+1)} > \alpha | f(\beta) \leq f(\alpha)\} \leq 1,$$

and its number of facets

$$\#\{\gamma^{(d-1)} < \alpha | f(\gamma) \geq f(\alpha)\} \leq 1.$$

A discrete Morse function assigns a single scalar value to every cell of K. Just as there was a way to determine the criticality of a point by examining its neighborhood in the smooth theory, the values of facets and co-facets of a cell determine whether or not that cell is critical.

Definition 6.1.3. (Critical cell) A d-cell $\alpha^{(d)}$ is **critical** if its number of co-facets

$$\#\{\beta^{(d+1)} > \alpha | f(\beta) \leq f(\alpha)\} = 0,$$

and its number of facets

$$\#\{\gamma^{(d-1)} < \alpha | f(\gamma) \geq f(\alpha)\} = 0.$$

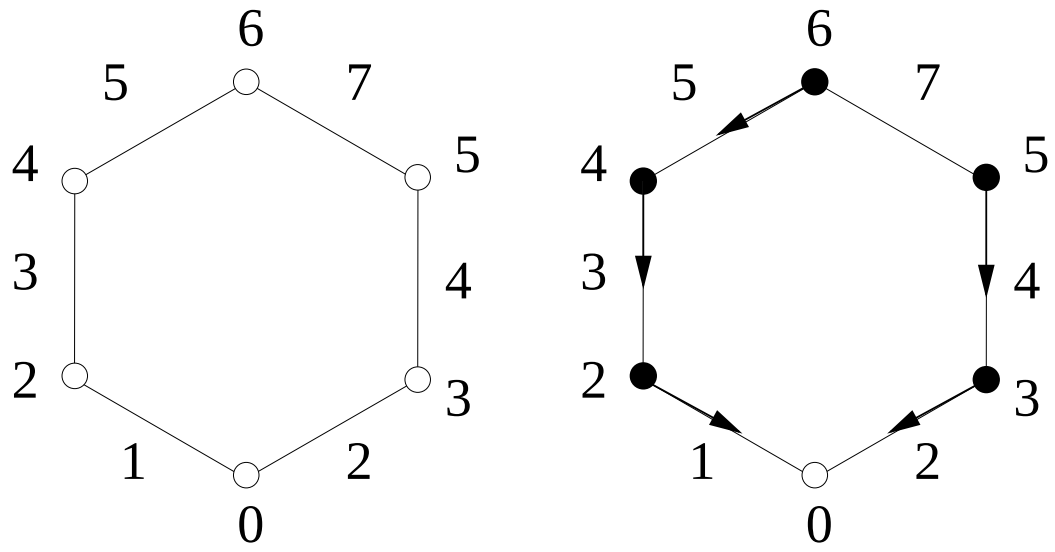


Figure 6.1.1: A discrete Morse function (left) assigned to the cells of a simple circle. The associated discrete gradient field (right) is a pairing of the vertices and edges. Note that the critical cells of the discrete Morse function correspond exactly to the unpaired cells of the discrete gradient field. In both cases, the vertex $f^{-1}(0)$ and the edge $f^{-1}(7)$ are the minimum and maximum, respectively.

Lemma 6.1.4. *Let $f : K \to \mathbb{R}$ be a discrete Morse function, then cell $\alpha \in K$ cannot fail both conditions of criticality simultaneously, i.e.,*

$$\#\{\gamma^{(d-1)} < \alpha | f(\gamma) \geq f(\alpha)\} \ + \ \#\{\beta^{(d+1)} > \alpha | f(\beta) \leq f(\alpha)\} \ \leq 1.$$

Alternately, we can say that its number of co-facets

$$\#\{\beta^{(d+1)} > \alpha | f(\beta) \leq f(\alpha)\} = 0,$$

or its number of faces

$$\#\{\gamma^{(d-1)} < \alpha | f(\gamma) \geq f(\alpha)\} = 0.$$

This result is very important in defining a notion of "flow" in the discrete setting. It is used in defining gradient arrows.

Figure 6.1.1 shows an example of a discrete Morse function and its associated critical cells. One of the fundamental results of discrete Morse theory relates homotopy equivalence between a

CW-complex and a minimal CW-decomposition by means of the critical cells.

Theorem 6.1.5. *Let $f : K \to \mathbb{R}$ be a discrete Morse function on a CW-complex K. Then K is homotopy equivalent to a CW-complex with exactly one cell of dimension d for each critical cell of dimension d.*

The continuous version of this theorem relates the critical points to the handlebody decomposition of a topological space (theorem 3.2.4). In fact, attaching a critical cell to the CW-complex is analogous to attaching a handle associated with a critical point in the smooth case.

In practice, constructing a discrete Morse functions from sampled values at vertices with the desired flow structure is very difficult. For example, assigning numbers to cells such that the axioms of a discrete Morse function are satisfied is very challenging. Instead, it is much easier to directly work with the discrete gradient, which leads to the following definitions:

Definition 6.1.6. (Discrete Vector) A **vector** in the discrete sense is a pair of cells $\{\alpha^{(d)} < \beta^{(d+1)}\}$, where we say that an arrow points from $\alpha^{(d)}$ to $\beta^{(d+1)}$. We call $\alpha^{(d)}$ the **tail** of the arrow, and $\beta^{(d+1)}$ the **head** of the arrow.

Intuitively, this simulates a direction of flow.

Definition 6.1.7. (Discrete Vector Field) A **discrete vector field** V on K is a collection of pairs $\{\alpha^{(d)} < \beta^{(d+1)}\}$ of cells of K such that each cell is in at most one pair of V.

The notion of a gradient involves the concept of monotonic flow, and each arrow in our a discrete gradient field points "down." Therefore, $\{\alpha^{(d)} < \beta^{(d+1)}\}$ is an arrow of the discrete gradient when $f(\alpha) \geq f(\beta)$. Note that this violates the second condition of criticality (definition 6.1.3) for α, and the first condition for β, so any cell paired in a gradient arrow is not critical. The properties of a discrete Morse function on a CW-complex (Lemma 6.1.4) ensure that each non-critical cell has exactly one pairing where it is either the head or tail of an arrow. Figure 6.1.1 shows how the arrows convey the concept of flow.

Definition 6.1.8. (Gradient Arrow, Gradient Vector) A **gradient arrow**, or alternatively, a **gradient vector**, in a discrete Morse function f on a CW-complex K, is a pair of cells (α, β), $\alpha, \beta \in K$,

where α is the unique facet of β, for which $f(\alpha) \geq f(\beta)$.

Note that according to this definition, gradient arrows point downward, in the direction of "steepest descent". In a smooth vector field, integral curves are obtained by tracing a path along the vector field. Similarly, in the discrete setting, a V-path follows the discrete vector field.

Definition 6.1.9. **(V-Path)** Given a discrete vector field V on K, a *V-path* is a sequence of cells

$$\alpha_0^{(d)}, \beta_0^{(d+1)}, \alpha_1^{(d)}, \beta_1^{(d+1)}, \alpha_2^{(d)}, \ldots, \beta_r^{(d+1)}, \alpha_{r+1}^{(d)}$$

such that for each i = 0,..., r, the pair $\{\alpha^{(d)} < \beta^{(d+1)}\} \in V$, and $\{\beta_i^{(d+1)} > \alpha_{i+1}^{(d)} \neq \alpha_i^{(d)}\}$.

The V-paths in a discrete gradient field are monotonic and do not contain loops. The following theorems describe the behavior of V-paths in a discrete gradient field and provide sufficient conditions for determining whether or not a discrete vector field is a discrete gradient field.

Theorem 6.1.10. *Let V be the discrete gradient field of a discrete Morse function f. Then a sequence of simplices*

$$\alpha_0^{(d)}, \beta_0^{(d+1)}, \alpha_1^{(d)}, \beta_1^{(d+1)}, \alpha_2^{(d)}, \ldots, \beta_r^{(d+1)}, \alpha_{r+1}^{(d)}$$

is a V-path if and only if $\alpha_i^{(d)} < \beta_i^{(d+1)} > \alpha_{i+1}^{(d)}$ for each i =0, 1, ... , r and

$$f(\alpha_0) \geq f(\beta_0) > f(\alpha_1) \geq f(\beta_1) > \ldots \geq f(\beta_r) > f(\alpha_{r+1}).$$

The monotonicity condition ensures that there can be no circular V-paths.

Theorem 6.1.11. *Let V be a discrete vector field. V is the discrete gradient field of some discrete Morse function f if and only if there are no non-trivial closed V-paths.*

A result of this theorem is that one can generate a discrete gradient field without knowing function values at cells, by creating discrete vectors such that every cell is in at most one vector and there are no non-trivial closed V-paths. In fact, there is an infinite number of discrete Morse functions that can be fitted to a discrete gradient field.

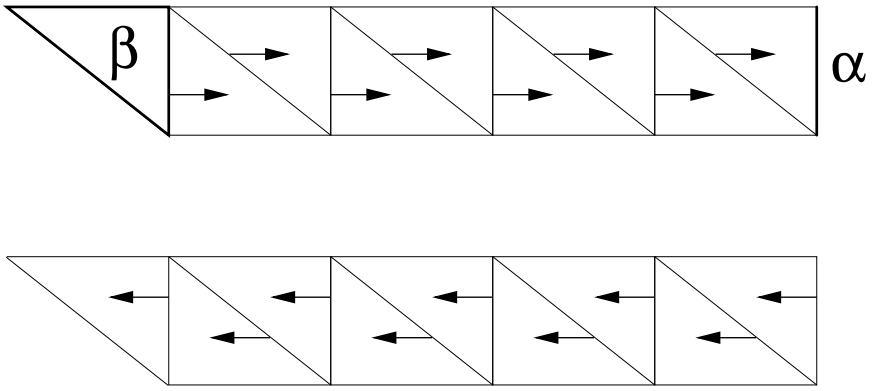


Figure 6.1.2: The gradient path from the boundary of β to α is reversed, canceling both critical cells.

We cancel pairs of critical points of a discrete gradient field by simply reversing the V-path. The following is the discrete analog of the cancelling handles theorem (theorem 3.8.2).

Theorem 6.1.12. ***(Discrete Canceling Handles)*** *Suppose $f : \mathbb{M} \to \mathbb{R}$ is a discrete Morse function such that $\alpha^{(p)}$ and $\beta^{(p+1)}$ are critical, and there is exactly one gradient path from the boundary of β to α. Then there is another Morse function $g : \mathbb{M} \to \mathbb{R}$ with the same critical cell except β and α are no longer critical. Furthermore, the gradient vector field of f is the same as the gradient vector field of g except along the unique path from the boundary of β to α.*

In a discrete gradient field, unpaired cells are critical. By only modifying the unique gradient path from β to α, we ensure that every other critical cell remains unpaired, and hence critical. Figure 6.1.2 illustrates how the gradient path is reversed to cancel two critical cells. Finally, we present a corollary that states that the discrete gradient of a function remains the discrete gradient of a function after the cancellation of a pair of critical cells.

Corollary 6.1.13. *The new vector field produced by cancellation of a pair of critical cells that are connected by a single path in a discrete gradient is itself a discrete vector field.*

The proof of this corollary relies on the fact that a pair of critical cells can only be cancelled if they are connected by a single path. If the cancellation were to introduce any non-trivial closed paths into the vector field, then before the reversal of the path, the closed path would have corresponded

to a split and merge of a V-path, and there would have been more than one path connecting the critical cells.

6.1.1 Discrete Morse Complex

In the smooth version of Morse theory, the cells of the Morse complex (definition 3.6.3) consisted of equivalence classes where two points are related when the integral lines passing through them shared a common destination. One can parametrize the *flow* at any point by injecting a massless particle and tracing its motion over time. Therefore, the flow $F_p(t)$ of a particle p evaluated at some time t gives the position of p after it has moved along the gradient vector field. The same intuition is used to define the flow in the discrete setting, with some crucial changes. First, the flow can only be evaluated at discrete intervals; in fact, we will be interested in the "flow" of a cell after taking a single gradient step. Second, in the discrete setting, V-paths are the equivalent of integral lines, and unlike integral lines, V-paths can merge or split. As a result, flow to or from a cell can merge or split, and the flow of a cell might be an ordered set of cells. The following definitions are necessary to formalize the notion of flow in the discrete setting:

Definition 6.1.14. (Chain) A **chain** is an abelian group: a set with an operator $+$ such that $a + b = b + a$. A chain of cells is denoted c_i, where i is the dimension of the cells.

Definition 6.1.15. (Orientation) The **orientation** of a cell is an order assigned to its vertices. The set of even permutations to this order are considered "positive", in the sense that they give the same orientation. The set of odd permutations are considered "negative", in that they give the opposite orientation.

Definition 6.1.16. (Induced Orientation) The faces of a cell α have an **orientation induced** by α, the vertices of the cell are a subset of those of α with the same ordering.

We define an inner product on cells that takes into account their orientation.

Definition 6.1.17. (Inner Product, $\langle c_i, c_j \rangle$) The **inner product** $\langle c_i, c_j \rangle$ of chains c_i and c_j in a

regular CW-complex is equal to

$$\sum_{\alpha \in c_i} \sum_{\beta \in c_j} \alpha \cdot \beta$$

where $\alpha \cdot \beta$ is one if $\alpha = \beta$ and the orientation of α is the same as the orientation of β, minus one if the orientations do not agree, and zero if $\alpha \neq \beta$.

Definition 6.1.18. (Boundary Operator, ∂) The **boundary operator** ∂ maps a cell to its facets, with the induced orientation on the facets. Formally, $\partial\alpha = \sum_{\beta<\alpha} \langle \partial\alpha, \beta \rangle \, \beta$.

Therefore, the boundary operator returns the facets of a cell α, along with a sign indicating whether or not the facet's own orientation in the underlying mesh agrees with the orientation induced on it by α. For all regular CW-complexes, the inner product of a cell with the faces of one of its co-faces $\langle \partial\alpha, \beta \rangle = \pm 1$. Next, we define an operator that simulates taking one step in the direction implied by the discrete gradient.

Definition 6.1.19. (Discrete Tangent Operator, $V(\beta)$) The **discrete tangent operator**, $V(\beta)$ maps the cell at the tail of a gradient arrow to the head of the gradient arrow. The discrete tangent operator assigns an orientation to the head under which the induced orientation of the tail is opposite to the tail's own orientation. Formally, $V(\beta) = -\langle \partial\beta, \alpha \rangle \, \alpha$, where β is the tail of a gradient arrow, and α is the head of a gradient arrow. If β is not the tail of a gradient arrow, then $V(\beta) = 0$.

Now we can define the discrete flow operator. This operator simulates taking a step in the negative gradient direction, also accounting for the fact that V-paths can split ant merge.

Definition 6.1.20. (Flow Operator, $\Phi(\alpha)$) The **flow operator** Φ is defined by

$$\Phi(\alpha) = \alpha + \partial V(\alpha) + V(\partial\alpha)$$

Figure 6.1.3 illustrates the flow operator applied to an edge of a mesh with a discrete gradient field defined on it.

The result of this operator is a chain of edges. Using these definitions, we will define the discrete Morse complex. In the smooth version of Morse theory, descending manifolds are "generated" at

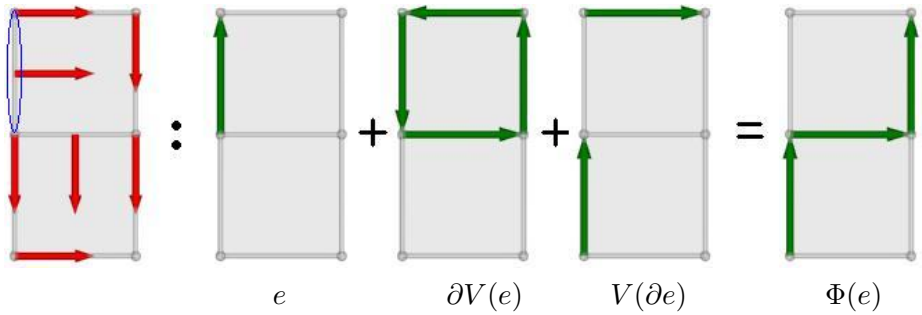


Figure 6.1.3: The flow operator is determined by the discrete gradient arrows (red) and the connectivity of the mesh. It applied to the circled edge e: $\Phi(e) = e + \partial V(e) + V(\partial e)$. An orientation (green arrow) is chosen for e, and it induces an opposite orientation on the 2-cell that is the head of the gradient arrow starting at e. It also induces opposite orientations on the edges that are the head of gradient arrows beginning at the facets of e. Cells summed with opposite orientations cancel one another out. The result of the flow on e is three edges in a chain.

critical points, by the set of integral lines that terminate at that critical point. The flow operator applied to any point in a stable manifold maps that point to another in the descending manifold. In fact, one can think of the critical point as the source of the flow that covers the entire descending manifold, and the entire manifold maps to itself under the flow operator. The same concept is used in discrete Morse theory to define the Morse complex.

Definition 6.1.21. (Φ-Invariant Chain) A chain c is **Φ-invariant** when $\Phi(c) = c$.

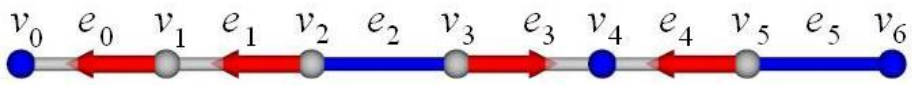
Definition 6.1.22. (Discrete Morse Complex) Let K be an oriented CW-complex with a discrete gradient vector field V. Let $C_p^\Phi(K, Z)$ denote the Φ-invariant p-chains of K. The boundary operator ∂ maps $C_p^\Phi(K, Z)$ to $C_{p-1}^\Phi(K, Z)$. The differential complex defined by

$$\Gamma^\Phi : 0 \to C_n^\Phi(K, Z) \xrightarrow{\partial} C_{n-1}^\Phi(K, Z) \xrightarrow{\partial} \cdots \xrightarrow{\partial} C_0^\Phi(K, Z) \to 0$$

is the **discrete Morse complex**.

The following is an alternate but equivalent definition of the discrete Morse complex.

Definition 6.1.23. (Discrete Morse Complex) Let K be a CW-complex with a discrete gradient vector field, V. The descending d-manifold of a critical d-cell α is the set of d-cells that are part of



$$\begin{aligned}
\Phi(v_0) &= v_0 + 0 + 0 = v_0 \\
\Phi(v_1) &= v_1 - v_1 + v_0 = v_0 \\
\Phi(v_2) &= v_2 - v_2 + v_1 = v_1 \\
\Phi(v_3) &= v_3 - v_3 + v_4 = v_4 \\
\Phi(v_4) &= v_4 + 0 + 0 = v_4 \\
\Phi(v_5) &= v_5 - v_5 + v_4 = v_4 \\
\Phi(v_6) &= v_6 + 0 + 0 = v_6
\end{aligned}$$

$$\begin{aligned}
\Phi(e_0) &= e_0 + 0 - e_0 = 0 \\
\Phi(e_1) &= e_1 - e_1 + e_0 = e_0 \\
\Phi(e_2) &= e_2 + e_1 + e_3 \\
\Phi(e_3) &= e_3 + 0 - e_3 = 0 \\
\Phi(e_4) &= e_4 + 0 - e_4 = 0 \\
\Phi(e_5) &= e_5 + e_4
\end{aligned}$$

Figure 6.1.4: The discrete gradient (red arrows) is defined on a mesh. We compute the flow of every cell. The Φ-invariant chains are $\{v_0\}$, $\{v_4\}$, $\{v_6\}$, $\{e_0, e_1, e_2, e_3\}$, and $\{e_4, e_5\}$. These are the 0- and 1-manifolds of the discrete Morse complex. Note that they are also the 0- and 1-cells of V-paths originating at the critical (blue) cells.

V-paths starting on the boundary of α. The **discrete Morse complex** is the complex formed by the descending manifolds.

This definition, although less formal, gives the same descending manifolds as definition 6.1.22, and is far more intuitive. Figure 6.1.4 illustrates the computation of both the invariant chains and the descending V-paths.

The descending manifold associated with a critical cell can be defined through this second definition of the discrete Morse complex.

Definition 6.1.24. (Descending Manifold) Let K be a CW-complex with a discrete gradient vector field V. The **descending manifold** associated with a critical d-cell α is the set of d-cells that belong to V-paths that start on the boundary of α.

The advantage of using these definitions is that it is easy to compute the homology of a CW-complex, as done by Forman [18]. For analysis, however, we ideally want to associate a descending manifold with every cell of the CW-complex, and these definitions skip the "in between" cells with dimension lower than the critical cell that generates the descending manifold. We resolve this by introducing *extended descending manifolds*.

Definition 6.1.25. (Extended Descending Manifold) Let K be a CW-complex with a discrete

gradient vector field V. The discrete Morse complex is formed by the complex of descending manifolds. Let α be a critical d-cell. We say that the **extended descending manifold** associated with α is the closure of the descending manifold of α, D_α, plus the extended descending manifolds of critical cells in the boundary of D_α.

This is a recusive definition that "fills in" the in-between cells of V-paths in such a way that every cell of K is assigned one or more descending manifolds. For example, the extended descending manifolds associated with e_2, e_5, v_0, v_4, and v_6 in figure 6.1.4 are $\{v_0, e_0, v_1, e_1, v_2, e_2, v_3, e_3, v_4\}$, $\{v_4, e_4, v_5, e_5, v_6\}$, $\{v_0\}$,$\{v_4\}$, and $\{v_6\}$, respectively.

Formally, discrete Morse theory only defines descending manifolds. Instead of taking an "inverse" flow on the dual of a CW-complex, we define ascending manifolds through modification of the second definition of the discrete Morse complex 6.1.23.

Definition 6.1.26. (Ascending Manifold) Let K be a CW-complex with a discrete gradient vector field V. The **ascending manifold** associated with a critical d-cell α is the set of d-cells that belong to V-paths that end on the co-boundary of α.

This definition ascending manifolds is symmetric to the definition of descending manifolds. Note that the discrete Morse complex was the complex of descending manifolds. For clarity, we will call this the *descending Morse complex*, and introduce the *ascending Morse complex*. We can define the extended ascending manifolds in a similar manner.

Definition 6.1.27. (Discrete Ascending Morse Complex) Let K be a CW-complex with a discrete gradient vector field, V. The ascending d-manifold of a critical d-cell α is the set of d-cells that are part of V-paths ending on the co-boundary of α. The **discrete ascending Morse complex** is the complex formed by the ascending manifolds.

Again, we refine this definition to include the in-between cells.

Definition 6.1.28. (Extended Ascending Manifold) Let K be a CW-complex with a discrete gradient vector field V. The discrete ascending Morse complex is formed by the complex of ascending manifolds. Let α be a critical d-cell. We say that the **extended ascending manifold** associated with

α is the co-closure of the ascending manifold of α, A_α, plus the extended ascending manifolds of critical cells in the boundary of A_α.

Now we are finally prepared to define the discrete Morse-Smale complex.

Definition 6.1.29. (Discrete Morse-Smale Complex) Let K be a CW-complex with a discrete gradient vector field V. The cell complex formed by the pairwise intersection of each extended ascending manifold and extended descending manifold is the **discrete Morse-Smale complex**.

6.2 Characterization of Cancellations for n-dimensional Morse-Smale Complexes

A function f is simplified by repeated cancellation of pairs of critical points. The local change in the Morse-Smale complex indicates the smoothening of the gradient vector field and hence of the function f. The ordering of critical point pairs is defined by *persistence*, which quantifies the importance of the topological feature associated with a pair. The *persistence* of a critical point pair is the absolute difference in value of f between the two points. We use the ordering given by persistence to reduce the number of critical points and hence remove topological features from f.

A cancellation operation is *valid* (*i.e.*, it can be realized by a local perturbation of the gradient vector field) on a pair of critical points if and only if there is exactly one arc connecting them in the complex. Therefore, the indices of the two critical points must differ by one. Also, any critical point pair that is connected by multiple arcs represents a configuration known as a *strangulation* or a *pouch*, for which there is no direct perturbation of the gradient that removes the critical point pair. We characterize the cancellation operation for Morse-Smale complexes of any dimensions in terms of a change in the combinatorial structure of the complex. The geometric change in the manifolds of critical points can be derived from this combinatorial change.

Cancellation: Let Γ be an Morse-Smale complex for a scalar function defined on a closed d-manifold $\mathbb{M}$. Let l and u be the lower and upper nodes of an arc a in Γ, with index i and $i+1$ respectively. Let A_l be the set of arcs that have l as one end point, A_u the set of arcs that have

u as one end point, N_l the set of nodes in the neighborhood of l, and N_u the set of nodes in the neighborhood of u.

The *combinatorial cancellation* of (l, u) changes the combinatorial structure of the Morse-Smale complex and is characterized as follows:

1. Create a new arc connecting every critical point of index $i + 1$ in N_l to every critical point of index i in N_u, and add them to Γ.

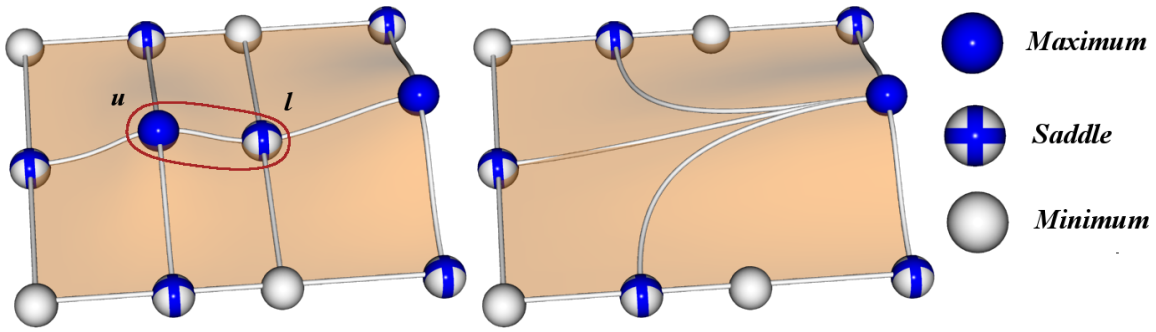
2. All arcs in A_l, A_u are removed from the complex, and l and u are also removed from the complex.

This operation changes the 1-skeleton of the Morse-Smale complex, however, it also represents a change in the embedding. This change can be derived from the combinatorial cancellation, and one simple way to maintain a valid embedding is to characterize it as a merging of the manifolds of the nodes involved in the cancellation. We call the change in the embedding the *geometric realization* of the cancellation and it is characterized as follows:

1. For every node of index $i + 1$ in N_l, merge its descending manifold with the descending manifold of u.

2. For every node of index i in N_u, merge its ascending manifold with the ascending manifold of l.

The cells of Γ after the cancellation are only different where there are new intersections of the changed ascending and descending manifolds. These intersections are represented by the new arcs in the combinatorial structure of the Morse-Smale complex. Although there are potentially $|N_l| \times |N_u|$ new arcs and cells created in the complex, the number of nodes in the complex is reduced by two, and eventually all the new arcs are also removed in saddle-extremum cancellations. Figure 6.2.1 illustrates this cancellation operation in the case of two-dimensional complexes.



(a) (b)

Figure 6.2.1: The circled arc connects a saddle l to a maximum u (a). Cancellation of (l,u) removes all arcs attached to l or u, and creates new arcs from the lower neighbors of u to the upper neighbors of l (b). In the two-dimensional case, this connects all the saddles neighboring u to the maximum neighboring l, in effect, merging l and u with the maximum).

6.3 Related Methods in Discrete Morse Theory

In our approach, we utilize discrete Morse theory as presented by Forman [18, 19]. Lewiner et al. [30, 31] showed how a discrete gradient field can be constructed and used to identify the Morse-Smale complex. However, complications arise in this method due to the fact that the gradient is constructed by creating acyclic hypergraphs. This is resolved using Union-Find, however, this kind of approach requires a hypergraph representation of the gradient, limiting its applicability to small data. Furthermore, the function value assigned to each cell is the average of its vertices, which does not necessarily define a discrete Morse function, and this leads to complications in ensuring the correct number and location of critical cells. Resolving this can even require modification of the input mesh. King et al. [28] presented a method for constructing a discrete gradient field that agrees with the large-scale flow behavior of the data defined at vertices of the input mesh, but again, this requires an explicit representation of the Hasse diagram. These discrete Morse theory based algorithms for constructing the discrete Morse-Smale complex have a critical shortcoming: they require processing of the entire dataset and a representation of the complex at the finest level of detail before any simplification can be done. In practice, this imposes limits on both the size, and the complexity of the data that can be handled.

6.4 Algorithmic Kernel

The kernel of our approach produces a discrete gradient field from a set of scalar samples at vertices of a CW-complex. The algorithm first assigns a temporary value to every cell in the complex, such that every cell is initially critical. This temporary function is called the discrete augmented function, and is a discrete Morse function. Then, in a number of sweeps through the complex, adjacent critical cells are cancelled in a way that pairs them in gradient arrows. This cancellation order is performed in a manner that produces a gradient vector field that agrees with the scalar flow restricted to edges of the complex. Any cells that are not paired by this algorithm are nodes of the Morse-Smale complex. The discrete gradient is used to compute the ascending and descending manifolds of the nodes, for which transversal intersection is a guaranteed property. Any V-path connecting a pair of nodes is an arc in the Morse-Smale complex.

6.4.1 Discrete Augmented Function

Our algorithm requires the sorting of all the cells of a particular dimension, and we assign function values to every cell. We use the *discrete augmented function* F, where each cell has a function value slightly larger than the highest value of its faces. This is similar to the augmented function presented in section 4.4, which assigns nodes of the artificial complex function values.

Definition 6.4.1. Discrete Augmented Function The **discrete augmented function** assigns a scalar value to every cell in a manner that its value is slightly larger than the value of its faces:

$$F(\alpha) = MAX\{\sigma : \sigma < \alpha\} + \epsilon.$$

In this manner, every cell is a critical cell in the discrete sense, and the formation of pairs performed by the algorithmic kernel corresponds to an ϵ-persistence cancellation of adjacent critical cells, creating an arrow between them. We use symbolic perturbation to achieve a global ordering and simulate differentiability.

In fact, this epsilon does not have to be defined. Instead, we implement the comparison of two

cells to take into account their dimension when their function values are the same, which simulates the addition of a positive epsilon.

Algorithm 6.4.2. GREATERTHAN(**cellID** a**, cellID** b)

Input: the identifiers of two cells a and b

Output: *true* if $a > b$, *false* otherwise

```
1: if F(a) == F(b) then
2:     if dim(a) == dim(b) then
3:         return a > b
4:     else
5:         return dim(a) > dim(b)
6:     end if
7: else
8:     return F(a) > F(b)
9: end if
```

6.4.2 Algorithm

This comparison operator works as follows: if two cells have the same scalar value, the dimension of the cells is compared. If their dimensions are also the same, then the identifier in the data structure is used to select which one is higher. In this manner, we can enforce a strictly increasing ordering over all the cells in a CW-complex.

6.4.3 Generating Gradient Arrows

Given a regular CW-complex K with scalar values defined for all cells by the discrete augmented function, we compute the discrete gradient by assigning gradient arrows in a greedy manner in ordered sweeps over cells of K. The sweeps are performed in order of increasing dimension, and from lowest to highest, simulating a "flood fill" of basins of the function.

Algorithm 6.4.3. COMPUTEGRADIENT(**complex K**)

Input: a CW-complex K with and ordering on all its cells

Output: K with gradient arrows assigned

```
1:  for i ∈ [0, .., d] do
2:      Iter = K →sortedCellIterator(i)
3:      while Iter→hasNext() do
4:          α = Iter→Item()
5:          if ! K →isMarked(α) then
6:              if K →hasPairableCoFacet(α) then
7:                  β = K →lowestPairableCoFacet(α)
8:                  K →pair(α, β )
9:                  K →mark(α); K →mark(β)
10:             else
11:                 K →setCritical(α)
12:                 K →mark(α)
13:             end if
14:         end if
15:     end while
16: end for
```

This algorithm iterates through all cells in increasing order of dimension and function value, assigning gradient pairs in a greedy manner. The $sortedCellIterator(i)$ iterates through the i-cells of K in order of increasing function value. The lowest cell of dimension i that has not been paired or set as critical is α, and its co-facets are searched for a possible pair. The function $hasPairableCoFacet(\alpha)$ returns true if there is a co-facet with exactly one facet that is not marked. If there are multiple such co-facets $lowestPairableCoFacet(\alpha)$ will return the lowest one, with ties being broken in the direction that would create the path with steepest descent. While any pairable co-facet could be paired with the cell, we select the one in the direction of steepest descent to represent gradient flow of the function. When $pair(\alpha, \beta)$ forms a gradient arrow, the cell

α is set as the tail and its co-facet β is marked as the head.

The discrete vector field that is produced is a discrete gradient field, since each cell is paired exactly once, and no loops can be created in V-paths, since each V-path has a critical cell as a source that cannot be further paired.

The flow across cells in a discrete Morse function does not necessarily correspond to scalar flow. However, we assign function values to all cells in a manner that allows gradient arrows to agree with the scalar flow. The particular sorting order of the cells of a dimension given by sorted cell iterator (line 2) and the lowest facet selected for pairing (line 7) determine the shape of the discrete gradient flow. If the sorting is done from lowest to highest, and the facet selected is in the direction of steepest descent, the discrete gradient flow generated will mostly agree with the scalar piecewise linear gradient.

6.4.4 Computing the Discrete Morse-Smale Complex

The Morse-Smale complex of a discrete gradient vector field is uniquely determined by the gradient paths. The definition of a discrete Morse-Smale complex identifies every cell with the ascending manifolds that can be found by tracing gradient paths downwards, and the descending manifolds that can be found by tracing gradient paths upwards. To compute a Morse-Smale complex from the discrete gradient vector field produced by the algorithm kernel, we find the critical cells, and compute the ascending and descending manifolds by following the gradient paths. All the cells of a path whose origin is a critical cell α belong to the ascending manifold of α. Symmetrically, all the cells of a path whose destination is a critical cell β belong to the descending manifold of β. These paths are computed using a breadth-first search through the discrete gradient field. The cells of the Morse-Smale complex are attained as the intersection of ascending and descending manifolds. The nodes and arcs forming the combinatorial structure of the complex are the critical cells and the gradient paths connecting the critical cells.

Computing the ascending and descending manifolds requires a complete traversal of the gradient paths. Our algorithm computes all cells of the Morse-Smale complex. However, many cases arise where analysis of the data only requires the combinatorial structure (1-skeleton) of the Morse-Smale

complex. In this case, we can perform a more efficient computation by only tracing ascending manifolds, and restricting the traversal to V-paths, not the full flow. Nevertheless, computing the ascending manifolds of all minima requires a complete traversal of the entire gradient field. If we are only interested in the combinatorial structure of the complex, this traversal is not necessary, and we compute 1-saddle-minima connections by tracing gradient paths downwards from the 1-saddles. It is guaranteed that there are exactly two paths that terminate at a minimum for each 1-saddle, and the paths cannot split. This makes the computation of 1-saddle minimum connections efficient.

In general, ascending and descending manifolds can merge in a discrete gradient field. We maintain the Morse-Smale complex by simulating a separation between ascending manifolds and descending manifolds. Note that one major advantage of using discrete Morse theory is that special rules for identifying manifolds of different dimensions are not required, as was the case in the algorithm presented in [26].

6.5 A Divide-and-Conquer Approach

In this section, we compute the Morse-Smale complex with a divide-and-conquer approach. Figure 6.5.1 shows an overview of the algorithm. The divide-and-conquer approach divides the dataset into parcels where the discrete gradient and Morse-Smale complex are computed locally on the boundary and in the interior. The boundary flow is fixed such that any flow passing through the boundary must pass through critical points restricted to the boundary. When two parcels are merged, the gradient and Morse-Smale complex on the new interior is updated. In particular, the steps in this process are the following: the dataset is split into parcels, and cells are classified as interior, boundary, or exterior; the discrete gradient is computed on the boundary and interior of each parcel, and then an Morse-Smale complex is computed on the interior of each parcel; the parcels are glued back together, merging the Morse-Smale complexes; and finally, the artifacts introduced by the merging are removed by cancellation of the ϵ-persistence pairs of the Morse-Smale complex. We describe each step in detail in the following.

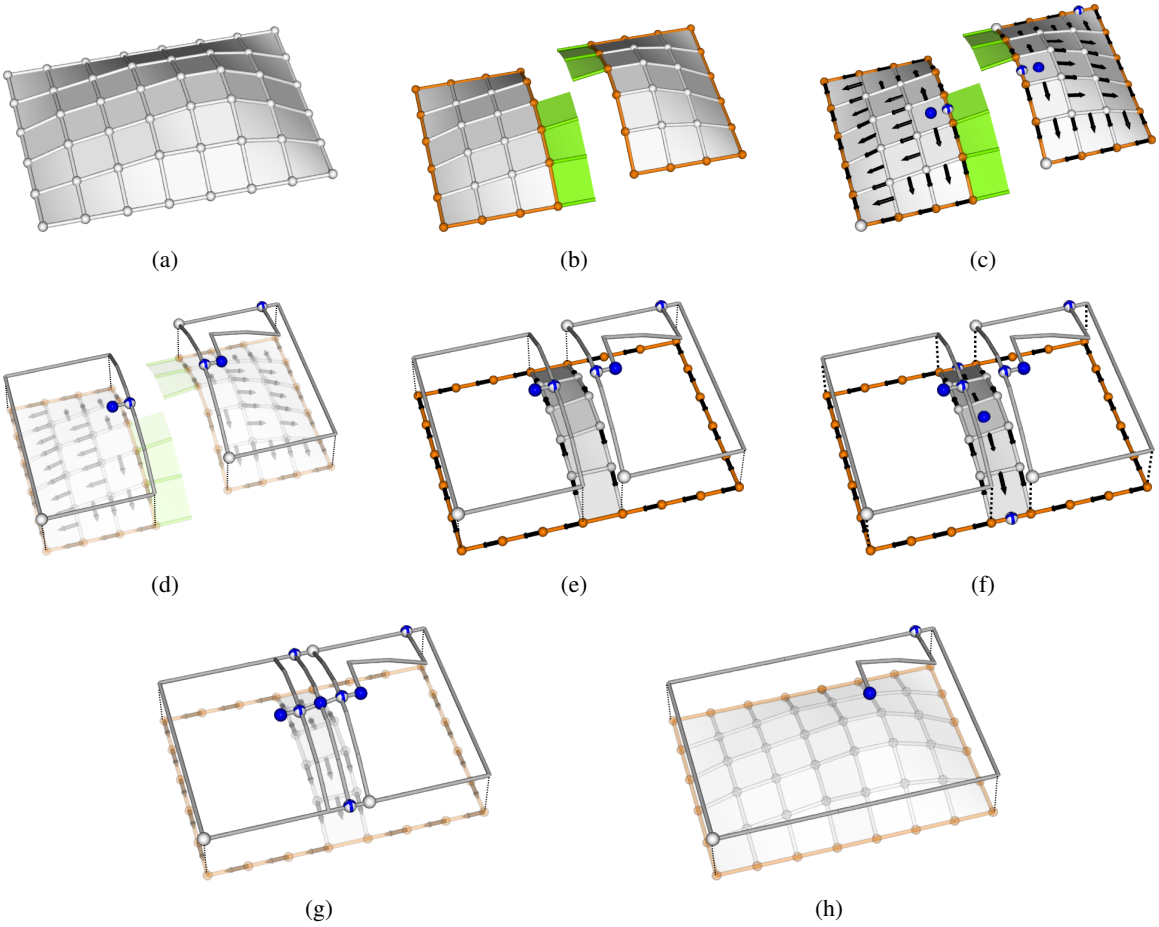


Figure 6.5.1: An overview of the algorithm. The dataset (a) is broken up (b) into parcels, and cells are classified as interior (grey), exterior (green), or boundary (orange). The discrete gradient is found on the boundary and interior of each parcel (c) and the Morse-Smale complex is computed (d). The parcels are merged, reclassifying boundary and exterior cells (e), and the discrete gradient is computed on the newly introduced boundary and interior cells (f). The complex is computed in these cells to complete the merging process (g). Simplification of ϵ-persistence pairs removes artifacts that have resulted from merging (h).

6.5.1 Splitting and Classifying

A dataset may be given in any number of formats. It is partitioned into *parcels* during a pre-processing stage. Each parcel is a set of cells with certain attributes, which will be discussed in section 6.6. Logically, a parcel P is a collection of spatially coherent cells which form a regular CW-complex, and is a subset of the input complex, $P \subseteq K$. The complex P has scalar values defined at its vertices.

The cells of a parcel P are categorized as interior, exterior, or boundary according to the following conditions:

1. A cell with one or more of its faces in $K - P$ is considered *exterior* to P.
2. A cell with a co-face in $K - P$ or exterior to P is considered *boundary*.
3. A cell in P with none of its faces or co-faces in $K - P$ or exterior is considered *interior* to P.

The boundary cells form a manifold without boundary of dimension less than d for any input mesh P, where the highest dimension of any cell in P is d. The steps of partitioning the data, and classifying interior, exterior, or boundary cells are illustrated in Figure 6.5.1 (a) and (b).

6.5.2 Gradient on a Parcel

Although the gradient computation kernel (algorithm 6.4.3) operates on any regular CW-complex, when computing the discrete gradient of a parcel we impose some restrictions. The boundary cells of a parcel represent the interface where flow can pass from one parcel to the next when the parcels are merged later on. To keep this interface as simple as possible, we first compute the discrete gradient on the boundary of a parcel and then on the interior. This re-ordering ensures that no boundary cell will be paired with an interior cell. The motivation behind this reordering is to restrict the number of places where flow can enter/exit the parcel from/to another parcel to the critical points on the boundary. This is an important property that will make merging parcels efficient in subsequent steps of the algorithm. The computation of a gradient is a step in the algorithm shown in Figure 6.5.1(c), where the gradient is first computed on each parcel, and then the gradient is computed in the interior.

6.5.3 The Morse-Smale Complex on a Parcel

We use the technique presented in section 6.4.4 to compute the Morse-Smale complex on a parcel. The result of computing the Morse-Smale complex on a parcel is shown in Figure 6.5.1(d).

6.5.4 Merging Parcels

Merging two parcels is a three-step process that involves gluing the two meshes together and updating and classification of cells, computing the discrete gradient on the new boundary and interior, and finally merging the two Morse-Smale complexes. Two meshes are glued together by updating the interior, boundary, or exterior classification on its cells. The interior cells remain interior, while the boundary cells can become interior cells or remain boundary, and the exterior cells can become interior or boundary or remain exterior. The same rules apply that were presented in section 6.5.1 for determining this classification. Figure 6.5.1 (e) shows the new classification of cells after this first step in the merging process.

We repeat the algorithmic kernel (algorithm 6.4.3) for finding the discrete gradient field on the merged parcels. First the gradient is computed for the cells that became boundary in the first step of the merging process, and then the gradient is computed for the cells that became interior in the first step of the merging process. Figure 6.5.1 (f) shows the discrete gradient computed on the new boundary and in the interior cells.

Finally, the Morse-Smale complexes of each parcel are merged. Due to the way we first computed the discrete gradient on the boundary and then in the interior in section 3.3, flow can only enter or leave a parcel through its boundary critical points. Therefore, we extend the Morse-Smale complex in each parcel by tracing gradient paths from all the newly classified interior and boundary critical cells. The only possible new connections in the merged Morse-Smale complex are between newly classified interior critical cells. This fact makes it possible to remove the discrete gradient on the interior of each parcel from memory prior to the merging process, allowing for the memory-efficiency of the divide-and-conquer approach. Figure 6.5.1 (g) shows how the complexes are merged by connecting them with critical cells in the new boundary and in the interior.

The mesh update operations for merging two chunks are handled by the representation for the data format. The classification of each cell in the interior (black and dark shaded) remains the same. The boundary cells (grey) now may become interior cells. The exterior cells (pink) may become interior or boundary. A new chunk is attained as the result of merging the two chunks.

Artifact Removal The merging of two parcels results in an Morse-Smale complex on the new parcel with extra nodes and arcs where the old boundaries were. These extra nodes and arcs have zero persistence, and are removed by canceling all 0-persistence pairs in the Morse-Smale complex of a parcel. Figure 6.5.1 (h) shows how the Morse-Smale complex in the interior is cleaned up by canceling 0-persistence arcs.

6.6 A Framework for Generality

The algorithm described in the previous section relies on queries that are supported by a wide range of data formats. The internal representation of a parcel can vary based on the needs or optimizations possible for any particular data format. For example, for regular data, the connectivity and classification of cells is attainable directly from their indices and the extents of the parcel, therefore the queries can be resolved in an efficient manner. A more general data format, such as an AMR grid, or simplicial complex, may require a more elaborate storage mechanism. The data format handles queries regarding characteristics of the cells in a parcel as well as the connectivity of the cells within that parcel.

6.6.1 Queries on the Parcel

To compute the discrete gradient, certain queries must be implemented for the data structure. Given a unique identifier id for a cell, our implementation supports these functions:

- dimension(id): returns the dimension of the cell
- isPaired(id): returns true if the cell has been paired
- isPairable(id): returns true if the cell has one unpaired facet
- greaterThan(id_1, id_2): returns true if id_1 has higher function value than id_2

A parcel must also be able to provide iterators that access cells and their neighbors, which are implemented by these functions:

- d-cellIterator(): returns an iterator over all d-dimensional cells in the parcel in sorted order
- boundary-d-cellIterator(): returns an iterator over all d-dimensional cells on the boundary of a parcel in sorted order
- neighborIterator(id): returns an iterator over the facets and co-facets of a cell

Finally, a parcel must also be able to change the state of some of its cells, which is enabled by the following functions:

- markCritical(id): marks the particular cell as a critical cell
- markAndPair(id_1, id_2): marks both as paired, and sets the pair of each to the other

To compute the Morse-Smale complex from the discrete gradient field, the following queries must be supported:

- criticalPointIterator(): returns an iterator over the critical points in a parcel
- getPair(id): returns the identifier of the cell that id is paired with

Finally, the queries necessary for visualization of the complex require that the geometric information of a cell can be recovered from its identifier:

- getGeometry(id): returns an array of vertices that are the 0-dimensional faces of id

6.6.2 Flow of Control

The basic steps of the algorithm discussed in section 3 are used to compute and merge the Morse-Smale complex on parcels, however, the order in which computation and merging take place are regulated by the *data manager*. The data manager is a data-format-specific module that organizes the flow of computation for maximal efficiency. In fact, each dataset may have its own data manager to optimize for any structural properties of the data format or feature queries in the function.

The details of dividing the data into parcels, ordering the computation of the gradient and Morse-Smale complex on parcels, ordering the merging of parcels, and any on-the-fly simplification of the

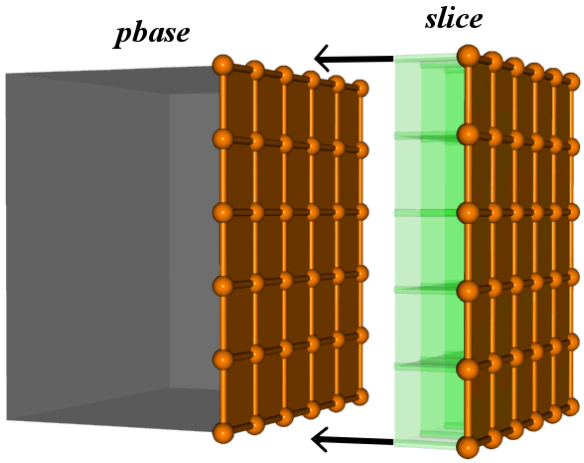


Figure 6.6.1: A new slice is created to be merged with $pbase$. The orange cells mark boundary, the green mark exterior, and the grey area indicates the processed interior of $pbase$. The discrete gradient on $pbase$ only needs to be stored on the orange "interface" slice.

Morse-Smale complex are left to the implementation. The interface with the algorithm is defined by the following functions:

1. computeGradAndComplex(Parcel p): computes the gradient and Morse-Smale complex on boundary of a parcel; the results are stored in the state of p. Additionally, the parcel is prepared for merging, by removing the interior cells from memory.

2. merge(Parcel $p1$, Parcel $p2$): combines $p1$ and $p2$ into $p1$, internally updating interior, boundary, and exterior classifications, computing the discrete gradient on the new boundary and interior cells, merging the Morse-Smale complexes, and performing an ϵ-persistence simplification. After merging, $p2$ is empty.

3. simplify(Parcel p, Filters f): performs simplification of the Morse-Smale complex on p according to filters defined in f.

A parcel may only be merged with another if the gradient and Morse-Smale complex have been computed on its boundary and in its interior, or if it composed entirely of exterior cells. It is the responsibility of the data manager to create parcels and load the data.

6.6.3 Case Study: Slices on a Grid

The implicit structure of data defined on a regular grid (with inherent indices i, j, and k) can be exploited to achieve an efficient implementation. For example, the dimension, geometric location, and neighbors of a cell can be derived from its index in the cases of simple Cartesian or uniformly spaced rectilinear grids. Also, rectangular parcels can be defined as lower and upper extents in each dimension. In this case, the extents determine whether a cell is interior, exterior, or boundary. Merging the meshes of two aligned parcels can be accomplished by simply modifying the extents. All these properties make it possible to devise highly efficient implementations of the queries on the parcel and its cells.

In the following, we consider the simple case of a dataset defined on a rectilinear grid with uniform spacing, aligned with the three coordinate axes. The simplest possible data manager creates a parcel for every slice along one axis of the data. For example, in a 3D uniform rectilinear axis-aligned grid of size $X \times Y \times Z$, the data manager creates an $X \times Y$ parcel for every z value. Although there are many ways to order the creation and merging of parcels, the simplest way is to accumulate them in a growing parcel along one axis. In this case the data manager would operate as follows:

1: Parcel $pbase$ = empty
2: **for** $z \in [0, .., Z-1]$ **do**
3: Parcel $slice$ = createXYSlice(z)
4: computeGradAndComplex($slice$)
5: merge($pbase$, $slice$)
6: simplify($pbase$, filters)
7: **end for**

Here the createXYSlice(z) reads the data from the input file corresponding the XY slice at z and initializes state variables in the slice. The slice that is created is an array of size $X \times Y \times 8$, required for storing the eight cells for every vertex of the data. The slices are merged with $pbase$, until $pbase$ contains the complex of the entire mesh. Figure 6.6.1 shows the addition of a slice onto

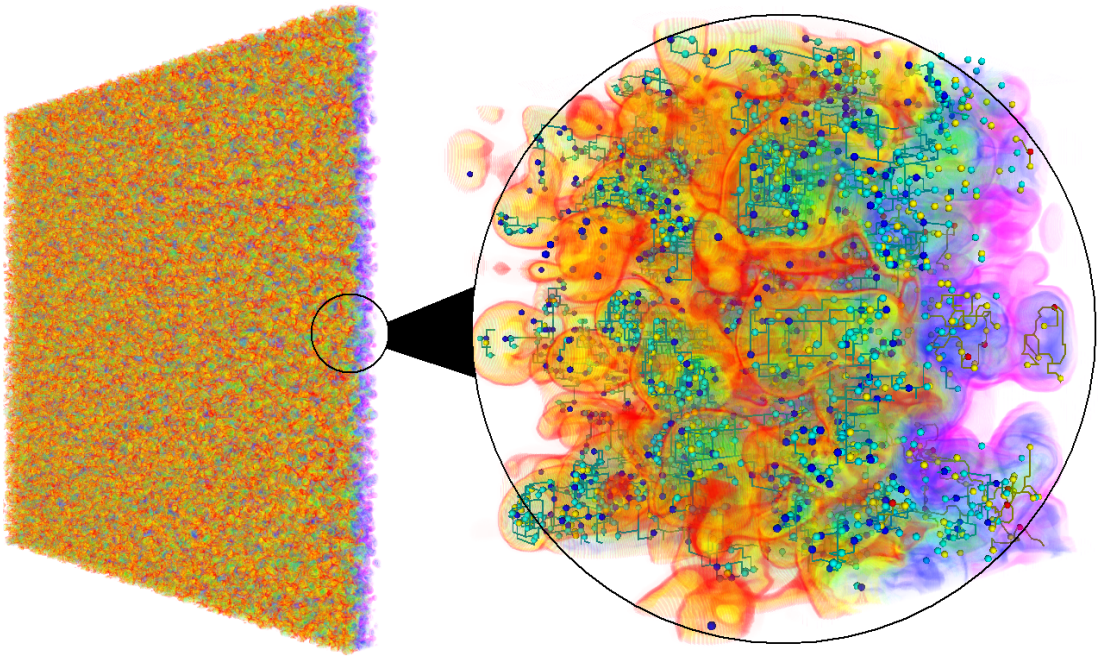
$pbase$. The simplification step controls the size of the complex as each slice is added on.

6.6.4 Results

We provide results for data given on a uniform rectilinear grid, using the slicing data manager. The results were generated on an off-the-shelf 2.21GHz AMD Athlon with 2.0Gb memory, the same hardware configuration used in [26]. We compare the performance of our algorithm to the previously fastest approach [26] in Table 6.1. Our run times are similar, however, while the approach in [26] has a dataset size limit of $256 \times 256 \times 256$, we also have produce results for one timestep of a simulation of a Raleigh-Taylor instability, which has a resolution of $1152 \times 1152 \times 1000$, shown in Figure 6.6.2. Laney et al. [29], used a parameter to select a 2-manifold level set, and the two-dimensional Morse-Smale complex used on the height map to identify bubbles in the turbulent mixing layer. A similar kind of analysis is possible using the full complex we computed of the three-dimensional data without the need for parameter selection, nevertheless, performing this kind of in-depth analysis is beyond the scope of this study. Our run times are larger for the small data set sizes because we simplify ϵ-persistence arcs on the fly, incorporating some of the post-processing into the construction time. The results of [26] do not include this extra time needed for simplification, and the complex extracted by that method must be simplified to remove noise and attain an Morse-Smale complex comparable to the result of our approach. Nevertheless, the optimizations made possible by our general framework make the run times similar. The size of the memory footprint was controlled by the on-the-fly simplification, with the overhead for storing parcels being 42Mb, and the size of the complex kept under 1.3Gb.

Data set	Size	(a)	(b)
Neghip	$64 \times 64 \times 64$	8s	7s
Hydrogen	$128 \times 128 \times 128$	47s	27s
Aneurism	$256 \times 256 \times 256$	5m 1s	3m 51s
Instability	$1152 \times 1152 \times 1000$	23h 15m 22s	∞

Table 6.1: An Morse-Smale complex is computed for well-known datasets. We compare the run time of our algorithm (a) to the fastest previously published algorithm presented in [26] (b).



Total Run Time	23h 15m 22s
∇ + Morse-Smale-complex on parcel	2h 9m 45s
merging parcels	2h 38m 52s
5% simplification	18h 12m 41s
# cancellations	51,004,765
# parcels	1,000
# merge operations	1,000
# remaining critical points	957,560
# remaining arcs	6,320,506

Figure 6.6.2: A single timestep of a dataset of a simulated Raleigh-Taylor instability simulating the mixing of two fluids. This timestep has a resolution of $1152 \times 1152 \times 1000$ and is an early timestep of the simulation. The data is noisy, therefore we perform a 5% persistence simplification to remove "excess features." We compute the complex for the entire dataset, and the inset shows a small subsection of the data with selected nodes and arcs of the complex. Minima and maxima (blue and red spheres) and their saddle connections trace out the bubble structure in the data. The maxima represent isolated pockets of high-density fluid that have crossed the boundary between the two fluids. The structural complexity is overwhelming, but our prototype allows interactive exploration and visualization, and selective inclusion/omission of user-specified components of the Morse-Smale complex.

6.6.5 Analysis of the Algorithm

RUN TIME ANALYSIS The run time analysis of the complete algorithm is heavily dependent on the particular implementation of parcels and the data manager. The run time analysis of the particular steps of the algorithm can be performed using certain reasonable assumptions, such as access to the faces and co-faces of a cell requiring constant-time processing. The discrete gradient computation on a parcel with n cells uses a sorted ordering provided by the parcel, which is of complexity $O(nlogn)$. The computation of the Morse-Smale complex on a parcel performs a depth-first search from each critical cell which will cover its entire ascending manifold. Since the ascending manifolds can merge, if there are $O(n)$ critical cells, in the worst case, this step can require $O(n^2)$ time. However, in practice, the number of critical points can be modeled as a constant k, and tracing the ascending and descending manifolds requires $O(n)$ time. Merging two parcels with m cells on the interface is accomplished in $O(mlogm)$ time, as the gradient computation again requires $O(nlogn)$, and the merging of the complexes requires $O(m)$ time. The cancellation of a pair of nodes where the number of neighbors of each is bounded by some value i requires at most $O(i^2)$ time. Therefore, removing the artifacts introduced in a merge operation requires $O(k\ i^2)$.

We analyze the run time for the particular implementation used for generating the results, where slices are attached to a growing base for regular data. Let n be the total size of the data. For each of $n^{1/3}$ slices, a slice is read from the data, and cells are created and initialized in a traversal of the slice taking $n^{2/3}$ time. The gradient and complex are computed on the slice taking $n^{2/3}logn^{2/3} + n^{2/3}$ time. Each slice is then merged in $n^{2/3}logn^{2/3} + n^{2/3}$ time, and simplified in $i\ k^2$ time for a total run time of $O(nlogn) + n^{1/3}i\ k^2$. We do not remove the constant final term, since in the worst case this can lead to a total number of n^2 operations.

The memory requirements of our method are determined mainly by two parts: the overhead required for storing the gradient on a parcel, and the storage required for the computed Morse-Smale complex. Once the Morse-Smale complex has been computed on a parcel, the interior cells can be removed from memory. In fact, a parcel, with its boundary gradient, external cells, and Morse-Smale complex, only needs to be kept in memory during a merge operation. Let a parcel

P have n interior cells and m boundary cells. During computation of the discrete gradient field, the total footprint of P is $(n + m) \times |\alpha| + |K| + |\Gamma|$, where $|\alpha|$ is the size of a single cell, $|K|$ is the memory overhead of the data structures storing the CW-complex, and $|\Gamma|$ is the size of the Morse-Smale complex computed on the parcel. During the merging of two parcels P_1 and P_2, the total amount of memory required is $(m_1+m_2)\times|\alpha|+|K_1|+|K_2|+|\Gamma_1|+|\Gamma_2|$. The Morse-Smale complexes Γ_1 and Γ_2 can be simplified independently prior to the merging operation to reduce their sizes. For regular data, the mesh connectivity is defined implicitly, therefore $K = 0$. In the particular implementation we used for generating the results, given a dataset of size $x \times y \times z$, each parcel is a slice of the data requiring $x \times y \times 8 \times |\alpha|$ space, and only two parcels (the base and the new slice) were kept in memory.

6.6.6 Implications of Divide-and-Conquer Approach

The discrete gradient is first computed on the boundary of a parcel and then in the interior, and the restriction of the flow on the boundary potentially creates different Morse-Smale complexes for the same data if the data is divided in different ways. In simulation of simplicity, order-dependence determines the structures identified whenever degeneracies are encountered, such as flat regions and multi-saddles. The augmented function and the flexible ordering of the pairing of cells allow us to pick a particular ordering such that the flow can be fixed first on the boundary, then on the interior of a parcel, while maintaining consistency. In practice, a subset of the cells of the $d-1$-dimensional Morse-Smale complex restricted to the boundary of a parcel form the intersection of the cells of the d-dimensional Morse-Smale complex with the boundary. As a result, after merging parcels and simplifying the artifacts introduced in the process, the choices made in dividing the data result in only slight geometrical differences in the computed Morse-Smale complexes. Most significantly, however, the complexes extracted are consistent, which means that they represent a Morse function arbitrarily close to the function defined by the scalar values at vertices. Figure 6.6.3 shows the difference when slicing the same data across the z axis and across the x axis. The discrete gradient produced is always a valid gradient field with monotonically descending V-paths.

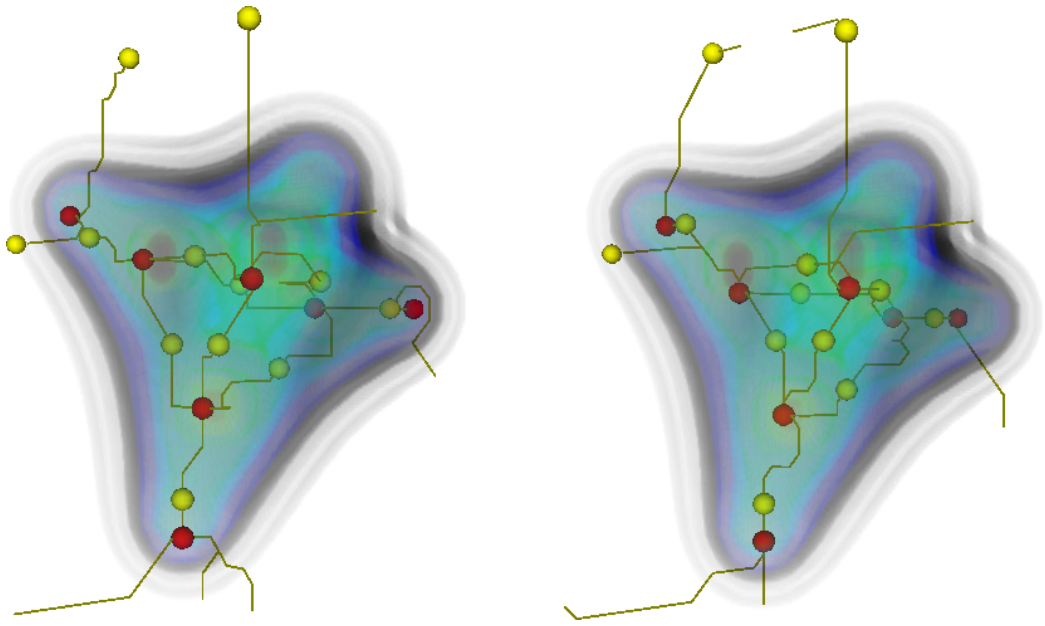


Figure 6.6.3: The critical points of the tetrahedrane are identified using slicing across the z axis (left) and across the x axis (right). The locations of the critical points vary up to a cell, as do the shape of the arcs connecting them. However, the fundamental structure that is found by both methods is the same. Note that the Morse-Smale complexes found with each slicing direction are consistent with one another, and both are consistent with the Morse-Smale complex found for the same dataset in [24].

6.6.7 Discussion

The algorithm we presented in this chapter is robust and efficient, and the framework is general and works for various data formats of any dimension. Our divide-and-conquer strategy allows for memory-efficient computation of the Morse-Smale complex and simplification on the fly to control the size of the output. We sacrifice some time efficiency to gain relatively more storage efficiency and scalability. Furthermore, the operations on each parcel are independent and can be computed in parallel, and our future work will be directed at a parallel implementation. The algorithm works for data of any dimension, and we will investigate using slicing across the time axis in 3D + time data to track features over time. The majority of time in computing the Morse-Smale complex is spent on simplification.

Chapter 7

Practical Considerations in Computing Morse-Smale Complexes

Name	Size	Values	Description	Challenge
Artificial	$64 \times 64 \times 64$	Byte	An artificially generated data set where every cell is critical and all arcs have zero persistence	No implicit ordering of arcs for cancellation
Hydrogen	$128 \times 128 \times 128$	Byte	The spatial probability distribution of an electron around a hydrogen atom in a strong magnetic field	Large flat regions with simple persistent features
Aneurism	$256 \times 256 \times 256$	Byte	A rotational X-ray scan of an aneurism	Flat regions interspersed with noise
Porous	$230 \times 230 \times 325$	Float	Distance to an interface surface in a simulated porous solid	Noisy floating-point data with flat ridge lines

Table 7.1: The data sets used for the generation of the timing data presented in this research were selected to examine particular aspects of performance.

Key issues remain un-addressed in previous chapters without which computation of the complex might have an unbounded memory footprint, identify an excessive number of critical points, or have undesirable artifacts on the boundaries. We examine the causes of such problems and present the necessary techniques for overcoming them in this chapter. These include reordering critical point cancellation operations for better performance, handling different boundary conditions, a generalized notion of simulation of simplicity to reduce the number of zero-persistence critical point pairs, and a highly efficient data structure for storing the Morse-Smale complex.

7.1 Data Sets

The timing and memory performance statistics presented throughout this chapter were generated on an off-the-shelf 2.21GHz AMD Athlon processor with 2.0Gb memory. Data sets were chosen to stress particular aspects of the Morse-Smale complex construction algorithm. The data sets used are summarized in table 7.1.

7.2 Efficient Cancellations

We use the description of the atomic cancellation operations provided in section 6.2 as a basis for examining the necessary major improvements to enable simplification of the Morse-Smale complex in practice.

Figure 6.2.1 shows this operation for a two-dimensional Morse-Smale complex. The a cancellation operation creates and destroys several arcs in the Morse-Smale complex, and therefore an efficient data structure is necessary to handle the operations involved. Let $a = (l, u)$ be an arc that is cancelled, n be the number of critical points of index $i + 1$ in N_l, and m be the number of critical points of index i in N_u. In the fist step of canceling a, $n \times m$ new arcs are added to the Morse-Smale complex. Each arc that is created must be inserted into the set of arcs of the nodes at its endpoints, resulting in $O(n \times m)$ INSERT operations. In the second step, the arcs connecting with the cancelled nodes are removed from the complex. Each arc that is removed from the complex must be deleted from the set of arcs of the nodes at its endpoints, resulting in $O(|A_l| + |A_u|)$ DELETE operations. Therefore, a data structure for storing the arcs at each node must support efficient INSERT and DELETE operations.

In early implementations [22, 26, 25, 24], all arcs connected to a node were stored in a linked list. INSERT operations were performed in constant time, however, the DELETE operations were linear time in the number of arcs $|A_n|$ connected to a node n. Therefore, the total running time for a cancellation was $O((n \times m) + ((|A_l| + |A_u|) \times A_m))$ where m is the node with the largest number $|A_m|$ of connected arcs in $N_u \cup N_l$.

The first term in this running time can lead to a quadratic increase in the number of arcs in the complex: not only does it create $n \times m$ new arcs, but every index-i node in N_u has m arcs added to its set of connected nodes, and every index-$(i + 1)$ node in N_l has n arcs added. Therefore, future cancellations between pairs of these nodes become more costly. Therefore, performing k cancellations in a naive ordering can therefore lead to an actual $kn \times km$ cost. Such situations often arise when large flat regions have many adjacent 0-persistence arcs, as is the case in most integer-valued data. The second term in the running time can also be prohibitive, as nodes with a

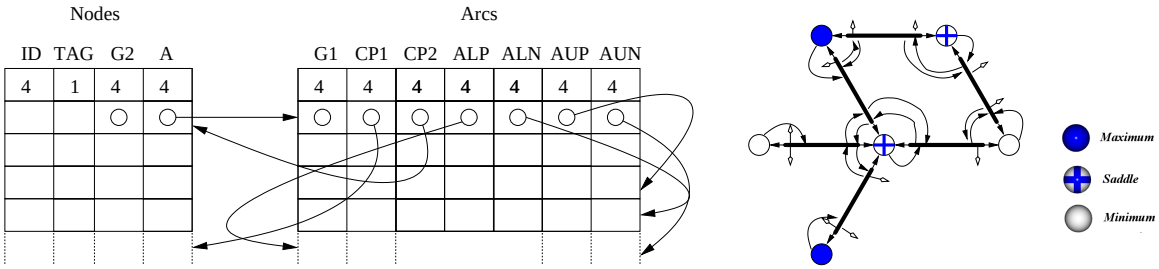


Figure 7.2.1: The nodes and arcs of the complex are stored as fixed-size elements in an array (a) with the arcs containing the link information of the arc lists of their lower and upper nodes. ALN and ALP are the indices of the next and previous arcs in the double-linked arc list of the lower node, and AUN and AUP are the indices of the next and previous arcs in the list of the upper node. Each arc additionally has a pointer to its lower and upper nodes, and a pointer to the geometry of the arc. (b) shows a sample two-dimensional complex with the pointers at each node and arc.

large number of connected arcs slow the cancellation of every node that is connected to it. In fact, a linear-time DELETE operation in the number of arcs connected to a node leads to quadratic-time cancellations in the number of arcs connected to each endpoint. We present a data structure for constant-time DELETE operations, and present several strategies for preventing a dramatic increase in the number of arcs.

7.2.1 Efficient Data Structure

We use a double-linked list to store the set of arcs A_n connected to a node n. The INSERT operation can be done in constant time, since new arcs are always inserted at the beginning of the list. A DELETE operation in a linked list requires finding the element to remove, the previous element, and the next element, and reconnecting to remove the element. Finding an element is linear in the size of the list, while reconnecting is constant. A key observation is that in a cancellation operation, finding an arc in A_n is done in constant time, since the arc element is previously identified during an iteration of the arcs of its other endpoint. The previous and next elements must still be found, and in a double-linked list this is done in constant time. Therefore, the entire DELETE of an arc takes constant time in a cancellation operation. This behavior represents an improvement over the linear time for the linked list used in previous implementations. This improvement is possible, because the

Data set	Artificial	Hydrogen	Aneurism	Porous
(a)	620	951	638	3224
(b)	5130	5426	5006	5510

Table 7.2: A chart showing the sustained cancellation rate to cancel arcs 20% of maximum persistence for various data sets. (a) lists the number of cancellations per second using a standard linked list, and (b) shows the cancellations/second using a double-linked list. Note that the porous data set has a high rate of cancellations even when using a linked list, because it has very few high-valence nodes.

number of arcs produced by the discrete Morse-Smale computation algorithm is small enough that the additional memory cost is insignificant. This optimization is not recommended for the algorithm presented in chapter 4, where initially 24 arcs must be stored for each vertex of the input.

Figure 7.2.1 illustrates the data structure used to store the nodes and arcs of the Morse-Smale complex. Although this data structure is 1.4 times larger than a single-linked list, the majority of the memory footprint of a complex is taken up by the geometry components, such as storing the set of line segments that define an arc. In fact, for a typical complex, the additional space required by the arcs to store the double-linked list is only 0.1%. Table 7.2 compares the performance of using a standard linked list versus a double-linked list.

7.3 Cancellation Strategy

Simplification of a function is achieved by repeated cancellation of critical point pairs. The ordering of critical point pairs is defined by *persistence*, which quantifies the importance of the topological feature associated with a pair. The *persistence* of a critical point pair is the absolute difference in value of f between the two points. Typically, in this ordering, the arc with the lowest persistence is cancelled first. However, such an ordering can lead to an unbounded memory footprint. Instead, we design a strategy to determine when a cancellation is delayed, based on conditions imposed on the neighborhood of the critical points to be cancelled. We still maintain the property that all cancellations must be valid, *i.e.*, the pair of nodes is connected by exactly one arc, and the arc to be cancelled has the lowest persistence of any in the neighborhood around the two nodes.

7.3.1 Limit $n \times m$

A straightforward technique to explicitly prevent the dramatic increase in memory is to delay a cancellation until the number of new arcs it would create, $n \times m$, is below some threshold t. Arcs are cancelled in order of persistence, however, if canceling an arc were to lead to the creation of more than t new arcs, the arc is be removed from the ordering and put in a delayed list, and another arc is selected for cancellation. Cancellation of arcs in the neighborhood of delayed arcs can reduce the number of arcs incident at the nodes at either endpoint. Delayed arcs are cancelled as soon as they create fewer than t new arcs. Table 7.3 shows the memory footprint and running times for several data sets as a function of t. While lower t values generally improve performance, they can lead to artifacts in the form of low persistence arcs that are perpetually delayed. When t is too high, the memory cost and run time increases, sometimes preventing the termination of the algorithm. However, t should be chosen as high as possible to keep the number of artifacts small.

7.3.2 Global Valence Control

The *valence* of a node is the number of arcs that have it as an endpoint. Nodes with a high valence are expensive to cancel, and tend to generate many new arcs, leading to more high-valence nodes. We present a strategy to prevent the creation of high-valence nodes by delaying the cancellation of arcs that lead to the creation of a high-valence node. When an arc is canceled, the valence of nodes in a neighborhood around the canceled pair changes. Let n be the number of index-$(i+1)$ nodes in the neighborhood N_l of the lower node, m be the number of index-i nodes in the neighborhood N_u of the upper node. Let $a_i \in N_l$ be a node in the neighborhood of the lower node, and $b_i \in N_u$ be a node in the neighborhood of the upper node, and define $V(a)$ to return the valence of node a. We define the weight of an arc as $W(a) = max(max(V(a_i)+m|a_i \in N_l), max(V(b_i)+n|b_i \in N_u))$. Intuitively this value is the largest valence that cancelling a would create. A cancellation is delayed if the weight of the arc is greater than a threshold t. Table 7.4 shows the memory footprint and running times for several datasets as a function of t. Similarly to the new arc limiting method, lower t values correspond to higher performance and an increased number of artifacts. Computing

Data Set	t=20	t=40	t=80	t=160	t=360	t=720
Artificial-Time	–	6m 34s	12m 40s	11m 51s	25m 31s	–
Artificial-#Arc	–	13,054K	14,392K	14,398K	25,652K	–
Artificial-Atf.	–	887,281	145,332	24,045	10,244	–
Hydrogen-Time	0.86s	1.20s	2.03s	2.45s	2.63	3.30s
Hydrogen-#Arc	34,997	35,670	42,005	42,325	45,860	52,483
Hydrogen-Atf.	186	4	0	0	0	0
Aneurism-Time	–	1m 35s	4m 59s	8m 10s	16m 45s	47m 32s
Aneurism-#Arc	–	1,971K	2,450K	4,508K	7,616K	12,320K
Aneurism-Atf.	–	23,320	1,625	122	34	6
Porous-Time	6m 25s	7m 04s	9m 32s	15m 45s	17m 16s	34m 47
Porous-#Arc	6,202K	6,402K	8,300K	9,224K	12,535K	35,541K
Porous-Atf.	352	202	54	17	6	5

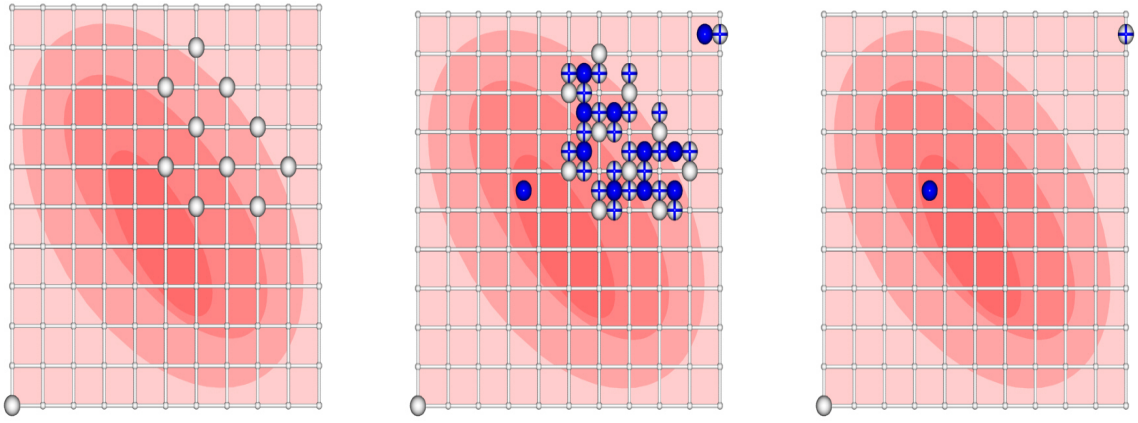
Table 7.3: For each data set, we provide as a function of t: the time required to cancel up to 20% persistence, the maximum number of arcs in the complex, and the number of low-persistence artifacts that could not be cancelled, delaying arcs whose cancellation would generate more than $n \times m$ new arcs. In general, a higher threshold results in longer run times, a larger memory footprint, but fewer artifacts. Blanks indicate non-termination. The Morse-Smale complexes were generated using the improved simulation of simplicity from Sect. 5.

Data Set	t=10	t=20	t=40	t=60	t=80
Artificial-Time	5m 20s	7m 35s	12m 40s	11m 51s	30m 52s
Artificial-#Arc	12,409K	13,003K	15,492K	16,255K	25,652K
Artificial-Atf.	724,381	45,351	15,045	1,244	
Hydrogen-Time	1.06s	1.25s	2.18s	3.06s	3.54s
Hydrogen-#Arc	35,206	37,288	43,110	47,757	52,123
Hydrogen-Atf.	72	0	0	0	0
Aneurism-Time	1m 32s	2m 15s	10m 21s	17m 45s	25m 26s
Aneurism-#Arc	1,870K	1,930K	4,603K	8,142K	11,242K
Aneurism-Atf.	26,076	18,450	102	3	2
Porous-Time	6m 05s	8m 36s	12m 29s	15m 54s	19m 24s
Porous-#Arc	5,730K	7,404K	10,004K	13,224K	15,853K
Porous-Atf.	266	65	23	6	4

Table 7.4: For each data set, we provide as a function of t: the time required to cancel up to 20% persistence, the maximum number of arcs in the complex, and the number of low-persistence artifacts that could not be cancelled, delaying arcs whose cancellation would produce a node of valence greater than t.

Data set	Hydrogen	Aneurism	Porous
Standard Simp.	846,784	2,661,251	13,172,740
BFS Simplicity	3571	180,267	1,074,734

Table 7.5: The total number of critical points identified using a slice-by-slice computation of the discrete gradient. The top row lists the numbers of critical points found when the standard simulation of simplicity is used to order cells. The bottom row shows the numbers of critical points identified with the on-the-fly ordering.



(a) (b) (c)

Figure 7.4.1: Let the $index$ of a vertex $v = (x, y)$ be $x + (y * xdim)$. Using standard simulation of simplicity, minima are identified at the boundaries of flat regions (a). When constructing an Morse-Smale complex, these minima further generate the necessary separating saddles and maxima (b), far more than the actual three persistent critical points (c).

$V(a_i)$ requires storage of the valence at each node, and an update of it whenever the neighborhood changes.

7.4 Improved Simulation of Simplicity

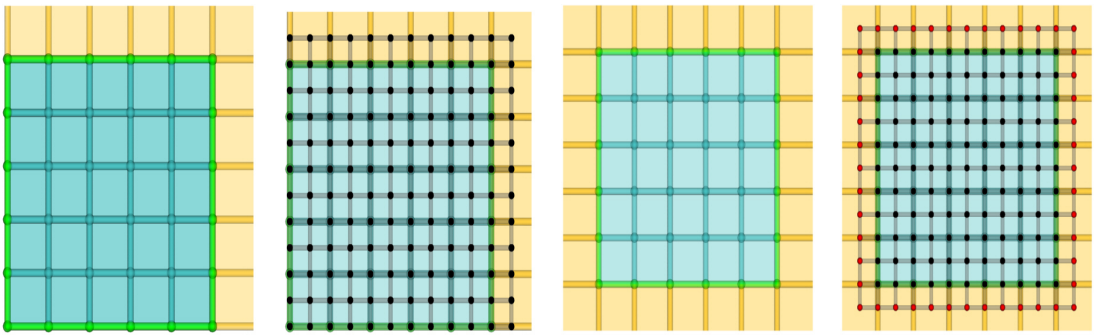
Edelsbrunner and Mücke [17] introduced simulation of simplicity to resolve degenerate conditions where the input function is not a Morse function. For example, when two vertices with equal scalar value are compared, simulation of simplicity consistently identifies the one with the higher index as having higher value. While this simple method of breaking ties provides a complete ordering of the vertices of a data set, it introduces critical points that only exist due to index comparison.

Figure 7.4.1 shows that for a scalar function with flat regions, such an ordering produces spurious critical points at the boundaries of the flat regions. When computing the Morse-Smale complex, these extra critical points introduce a significant processing and data size overhead.

The $sortedCellIterator(i)$ simulates simplicity, since it defines an ordering of cells of dimension i. This iterator can be implemented as repeated application of a $getLowestCell(i)$ operation, which returns the current lowest unmarked i-cell. This allows for an ordering that is determined on-the-fly. In particular, we implement the $getLosestCell(i)$ to return the cell with the lowest value, with ties going to a cell that can be paired. In fact, with this ordering, the gradient pairing performs a breadth-first search across flat regions to pair all possible cells, reducing the number of critical cells found.

This operation is implemented using two priority queues, $simpleOrder$ and $bfsOrder$. The $simpleOrder$ priority queue is sorted based on the original simulation of simplicity, *i.e.*, comparing cells by value and index. The $bfsOrder$ priority queue is sorted based on scalar value and insertion time: a cell a is inserted after all other cells already in the queue with the same value. The $getLowestCell(i)$ operation is implemented as follows:

```
1: getLowestCell(int i)
2: while ! bfsOrder→empty() && ! bfsOrder→top()→isMarked() do
3:     bfsOrder→pop()
4: end while
5: iCell α = bfsOrder→top()
6: while ! simpleOrder→empty() && ! simpleOrder→top()→isMarked() do
7:     simpleOrder→pop()
8: end while
9: iCell β = simpleOrder→top()
10: if α→value() ≤ β→value() then
11:     bfsOrder→pop()
12:     return α
13: else
```



(a) (b) (c) (d)

Figure 7.5.1: A 2-manifold with boundary (green) is made periodic by attaching cells (orange) along boundaries (a). The cells of gridded data can be represented by vertices in a refined mesh (b), and the neighborhood of cells is computed using modular arithmetic. A 2-manifold with interior and boundary undergoes one-point compactification by attaching cells (orange) in a layer around the boundary (c). The red vertices (d) indicate the cells that remove the boundary, and flow must not cross these cells.

14: simpleOrder→pop()

15: return β

16: **end if**

Reducing the number of critical cells improves the performance by avoiding tracing arcs through the gradient field and avoiding cancellation of zero-persistence pairs. The performance, number of critical points found, and memory footprint are summarized in Table 7.5. Although this method was primarily designed to help overcome the flat regions in integer-valued data, a large improvement was also seen in the porous data set, a floating-point data set. This behavior indicates that discretizing a function makes it very sensitive to the ordering of cells, even when the original values are reasonably distinct floating-point numbers.

7.5 Boundary Conditions

The Morse-Smale complex is defined for scalar functions defined on manifolds without boundary. Boundaries of a manifold present problems in computation of the Morse-Smale complex, since they require special rules for the identification and simplification of critical points. In practice, it is

much easier to convert the space to a manifold without boundary, and avoid the special conditions. Three techniques for removing the boundary are (1) assuming a periodic boundary, (2) one-point compactification, and (3) mirroring along the boundary.

7.5.1 Periodic Boundary Conditions

Let K be a regular complex that is a mesh representation of an n-manifold with boundary $\mathbb{M}$. Let $B \subseteq K$ be the set of cells that represents the boundary of $\mathbb{M}$. A periodic boundary condition is defined by repeated attachment of pairs of subsets of the boundary. Figure 7.5.2 shows this process. The implementation of periodic boundary conditions relies on the ability to create a continuous onto map between subsets of the boundary. In practice, $\mathbb{M}$ is usually embedded in some Euclidean space, and the mapping between subsets of the boundary is given implicitly by the spatial location of the boundary cells. For example, given a 2-manifold grid having a boundary consisting of n cells along the x-axis and m cells along the y-axis, a cell with coordinates $(0, i)$ is attached to a cell with coordinates $(n-1, i)$, and a cell with coordinates $(i, 0)$ is attached to a cell with coordinates $(i, m-1)$.

Implementation of periodic boundary conditions for grids can be achieved by using modular arithmetic to compute the neighbors of a cell. For example, on a two-dimensional grid G with $X \times Y$ vertices, a $(2 \times X) \times (2 \times Y)$ grid G^2 has a vertex representing every cell of the original grid. The cell at (i, j) of G^2 has four neighbors, $((i-1+(X \times 2))\%(X \times 2), j)$, $((i+1)\%(X \times 2), j)$, $(i, (j-1+(Y \times 2))\%(Y \times 2))$, and $(i, (j+1)\%(Y \times 2))$. Figure 7.5.1 illustrates the new cells that are created to attach the boundaries.

Using periodic boundary conditions to compute the Morse-Smale complex is only appropriate where the data itself assumes periodic boundary conditions. Arcs of the complex can cross the periodic boundary, representing a flow across the boundary.

7.5.2 One-point Compactification

One-point compactification removes the boundary of a manifold by attaching every boundary cell to a vertex at infinity. Figure 7.5.3 shows this process. The vertex at infinity is assumed to have

a value of negative infinity, so that any arc connecting to it has infinite persistence and cannot be cancelled.

Implementation of one-point compactification for gridded data can be achieved by creating a refined grid of cells with an extra layer of cells around the boundary. Figure 7.5.1 shows this configuration. For example, on a two-dimensional grid G with $X \times Y$ vertices, a $(2 \times X) + 1 \times (2 \times Y) + 1$ grid G^2 has a vertex representing every cell of the original grid and an extra layer of cells all around that attach the boundary to the point at infinity. The point at infinity is not represented explicitly.

During computation of the discrete gradient field, flow must not cross the attaching cells to prevent the point at infinity from affecting the Morse-Smale complex on the interior. Therefore, attaching cells can only be paired with other attaching cells, and interior cells can only be paired with interior cells. Similarly, only arcs whose endpoints are either both attaching cells or both interior cells may be cancelled, as such a cancellation would simulate flow crossing the attaching cells to the point at infinity.

7.5.3 Mirrored Boundary

Let K be the CW complex representing an n-manifold $\mathbb{M}$ with boundary, let $B \subseteq K$ be the boundary cells, and $I = K \backslash B$ be the interior cells. There are two ways to achieve mirrored boundary conditions:

1. The input CW complex K is duplicated and the copy is denoted K'. Each cell on the boundary of K is attached to the copy of itself in K'.

2. The interior I of the CW complex K is duplicated and the copy is denoted I'. The cells of I' share a common boundary with the cells of I.

In each case, the function value of a vertex in the mirrored CW complex is the same as its function value in the original CW complex. Figure 7.5.4 shows these two kinds of mirroring conditions. Mirroring removes the boundary without creating many boundary artifacts. In theory, the discrete gradient and Morse-Smale complex computed on either side of the boundary is a mirror

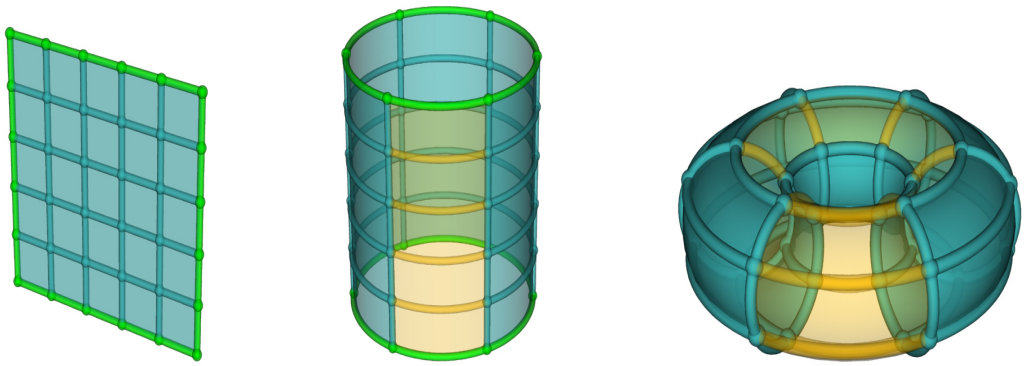
of the other. To achieve this stat in practice, the pairing of cells must be restricted when creating the discrete gradient, and cancellations that would lead to flow crossing the mirror boundary are not allowed.

This approach can be implemented in a straightforward manner by allowing only pairing of cells where both cells are in the boundary or in the interior, and allowing only cancellations where both nodes are in the interior of K or both are on the boundary. Note that when considering these conditions, the resulting Morse-Smale complexes are the same whether we use the first or second mirroring method, since any additional critical cells created in the first method are created in zero persistence pairs, and are immediately removed.

7.6 Discussion

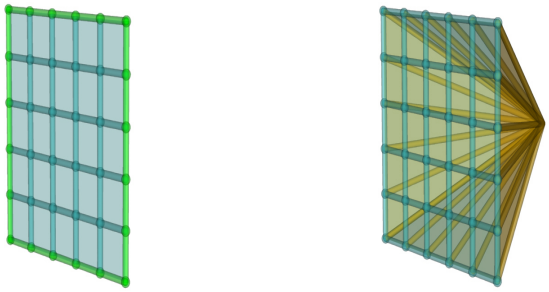
Both techniques for reordering cancellations make it possible to simplify an Morse-Smale complex without entering a situation where unregulated cancellations lead to unbounded memory usage. The double-linked list introduced to store the arcs connected to a node further provides a decrease in run-time complexity of cancellation operations. Using the improved simulation of simplicity results in a more efficient implementation, especially in data sets with large flat regions. In the case of the hydrogen dataset, the improved simulation of simplicity generated 200 times fewer critical cells than the standard simulation of simplicity. The various techniques for handling boundary conditions make it possible to choose the most appropriate conditions for the analysis of a particular data set. The techniques we have presented in this chapter are not just for improving performance; without them, computation and simplification of the Morse-Smale complex would not be possible, even for simple data. For example, without the improved simulation of simplicity, the Morse-Smale complex of the hydrogen data set has almost one million low-persistence critical points, due to a naive simulation of simplicity. Simplification of this large complex would not terminate without re-ordering of cancellations. The hydrogen data set is a small data set with fewer than 100 persistent critical points, and the Morse-Smale complex can be computed and simplified in less than a minute using the techniques discussed in this chapter. The techniques we have presented enable efficient

implementation of algorithms to compute and simplify the Morse-Smale complex.



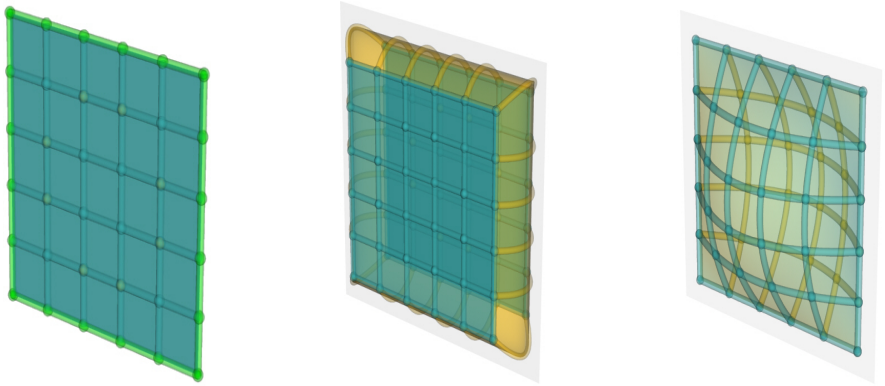
(a) (b) (c)

Figure 7.5.2: A 2-manifold with boundary (a). The subsets of the boundary corresponding to the left and right sides are attached to one another (b). The new boundary circles on the top and bottom are attached (c) to form a 2-manifold without boundary.



(a) (b)

Figure 7.5.3: A 2-manifold with boundary (green) is made periodic by attaching cells (orange) along boundaries (a). The cells of gridded data are represented by the vertices a refined mesh (b), and the neighboring cells are computed using modular arithmetic.



(a) (b) (c)

Figure 7.5.4: A 2-manifold with interior (blue) and boundary (green) cells (a). Mirroring using the first method (b) duplicates the entire mesh and attaches boundary cells to their copies. Mirroring using the second method (c) only duplicates the interior. Note that a retracting the new boundary in (b) along the attaching cells yields the same mesh as in (c).

Chapter 8

Conclusions and Possible Future Directions

In this dissertation, we have presented two algorithms for computing Morse-Smale complexes on data of any dimension. The first method (chapter 4) is based on simplification of the artificial complex, and is the first practical method for computing a Morse-Smale complex. This method overcomes challenges faced by other methods by bypassing degenerate conditions such as multi-saddles and non-transversal intersections: the artificial complex generated according to the augmented function has a regular structure, and valid cancellations in this complex cannot lead to degenerate conditions. We improved this algorithm by computing the complex in a sweep plane approach.

In chapter 5, we showed how the Morse-Smale complex could be used to extract features from a real-world dataset. In particular, we performed analysis on the structure of the arcs of the complex to find a stable representation of the core structure of porous solid. The resulting quantitative analysis of the core structure answered an important question in material sciences.

Next, we presented another algorithm (chapter 6) for computing the Morse-Smale complex, one based on discrete Morse theory. The kernel of this alrogithm is the procedure to construct a discrete gradient field from discrete samples at the vertices of a CW-complex. This method is dimension-independent and robust. We presented a divide-and-conquer approach using this kernel, to enable computation of discrete Morse-Smale complexes for large data. Furthermore, we presented a general framework that can support many data formats, such as regular grids, simplicial meshes, and AMR meshes, while still enabling efficient implementations where the data format permits. We demonstrated the first-ever computation of the Morse-Smale complex for a dataset with 1B vertices.

Finally, we addressed some of the practical problems (chapter 7) in computing Morse-Smale complexes. These issues include improving the performance of cancellations by using an improved data structure, reordering cancellations to avoid the in-practice creation of an excessive number of arcs, improving the simulation of simplicity to reduce the number of zero-persistence critical points found by the discrete algorithm, and handling different boundary conditions.

There are many directions in which to take future work. The algorithms for computing the Morse-Smale complex presented in this dissertation sacrifice geometric accuracy for topological robustness; a piecewise constant interpolant, or discrete representation of the function is assumed.

A future direction for research might include computing the Morse-Smale complex that has small associated error with higher degree interpolants. Another direction might be the analysis of time-varying data using either higher dimensional Morse-Smale complexes or time-varying Morse-Smale complexes. One of the key areas for progress is in application of the techniques presented in this dissertation to real-world data. By developing robust and efficient software to automate the computation and filtering of the Morse-Smale complex, the techniques for analysis presented in this dissertation will be accessible to the broader scientific community.

Bibliography

[1] R. F. W. Bader. *Atoms in Molecules*. Clarendon Press, Oxford, 1995.

[2] S. Beucher. Watershed, hierarchical segmentation, and waterfall algorithm. *Mathematical Morphology and its Applications to Image Processing*, pages 69–76, 1994.

[3] S. Beucher and C. Lantuéjoul. Use of watersheds in contour detection. *In Proceedings of International Workshop on Image Processing, Real-Time Edge and Motion Detection/Estimation*, September 1979.

[4] R. L. Boyell and H. Ruston. Hybrid techniques for real-time radar simulation. In *IEEE Proceedings Fall Joint Computer Conference 63*, pages 445–458, 1963.

[5] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):385–396, 2004.

[6] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. In *SODA '00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 918–926, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

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

[8] K. Cole-McLaughlin, H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Loops in reeb graphs of 2-manifolds. *Discrete Comput. Geom.*, 32(2):231–244, 2004.

[9] G. E. Cooke and R. L. Finney. *Homology of Cell Complexes*. Princeton University Press, Princeton, NJ, 1967.

[10] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2001.

[11] N. D. Cornea, D. Silver, and P. Min. Curve-skeleton applications. *IEEE Conf. Visualization*, 0:95–102, 2005.

[12] H. Digabel and C. Lantuéjoul. Iterative algorithms. *Second European Symposium on Qualitative Analysis of Microstructures in Material Science, Biology and Medicine*, pages 85–99, 1978.

[13] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Trans. Graph.*, 25(3):1057–1066, 2006.

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

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

[16] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete and Computational Geometry*, 28(4):511–533, 2002.

[17] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9(1):66–104, 1990.

[18] R. Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134(1):90–145, 1998.

[19] R. Forman. A users guide to discrete morse theory. In *Proc. of the 2001 Internat. Conf. on Formal Power Series and Algebraic Combinatorics, A special volume of Advances in Applied Mathematics*, page 48, 2001.

[20] H. Freeman and S. P. Morse. On searching a contour map for a given terrain elevation profile. *Journal of the Franklin Institute*, 284(1):1–25, 1967.

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

[22] A. Gyulassy, P.-T. Bremer, B. Hamann, and V. Pascucci. A practical approach to morse-smale complex computation: Scalability and generality. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1619–1626, 2008.

[23] A. Gyulassy, M. Duchaineau, V. Natarajan, V. Pascucci, E. Bringa, A. Higginbotham, and B. Hamann. Topologically clean distance fields. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1432–1439, 2007.

[24] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann. Topology-based simplification for feature extraction from 3d scalar fields. In *Proc. IEEE Conf. Visualization*, pages 535–542, 2005.

[25] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann. A topological approach to simplification of three-dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):474–484, 2006.

[26] A. Gyulassy, V. Natarajan, V. Pascucci, and B. Hamann. Efficient computation of morse-smale complexes for three-dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1440–1447, 2007.

[27] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[28] H. King, K. Knudson, and N. Mramor. Generating discrete morse functions from point data. *Experimental Mathematics*, 14(4):435–444, 2005.

[29] D. Laney, P.-T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1053–1060, 2006.

[30] T. Lewiner. Constructing discrete Morse functions. Master's thesis, Department of Mathematics, PUC-Rio, 2002.

[31] T. Lewiner, H. Lopes, and G. Tavares. Applications of Forman's discrete Morse theory to topology visualization and mesh compression. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):499–508, 2004.

[32] A. T. Lundell and S. Weingram. *The Topology of CW-Complexes*. Van Nostrand Reinhold, New York, NY, 1969.

[33] D. M. Mark. Topological properties of geographic surfaces. *In Proceedings of Advanced Study Symposium on Topological Data Structures*, October 1977.

[34] D. M. Mark. Topological randomness of geomorphic surfaces. *PhD thesis, Simon Fraser University, Burnaby*, 1977.

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

[36] J. C. Maxwell. On hills and dales. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, XL:421–427, 1870.

[37] F. Meyer. Topographic distance and watershed lines. *Signal Process.*, 38(1):113–125, 1994.

[38] F. Meyer and S. Beucher. Morphological segmentation. *Journal of Visual Communication and Image Representation*, 1(1):21–45, 1990.

[39] J. Milnor. *Morse Theory*. Princeton Univ. Press, New Jersey, 1963.

[40] M. Morse. Equilibria in nature-stable and unstable. *Proceedings of the American Philosophical Society*, 93(3):222–225, 1949.

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

[42] L. Najman and M. Schmitt. Watershed of a continuous function. *Signal Process.*, 38(1):99–112, 1994.

[43] V. Natarajan and H. Edelsbrunner. Simplification of three-dimensional density maps. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):587–597, 2004.

[44] V. Pascucci and K. Cole-McLaughlin. Efficient computation of the topology of level sets. In *VIS '02: Proceedings of the conference on Visualization '02*, pages 187–194, Washington, DC, USA, 2002. IEEE Computer Society.

[45] V. Pascucci, G. Scorzelli, P.-T. Bremer, and A. Mascarenhas. Robust on-line computation of reeb graphs: simplicity and speed. *ACM Trans. Graph.*, 26(3):58, 2007.

[46] J. L. Pfaltz. Surface networks. *Geographical Analysis*, 8(1):77–93, 1976.

[47] J. L. Pfaltz. A graph grammar that describes the set of two-dimensional surface networks. In *Proceedings of the International Workshop on Graph-Grammars and Their Application to Computer Science and Biology*, pages 379–388, London, UK, 1979. Springer-Verlag.

[48] S. Rana and J. Morley. Surface networks. *UCL (University College London), CASA (Centre for Advanced Spatial Analysis), Centre for Advanced Spatial Analysis (UCL)*, 2002.

[49] 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*, 222:847–849, 1946.

[50] J. B. T. M. Roerdink and A. Meijster. The watershed transform: definitions, algorithms and parallelization strategies. *Fundam. Inf.*, 41(1-2):187–228, 2000.

[51] Y. Shinagawa and T. L. Kunii. Constructing a reeb graph automatically from cross sections. *IEEE Comput. Graph. Appl.*, 11(6):44–51, 1991.

[52] Y. Shinagawa, T. L. Kunii, and Y. L. Kergosien. Surface coding based on Morse theory. *IEEE Comput. Graph. Appl.*, 11(5):66–78, 1991.

[53] S. Smale. Generalized Poincaré's conjecture in dimensions greater than four. *Ann. of Math.*, 74:391–406, 1961.

[54] S. Takahashi, G. M. Nielson, Y. Takeshima, and I. Fujishiro. Topological volume skeletonization using adaptive tetrahedralization. In *GMP '04: Proceedings of the Geometric Modeling and Processing 2004*, page 227, Washington, DC, USA, 2004. IEEE Computer Society.

[55] S. Takahashi, Y. Takeshima, and I. Fujishiro. Topological volume skeletonization and its application to transfer function design. *Graphical Models*, 66(1):24–49, 2004.

[56] S. P. Tarasov and M. N. Vyalyi. Construction of contour trees in 3d in o(n log n) steps. In *SCG '98: Proceedings of the fourteenth annual symposium on Computational geometry*, pages 68–75, New York, NY, USA, 1998. ACM Press.

[57] M. J. van Kreveld, R. van Oostrum, C. L. Bajaj, V. Pascucci, and D. Schikore. Contour trees and small seed sets for isosurface traversal. In *Symposium on Computational Geometry*, pages 212–220, 1997.

[58] Z. Wood, H. Hoppe, M. Desbrun, and P. Schröder. Removing excess topology from isosurfaces. *ACM Transactions on Graphics*, 23(2):190–208, 2004.