The Geometry and Topology of the Cosmic Web
Spring 2014 CAMCOS report

San José State University

Au, Hai Nguyen Boersma, Catherine * Fitch, Joey
Goulette, David Liu, Jian-Long Litrus, Matthew
Morrison, Brandon Scargle, Jeff † Simić, Slobodan ‡

August 1, 2014

Contents

*Project leader - email: catherineboersma@hotmail.com
†Team Sponsor, NASA
‡CAMCOS Director and Faculty Supervisor, SJSU

1 Introduction

In this paper, we present a survey of promising approaches to the characterization of the geometry and topology of the distribution of galaxies within the visible universe, and report on the application of such methods to a subset of this data.

In particular, we worked with a subset of the Sloan Digital Sky Survey (SDSS), which is optical data collected from the 2.5 meter telescope at the Apache Point Observatory in New Mexico. Each point in the SDSS catalogue is an entire galaxy composed of hundreds of billions of stars. On the largest scales, structures can be identified by eye. These structures have been coined "The Cosmic Web" [1] and are comprised of over-dense and under-dense regions. Galaxies tend to group together and form compact structures called clusters, as well as elongated filaments and sheet-like walls. The cosmic web fills only a fraction of all space, leaving large mostly-empty regions, called voids. The subset of the SDSS considered in this work consists of about 134,000 galaxies. We make the following simplifying assumptions: (a) that the masses of each galaxy are nearly equal, and (b) that they are all relatively close to earth. These assumptions allow us to ignore the evolution of the cosmic web over time and, instead, allow us to treat the data as a snapshot (in time). In order to identify structures, we first partition the data, then group points together into classes, and finally classify the resulting structures into clusters, one-dimensional filaments, two-dimensional walls, and voids.

In this work, various different methods are brought to bear on the problem. The goal in each case is the same: to provide a computationally simple algorithm that is parameter-free (or, has as few parameters as possible) to identify multi-scale structures in the data. Inspired by the power of Discrete Morse Theory (DMT) and yet longing for a computationally simpler approach, we explore variations of HOP, a group-finding algorithm formulated by Eisenstein and Hut[2].

The first phase of this work involved adapting and applying DMT to the data set. Next, we explored the application of HOP to this problem. Along the way, variants of HOP were devised and applied. A particular variant (termed "Dimensional-HOP") is introduced and is shown to provide results similar to Discrete Morse Theory. Following this, we study the application of α-shapes. Finally, we introduce a novel technique that combines α-shapes with HOP, leading to new and interesting results.

The following is an overview of the rest of this paper. We begin by defining terms which are used throughout the various approaches. Then, we dedicate a chapter each to motivating, describing and summarizing results and lessons learnt from the various approaches that we considered and applied. In the penultimate chapter, we describe α-geodesics – our novel technique that combines two well-known approaches, α-shapes and HOP geodesics. We motivate this approach and describe its advantages over the existing methods. Finally, we

conclude with areas for further research. At the end, we provide appendices that contain mathematical preliminaries and formalisms that are used throughout the paper, as well as a more formal treatment of Discrete Morse Theory.

2 Definitions & Terminology

- **Our Data** - The data we are working with consists of 133,991 galaxies imaged in the Sloan Digital Sky Survey, hereafter SDSS [3].
- **Voronoi Diagram** - Let X be a space equipped with a distance function d. Let K be a set of indices and let $(P_k)_{k \epsilon K}$ be an ordered collection of non-empty subsets in X. The Voronoi cell, R_k, associated with the site P_k is the set of all points in X whose distance to P_k is not greater than their distance to the other sites P_j, where j is any index different from k. A Voronoi diagram (or Voronoi tessellation) is an ordered collection of all the Voronoi cells. In the SDSS data, each galaxy is a data point, the space is $\mathbb{R}^3$ and the metric is euclidean.

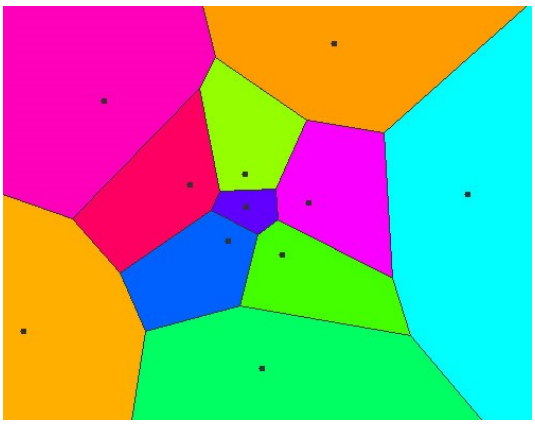


Figure 1: A Voronoi diagram in $\mathbb{R}^2$ for a set of eleven points. Each Voronoi cell is identified with a different color.

- **Delaunay Tessellation** - The Delaunay tessellation is the dual graph of the Voronoi Tessellation. In the Delaunay tessellation of the SDSS data, each galaxy is the vertex of a triangle in $\mathbb{R}^2$ or a tetrahedron in $\mathbb{R}^3$. The Delaunay tessellation consists of triangles in $\mathbb{R}^2$ such that when two Voronoi cells are adjacent to one another, there is an edge between their data points in the Delaunay tessellation.

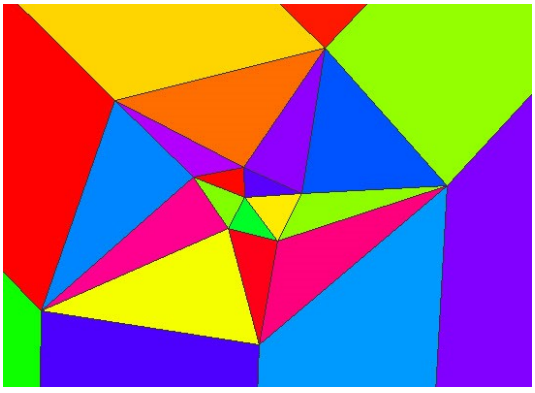


Figure 2: A Delaunay tessellation in $\mathbb{R}^2$ for a set of eleven points. Each point is the vertex of a triangle.

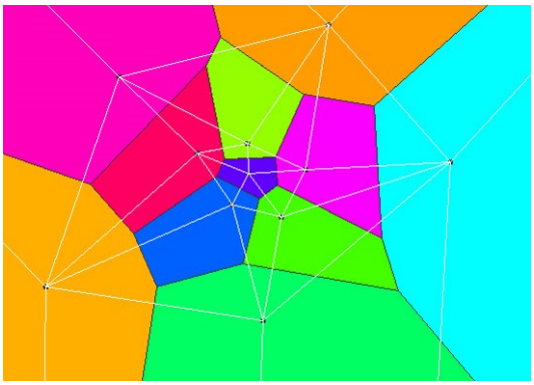


Figure 3: The dual nature of the Voronoi and Delaunay tessellations. The Voronoi cells are colored polygons and the Delaunay tessellation contains an edge between two adjacent Voronoi cells.

- **K-simplex** - A k-simplex is the k-dimensional analog of a triangle. For example, a 0-simplex is a point, a 1-simplex is a line segment, a 2-simplex is a triangle, and a 3-simplex is a tetrahedron.
- **Simplicial Complex** - (Definition 3.3 in [4]). A simplicial complex K is a finite union of simplices such that
  - Any face of a simplex in K also belongs to K.
  - The intersection of two simplices in K is empty or a simplex of dimension lower or equal to the highest dimensional simplex they share.

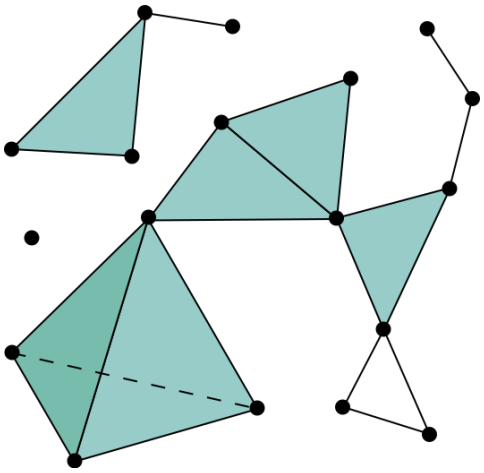


Figure 4: A valid simplicial complex

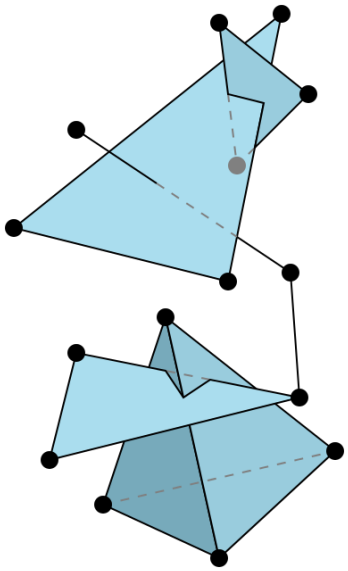


Figure 5: An arrangement of simplices which do not form a valid simplicial complex

- **Point Density** - Let ρ be the density of a galaxy point, p. We define the point density to be the inverse volume of a point's Voronoi cell in 3d.

$$\rho(p) = \frac{1}{\text{Vol}\,(p)} \tag{1}$$

3 Related Literature

In this section, we provide a brief overview of related literature.

Galaxy clusters were first identified in the 1970's as large collections of galaxies spanning hundreds of millions of light years. Many cluster-finding algorithms have been developed such as HOP [2], Friend of Friends [5], and Spherical Overdensity [6]. More recently, a group of astrophysicists gathered to compare and contrast the plethora of cluster-finders. Their results are summarized in [7].

Similarly, large void regions in the galactic distribution were first identified and studied in 1981. Several void-finding algorithms have been developed such as ZOBOV [8] and the Watershed Void Finder [9]. A systematic comparison of various void-finders is described in [10].

There are fewer algorithms which describe the one-dimensional filamentary collections of galaxies which string clusters together, and the two-dimensional walls or sheets such as the Sloan Great Wall. One method, entitled the Multiscale Morphology Filter (MMF) uses the eigenvalues of the hamiltonian to classify structures [11]. Another algorithm, DisPerSE, uses Discrete Morse Theory to identify clusters, walls, filaments and voids [4].

4 Discrete Morse Theory

In this section, we present Discrete Morse Theory (or, DMT) in brief and describe results from its application to the data set.

4.1 Overview

Morse theory provides a characterization of the topology (or, shape) of a space by studying a *Morse function* (or, a smooth underlying function) on that space. In particular, it provides a CW-decomposition of the space. To do so, Morse theory uses information about critical points of the smooth function. For a complete description of smooth Morse theory, see [12]. Since our data is a discrete, point-cloud distribution, we turn to the theory of Discrete Morse Theory (or, DMT), as outlined by Forman in [13]. DMT uses a discrete Morse function on a simpicial complex in order to identify the underlying topology, i.e., in particular, the CW decomposition, of the simplicial complex. This decomposition is also known as the Morse-Smale decomposition.

4.2 Methodology

The algorithm we employ to compute the Morse-Smale complex for the SDSS data is the divide-and-conquer algorithm developed by Gyulassy [14]. This variant of the DMT algorithm requires sample densities from a regular grid. To achieve this, we first assign a *density* to each point, where we define the inverse Voronoi volumes for a point as its density. We then interpolate this density everywhere through the DTFE algorithm [15]. We then sample a regular grid from the interpolated densities. This sample is then readily consumed by Gyulassy's algorithm to provide the critical points and Morse-Smale complex.

4.2.1 Interpolation

The particular DMT implementation selected for our data set required an underlying scalar function on a regular grid. Since point density is defined *a priori* only at the galaxy points, we need a way to interpolate the point density everywhere (i.e. including in empty regions). To do so, we considered two methods: (a) the Delaunay Tessellation Field Estimator (DTFE), and (b) Natural Neighbors. DTFE was selected as the most suitable method for our data. Both methods are described below.

- **DTFE**
  DTFE defines the interpolated density at a point to be the interpolation of densities on the vertices of the Delaunay triangle that contains that point, with respect to some Delaunay triangulation equipped with densities defined at all vertices.

  The purpose of this method is to reproduce a volume-covering reconstruction of a density field from a set of discrete data points sampling this field.

  The DTFE method comprises of the following three steps.

  1. Perform a Delaunay Tessellation on the data set. In 2D, this results in a triangulation, and in 3D, a tetrahedral tessellation.
  2. Estimate the density field at each data point of the set. This is done by defining the density as the inverse of the sum of the volumes of the *contiguous* cells (all Voronoi cells that share the data point as a vertex). Using the contiguous cell instead of the individual Voronoi cell of each data point will ensure that the mass is conserved.

     $$\hat{\rho}(x_i) = \frac{1}{\mathrm{Vol}\,(W_i)}$$

  3. Interpolate the density field. This is achieved by defining the density at any point as the weighted sum of the densities of the Delaunay triangle that contains that point, where the weights are defined simply as the respective Euclidean distance of the point to the vertices.

- **Natural Neighbor**
  Natural Neighbor interpolation (or Sibson's interpolation) has several advantages over simpler methods such as linear and spline interpolation because it can construct a smoother underlying function on the data set. The method first uses the Voronoi tessellation on the entire density field. Then, for every new point that was introduced, we can construct a "local" Voronoi cell of it without disturbing the neighborhood by constructing a perpendicular bisector to each segment that connects the new point and a data point in an adjacent Voronoi cell. Each neighboring Voronoi cell contributes a factor of its volume to the resulting new Voronoi cell. The interpolation at this new point will be obtained by adding up the weights of the neighboring cells with respect to the ratio of their volume contribution to the new cell.

  $$G(x, y) = \sum_{i=1}^{n} w_i f(x_i, y_i)$$

4.3 Results

Using this method, we obtain the critical points as well as the ascending and descending manifolds in the data set. However, due to the complicated nature of the data in three dimensions, we were unable to simplify and interpret the results usefully.

5 HOP

The method of HOP was first developed by Eisenstein and Hut (1997) [2]. The objective of HOP is to provide a simple way of grouping data points in a flexible and meaningful way. HOP is flexible enough to combine with many other related methods.

HOP operates on any point-set data with given neighboring (i.e. edge) information and an applicable one-to-one scalar function defined on each vertex.

We define neighboring information using the edges of the Delaunay Graph, and we used the inverse volume of the Voronoi cells in 3d to assign a scalar value to each vertex. This value may plausibly be interpreted as "local density": a vertex surrounded by many data

points will have small Voronoi cell volume and thus a high density value; conversely, a vertex in a void region will have a large Voronoi cell volume and, therefore, a low density value.

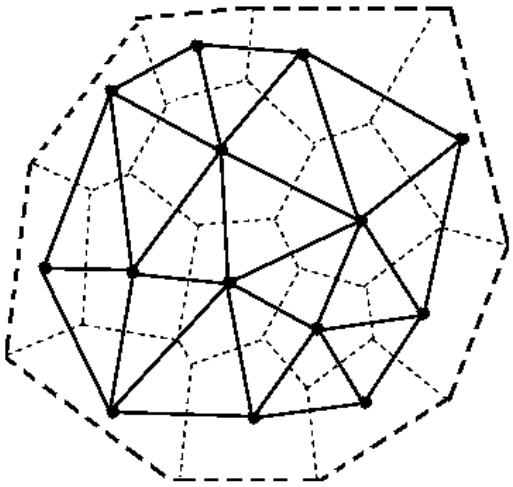


Figure 6: A valid HOP graph, with Delaunay edges (solid lines) for neighboring information and Voronoi cells (dotted lines) for inverse-volume density values.

Note that, although Voronoi volumes are not technically a one-to-one mapping, the random nature of our data distribution implies that there is almost certainly no two Voronoi cells of equal volume.

5.1 Algorithm

A ***HOP edge*** is an edge in the Delaunay triangulation which connects a vertex to the neighboring vertex with the highest point density value, if any higher-valued neighbor exists. The HOP procedure is described pictorally in figure 7: starting from any vertex (here circled in gray), consider all of that vertex's neighbors (circled in green), and then *hop* to the highest valued vertex neighbor. Recall that, in our case, "highest valued" is equivalent to the vertex with the smallest Voronoi cell (here colored blue).

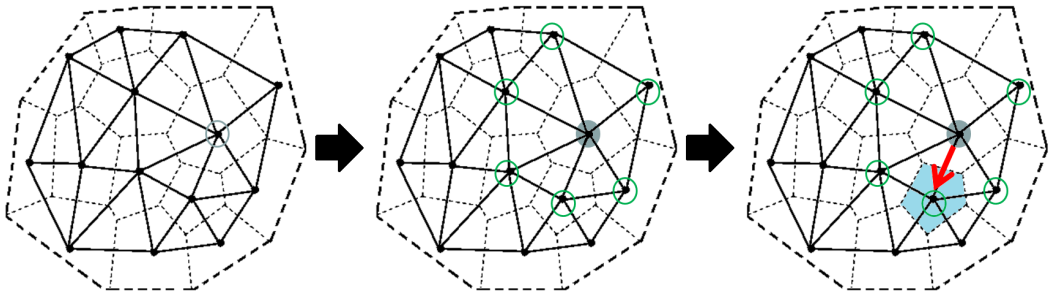


Figure 7: The HOP procedure.

If a vertex has no higher-valued neighbor to hop to, this vertex is then considered a maximum (shown in red). It is also equivalent to say: a maximum is any vertex that is higher-valued than all of its neighbors (i.e. has a smaller Voronoi cell).

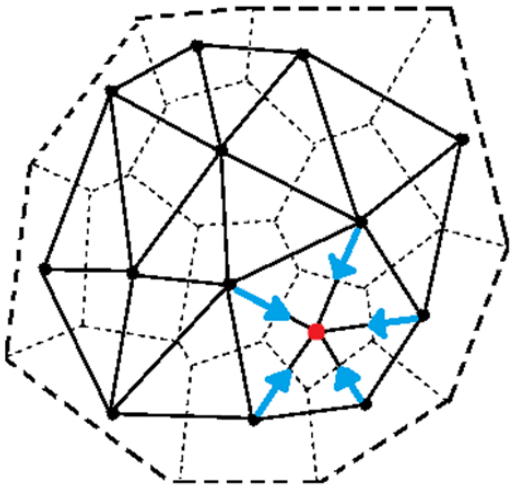


Figure 8: A HOP maximum.

In this manner, we can follow directed chains of HOP edges, called ***HOP paths***, and identify maxima at the termination of these paths.

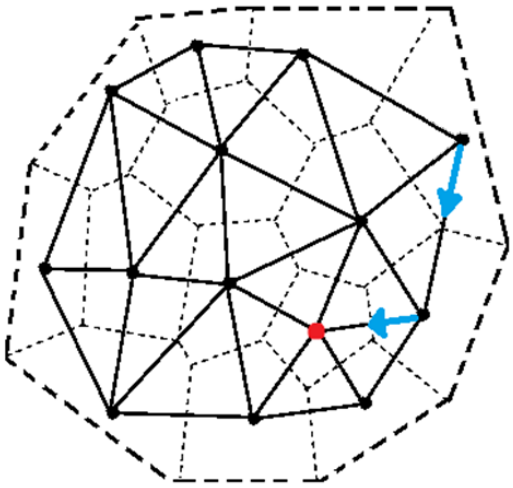


Figure 9: A HOP path, indicated by blue arrows, terminating at a HOP maximum in red.

Note: We can also reverse the algorithm in order to hop to the *lowest*-valued neighbor; identifying minima at the termination of these reverse-HOP paths.

5.2 Results

We can extract much useful information from this simple algorithm. ***Max classes*** are the set of vertices whose HOP paths all terminate at the same maximum, including the maximum (two max classes shown here in green and blue). This gives us a simple and elegant way of grouping related data points.

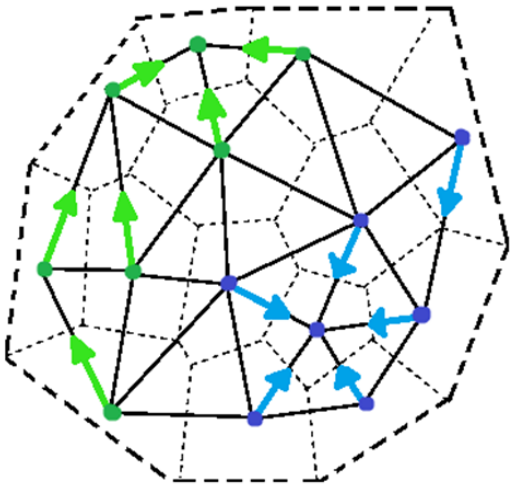


Figure 10: Two different HOP max classes, colored green and blue.

Max classes can be analyzed to obtain such information as (a) ***boundary points***: vertices which lie on the edges of their respective max classes, shown in orange; (b) ***boundary edges***: Delaunay edges which connect two points on the boundary of the same max class, shown in red; (c) ***boundary graphs***: the combined set of a single max classs boundary points and boundary edges; and (d) ***bond edges***: Delaunay edges that connect two vertices in different max classes, shown in purple.

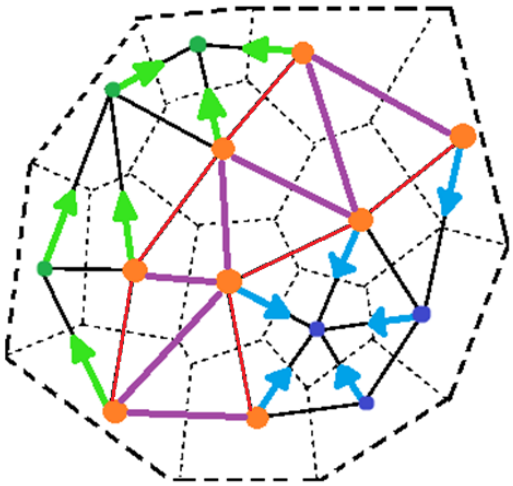


Figure 11: A HOP boundary graph is the collection of boundary points (orange) and bond edges (purple). The boundary edges are indicated in red.

These pieces are useful in providing information for further post-processing of the data, such as merging groups, classifying structures, etc. A formal treatment follows.

5.3 A graph-theoretic treatment

Denote the set of points in the raw data $\mathcal{P}$. For a point $p \in \mathcal{P}$, let $V(p) :=$ "the Voronoi cell for p".

Definition 1. Let $d_1, d_2 \in \mathcal{P}$. The two associated Voronoi cells $V(d_1)$ and $V(d_2)$ are ***adjacent*** if they have non-empty intersection. We also may say that $V(d_1)$ and $V(d_2)$ are ***neighboring*** cells, or that they are ***neighbors***.

One can clearly see from the geometric definition of Voronoi cells, that two cells can *only* intersect on their boundary. In $\mathbb{R}^3$ this boundary intersection is a polygonal face and in $\mathbb{R}^2$ the common boundary for two adjacent Voronoi cells is a line segment.

Because each Voronoi cell in our work is a convex polygonal area in two dimensions or a convex polyhedral region in three dimensions, the area or volume of $V(p)$ is a well defined real number. That is, of course, except for the Voronoi cells on the boundary of the data space which have infinitely large area or volume. These Voronoi cells have a vertex at the point at infinity. These cells could be assigned the "value" ∞ if the volume is infinite, but we choose to throw out these cells and only work with the cells that have finite area or volume. So we will not consider them. Thus we have a function $Vol : \mathcal{P} \to \mathbb{R}^+$, defined,

$$\text{Vol}\,(p) = \text{“the volume (or area) of } V(p)\text{”}$$

In practice, when we are using SDSS data with full floating point precision, this function is one-to-one. So no two Voronoi cells have the same volume. If it were not one-to-one, then we could easily perturb the data or the volumes by an arbitrarily small ϵ and make it one-to-one without altering the structure of the data. So we will assume throughout that the Voronoi volume function is one-to-one. We define a density function $\rho : \mathcal{P} \to \mathbb{R}^+$ by

$$\rho(p) = \frac{1}{\text{Vol}\,(p)}. \tag{2}$$

Since Vol is one-to-one, then ρ is clearly one-to-one. This will be the scalar function we use for *HOP*. Whenever we refer to "the density of p" then we are referring to value $\rho(p)$.

Recall that the Delaunay Tessellation of our data set is a three dimensional simplicial complex which tessellates the data space. Thus it is a collection of closed tetrahedra that intersect only on their boundary and the union of the collection of tetrahedra is the entire data space including the convex hull boundary. In this section we will be interested in the graph determined by this simplicial complex.

Definition 2. The ***Delaunay graph***, $\mathcal{D}$, is the 1-skeleton of the Delaunay simplicial complex determined by the data set. If $\mathcal{P} \subset \mathbb{R}^2$, then it is the 1-skeleton of a triangulation of the data space. If $\mathcal{P} \subset \mathbb{R}^3$, then it is the 1-skeleton of the complex of tetrahedra. In either case, the vertices of $\mathcal{D}$ are the points in $\mathcal{P}$, and the edges, $\mathcal{E}$, are the 1-cells in the Delaunay Complex. Thus the graph is $\mathcal{D} = (\mathcal{P}, \mathcal{E})$.

Recall that a Voronoi Cell in $\mathbb{R}^1$ is a closed line segment, in $\mathbb{R}^2$ it is a closed, convex polygon, and in $\mathbb{R}^3$ it is a closed, convex polyhedral region (ignoring the infinite boundary cells). The Voronoi Diagram is the union of the cells. We will be mostly interested in the Voronoi Cells in $\mathbb{R}^3$ determined by a finite set of data points. But we define the *graph* of a Voronoi cell and Voronoi Diagram for two and three dimensions here.

Definition 3. Let $\mathcal{P}$ be a finite subset of $\mathbb{R}^2$ or $\mathbb{R}^3$. A ***Voronoi cell graph*** for a point $p \in \mathcal{P}$ is the set of vertices and edges that comprise the 1-skeleton of the Voronoi cell for p. We denote this graph as $\mathcal{V}_p$. In $\mathbb{R}^2$, $\mathcal{V}_p$ is the entire boundary of the convex Voronoi cell polygon. In $\mathbb{R}^3$, $\mathcal{V}_p$ is the one dimensional wire-frame that determines the boundary of the Voronoi cell for d. In either dimension, $\mathcal{V}_p = (V, E)$ where V is the set of vertices and E is the set of edges for the cell. The ***complete Voronoi graph***, is the union of all the Voronoi cell graphs for all $p \in \mathcal{P}$. That is,

$$\mathcal{V} = \bigcup_i V(p_i),$$

where i indexes over the data points.

We may refer to the graph $\mathcal{V} = (V, E)$ and it is understood that V is the union of the vertices of the individual cells and similarly for E. Also take care to note that $\mathcal{V}_p$ refers to the *Voronoi cell graph* for p, but $V(p)$ is the *geometric Voronoi cell* for p. (So $V(p)$ is a closed polyhedral area in $\mathbb{R}^2$ or a closed polyhedral volume in $\mathbb{R}^3$.)

As we stated earlier, the geometric definition of Voronoi cells implies that if two Voronoi cells have non-empty intersection, then they must intersect at their boundary (and only the boundary). This fact, along with the dual nature of $\mathcal{D}$ and $\mathcal{V}$ implies the following: a pair of data points $d_1, d_2 \in \mathcal{D}$ are adjacent if and only if their Voronoi cells $V(d_1)$ and $V(d_2)$ have non-empty intersection, which is true if and only if $V(d_1)$ and $V(d_2)$ are neighboring Voronoi cells. So discussing neighboring vertices in $\mathcal{D}$ is equivalent to discussing neighboring

Voronoi cells in the Voronoi complex.

For the remaining discussion we will refer to "forward *HOP*," which hops to neighbors of higher density, as HOP^+. We will denote "backward *HOP*," which hops to neighbors of lower density, as HOP^-.

Because $\mathcal{P}$ is a finite set, every data point, d, has a finite number of neighbors in $\mathcal{D}$. Let the neighborhood of d be $N(d) = \{n_1, n_2, \ldots, n_k\}$. Since the density function ρ is injective, then

$$\max\big(\rho(n_1), \rho(n_2), \ldots, \rho(n_k)\big) \tag{3}$$

exists and is unique. Let's say n_i is the neighbor with the maximum density. If $\rho(d) > \rho(n_i)$ then the density of d is greater than all of it's neighbors and so d is called a ***HOP maximum***. Otherwise, $\rho(d) < \rho(n_i)$ in which case we call vertex n_i the ***HOP$^+$ neighbor*** for d. We can similarly exchange "max" for "min" in equation (3) and with a similar argument obtain the analogous definitions for HOP^-. A ***HOP minimum*** is a point, d, whose density is lower than all of it's neighbors. And, if d is not a minimum, then d has a unique minimum density neighbor which we call a ***HOP$^-$ neighbor***.

We wish to describe *HOP* in the language of graph theory. Because the *HOP* algorithm hops from vertex to vertex in uniquely defined steps, the resulting *HOP* graph will be a directed graph. The vertices of the *HOP* graph will be $\mathcal{P}$ of course. And the edges are defined in the following way.

Definition 4. Let $u \in \mathcal{P}$ be a data point which is not a maximum. A ***HOP$^+$ edge*** is a directed edge from u to it's unique HOP^+ neighbor v. We denote this directed edge $u \xrightarrow{+} v$. Similarly, let $x \in \mathcal{P}$ be a data point that is not a minimum, then a ***HOP$^-$ edge*** is a directed edge from x to it's unique HOP^- neighbor y. We denote this directed edge $x \xrightarrow{-} y$.

Now, with the aid of the above discussion, we can state a brief outline for the HOP^+ algorithm:

Algorithm 5 (HOP^+)**.** We begin by choosing a particular initial vertex $d_0 \in \mathcal{P}$. We call this point the "current data point." Then execute the following:

1. If the current data point is a HOP^+ maximum, then stop. If it is not a maximum then proceed to step 2.
2. The current data point is not a max, so hop to its unique HOP^+ neighbor d_1. Now make d_1 the new "current data point" and go to step 1.

The obvious analogous algorithm can be defined for HOP^-.

Lemma 6. *If $\mathcal{P}$ is a finite set of data points, then $\mathcal{P}$ has a HOP minimum and a HOP maximum.*

Proof. Since $\mathcal{P}$ is a finite set and since our density function ρ is injective, then $\mathcal{P}$ must contain a vertex with global maximum density and a vertex with global minimum density. Thus there exists at least one *HOP* maximum and at least one *HOP* minimum for any finite set $\mathcal{P}$.

□

Since the steps in the $HOP^\pm$ algorithm are well defined, the algorithm produces a sequence of vertices of length at least one (the initial vertex). Also note that each time the HOP^+ algorithm reaches step 2, it hops to a higher density neighbor (similarly HOP^- hops to a lower density neighbor). Based on these two facts, we have the following lemma.

Lemma 7. *Let d_0 be the initial vertex in the HOP^+ algorithm and let the sequence $(d_0, d_1, \ldots, d_k)$ be the sequence consisting of d_0 followed by k iterations of the HOP^+ algorithm for $k \geq 0$. Then,*

$$\Big(\rho(d_0), \rho(d_1), \ldots, \rho(d_k)\Big)$$

is a monotonically increasing sequence. Therefore the $k+1$ vertices in the sequence are unique.

Proof. The monotonicity of the sequence follows from the definition of the algorithm. The uniqueness of the $k+1$ entries in the sequence follows from the fact that ρ is injective. □

A similar lemma clearly holds for HOP^- but the sequence would be monotonically decreasing.

Theorem 8. *The HOP^+ and HOP^- algorithms are well defined and they must terminate regardless of the choice of initial point.*

Proof. We prove this for HOP^+. The proof for HOP^- is similar. To prove this we simply need to show that the sequence of vertices determined by the algorithm must reach a maximum in finitely many steps. First note that lemma 7 shows that the sequence of vertices determined by the HOP^+ algorithm cannot have a repeated vertex. And lemma 6 shows that at least one maximum exists in $\mathcal{P}$. Therefore, since $\mathcal{P}$ is a finite set of points, the sequence of vertices determined by the HOP^+ algorithm must reach some maximum in finitely many steps. □

The above results allow us to define the following.

Definition 9 (HOP paths)**.** A ***HOP^+ path*** beginning at an initial vertex d_0, is a sequence of vertices in $\mathcal{P}$ determined by the HOP^+ algorithm along with the HOP^+ edges determined by the sequence. Thus, a HOP^+ path is of the form:

$$d_0 \xrightarrow{+} d_1 \xrightarrow{+} d_2 \xrightarrow{+} d_3 \xrightarrow{+} \cdots \xrightarrow{+} d_k$$

. This HOP^+ path is denoted $d_0 \nearrow d_k$. We define a HOP^- path in the obvious similar way and denote a HOP^- path $d_0 \searrow d_k$.

Note that if d_k is not a maximum, then $d_0 \xrightarrow{+} d_k$ can be extended further by the HOP^+ algorithm. When this HOP^+ path terminates, the termination point will be a maximum.

Theorem 10. *Let $u \in \mathcal{P}$ be the initial point for the $HOP^{\pm}$ algorithms. Then the HOP^+ algorithm will determines a unique HOP^+ path which terminates at a maximum, M. Thus the path $u \xrightarrow{+} M$ cannot be extended and M is the unique maximum vertex for u. Similarly HOP^- gives us a unique path $u \xrightarrow{-} m$ which terminates at the unique minimum vertex for u.*

Proof. Let $u_0 \in \mathcal{P}$.

Case 1. u_0 has no higher-valued HOP^+ neighbors, in which case u_0 is its own unique maximum vertex and the HOP^+ path is unique (simply $u_0 \nearrow u_0$).

Case 2. u_0 has a unique HOP^+ neighbor u_1 (uniqueness shown earlier).

Induction: for all u_k, $k \in \mathbb{Z}$, u_k is either its own unique maximum or it has a unique HOP^+ neighbor u_{k+1}.

By ***theorem 8***, HOP^+ must terminate at u_j for some $j \in \mathbb{Z}$.

$\therefore$ Because every step is unique, and the HOP^+ path eventually terminates at some u_j, the HOP^+ path $u_0 \nearrow u_j$ is unique and u_j is the unique maximum vertex for u_0.

□

Note that if the initial point u is a maximum to begin with, then u is its own representative maximum and its HOP^+ path, $u \nearrow u$ has length zero (since it doesn't have an edge). A similar comment holds if u is a minimum. Thus *every* point in $\mathcal{P}$ is associated with a unique minimum and maximum.

The uniqueness of these paths allows us to define a relation in the following way.

Definition 11. We declare $u \sim_+ v$ if the HOP^+ path for u and v both terminate at the same maximum M, that is, $u \nearrow M$ and $v \nearrow M$. Similarly we define $u \sim_- v$ if $u \searrow m$ and $v \searrow m$.

Proposition 12. *The relations defined above are equivalence relations which partition the data.*

This allows us to define the following.

Definition 13. Let M be a maximum and let m be a minimum. The ***max-class*** $\widetilde{M}$ for M is the set of all points whose HOP^+ path terminates at M. Thus it is the equivalence class

$$\widetilde{M} = \{u \in \mathcal{P} \,:\, u \nearrow M\}.$$

Similarly, the ***min-class*** for m is the equivalence class

$$\widetilde{m} = \{u \in \mathcal{P} \,:\, u \searrow m\}.$$

Theorem 14. *Let M be a max. The union of all of the HOP^+ paths which terminate at M is a directed graph, G, which is weakly connected. Furthermore, the underlying graph for G is a spanning tree for $\widetilde{M}$.*

Theorem 15. *Let M be a max and m be a minimum in $\mathcal{P}$. Then the induced subgraph, $\mathcal{D}[\widetilde{M}]$, is connected. Also, $\mathcal{D}[\widetilde{m}]$ is connected.*

Proof. Let $u, v \in \mathcal{D}[\widetilde{M}]$, and let $\sim$ be the connectedness relation.

By definition of $\mathcal{D}[\widetilde{M}]$ membership, there exist HOP^+paths $u \nearrow M$ and $v \nearrow M$, and so $u \sim M$ and $v \sim M$.

Recall that, by ***proposition 12***, connectedness is an equivalence relation.

Since $u \sim M$ and $v \sim M$, then, by symmetry and transitivity, $u \sim v$.

$\therefore \mathcal{D}[\widetilde{M}]$ is connected.

Similar arguments hold for $\mathcal{D}[\widetilde{m}]$.

□

We call $\mathcal{D}[\widetilde{M}]$ a ***max-class graph*** and $\mathcal{D}[\widetilde{m}]$ a ***min-class graph***. Remember that $\widetilde{M}$ and $\widetilde{m}$ are only the vertices in the respective class. But $\mathcal{D}[\widetilde{M}]$ includes all the edges in $\mathcal{D}$ which have endpoints in $\widetilde{M}$.

For our work we will need to classify all vertices and edges in $\mathcal{D}$. The neighborhood of u in $\mathcal{D}$, is the set of all vertices in $\mathcal{D}$, which are adjacent to u, denoted $N(u)$. The vertices in $N(u)$ may or may not be in the same max-class as u.

Definition 16. Let u be a vertex in the max-class $\widetilde{M}$ with neighborhood $N(u)$. We call u an ***interior point*** for $\widetilde{M}$ if every vertex in $N(u)$ is also in $\widetilde{M}$. Otherwise u is a ***boundary point***. The same definitions apply for a min-class.

So interior points are adjacent only to vertices in the same max-class or min-class, whereas boundary points have an adjacent vertex that is a member of a different class. We will be very concerned with the boundary points of a max or min-class.

This definition in a certain sense deals with the boundary of the data space, which we often want to disregard. If a point p is on the boundary of the data space, and all of its adjacent vertices are in the same max class as p, then it is an interior point for that max-class (even though it is on the boundary of the data space). Our definition of boundary point emphasizes the boundary *between* max/min-classes, not the boundary of the data space itself.

Definition 17. A ***bond edge*** is an edge $e \in \mathcal{D}$ that is incident to a pair of vertices u and v which are members of different max classes. Thus $u \nearrow M_i$ and $v \nearrow M_j$ and $i \neq j$. We call the pair, $\{u, v\}$, a ***boundary pair*** which determines the Delaunay edge between them: $e = u \leftrightarrow v$. We say that e is a bond edge for $\mathcal{D}[\widetilde{M}_i]$ and e is also a bond edge for $\mathcal{D}[\widetilde{M}_j]$ since e is attached to a vertex in each max-class. Since $\mathcal{D}[\widetilde{M}_i]$ and $\mathcal{D}[\widetilde{M}_j]$ share a bond edge e, we say that M_i is a ***neighboring maximum*** to M_j (or that they are ***max-neighbors***) and that their max-classes border each other, as illustrated below in figure 12.

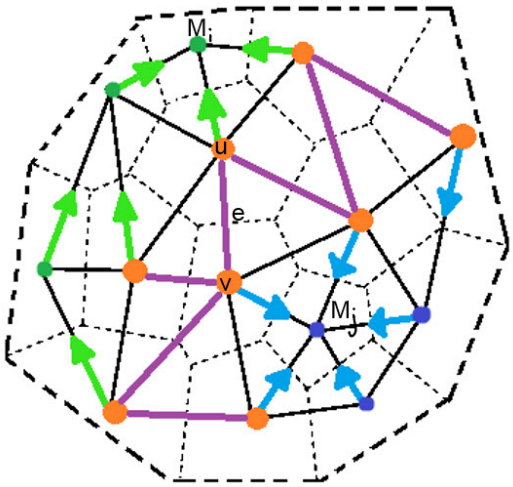


Figure 12: The point u is a boundary point of $\mathcal{D}[\widetilde{M}_i]$ and v is a boundary point of $\mathcal{D}[\widetilde{M}_j]$. The pair $\{u, v\}$ is a boundary pair which determines the bond edge e.

This choice of the term "bond" is intentional. If we take the union of the vertices in two neighboring max-classes $A = \widetilde{M}_i \cup \widetilde{M}_j$, then the induced subgraph $\mathcal{D}[A]$ is the subgraph of the Delaunay graph consisting of the max-class graph for M_i as well as the max-class graph for M_j along with all of the bond edges they share. This set of bond edges in this case, is a bond in the graph theory sense; it is a minimal non-empty edge cut (see definitions 72, 73, and 74 in Appendix B). Therefore, the set of bond edges, F, is the minimal set of edges we need to delete from A in order to disconnect A in such a way that we completely separate the two max classes into separate components. Thus $A - F = \mathcal{D}[\widetilde{M}_i] \cup \mathcal{D}[\widetilde{M}_j]$, A is now disconnected, and these two max-class graphs are the two connected components of $A - F$. Similar comments hold for minimums.

Definition 18. Let $e = u \leftrightarrow v$ be an edge in the max-class graph $\mathcal{D}[\widetilde{M}]$. We define e to be a ***boundary edge*** for $\mathcal{D}[\widetilde{M}]$ if e is an edge in some triangle, t, in the Delaunay Graph, such that t has a vertex, w, which is not in $\widetilde{M}$. Otherwise e is an ***interior edge***.

Since $e \in \mathcal{D}[\widetilde{M}]$, then u and v must be in $\widetilde{M}$. Also note that the vertices of the triangle t are u, v, and w. Since $w \notin \widetilde{M}$, then $u \leftrightarrow w$ is a bond and $v \leftrightarrow w$ is a bond. Thus, the endpoints of e *must* be boundary points of $\widetilde{M}$. This definition also prevents e from being an edge that cuts through the middle of the max class graph. Because if it did, then every triangle containing e would have *all* of its vertices in $\widetilde{M}$.

Thus, when we are considering HOP^+, every edge in $\mathcal{D}$ is either a bond edge connecting two max-class graphs, a boundary edge for some max-class graph, or an interior edge for some max class graph. The same holds for HOP^-. This allows us to define the boundary of a max or min-class graph.

Definition 19. Let M be a HOP maximum with max-class graph $\mathcal{D}[\widetilde{M}]$. The ***boundary graph*** of $\mathcal{D}[\widetilde{M}]$, denoted $\partial\,\mathcal{D}[\widetilde{M}]$, is a graph made up of all the boundary vertices and boundary edges for $\mathcal{D}[\widetilde{M}]$.

Since every boundary edge for $\widetilde{M}$ has endpoints that are boundary points for $\widetilde{M}$, then $\partial\,\mathcal{D}[\widetilde{M}]$ is well defined.

Proposition 20. *A boundary graph cannot have any isolated vertices.*

5.4 Benefits and Drawbacks of HOP

5.4.1 Benefits

One of the biggest benefits to using HOP is its algorithmic simplicity. This makes it easy to both understand and compute, without ever showing much serious complication. Usually

any unintuitive max classes are a consequence of the choice of density function or neighboring information. On this note, however, HOP allows a lot of room for pre- and post-processing of data in order to tailor the program toward individual project pursuits. In this sense, HOP can be used more like a tool then a full algorithmic approach.

Other benefits include the fact that HOP is monotonically increasing, with guaranteed and unique convergence. It is parameter-free in the sense that, once input choices are decided, the algorithm is objective. HOP can also be run on data of any dimension. Lastly, the fact that HOP is easily reversed to find minima and other converse information allows for a lot of information to be simply and flexibly extracted.

5.4.2 Drawbacks

There are a couple inherent features of HOP that may be undesirable in certain contexts. HOP is extremely "local" in the sense that a data point's HOP path is entirely determined by only its immediate neighbors. Small outliers in data can thus affect the HOP outcomes of all neighboring points; overall density measures on the data space as a whole can be distorted in this way. This can potentially be addressed through different choices of neighboring information or scalar function. On this note, however, HOP is entirely reliant on the initial choice of input information: this sensitivity to neighboring information and scalar functions can lead to some unintuitive and undesirable max class results if the input choices are not perfect (and perfection is often hard to achieve!). On the other hand, HOP does allow much room for post-processing steps to try addressing some of these undesirable situations.

6 Variants of HOP

A peculiar feature of results obtained from HOP was that it performed poorly in some regions where the human eye could clearly identify structure – often, HOP returned a plethora of max classes within structures. It effectively splintered "whole" structures into small classes, without any obvious way of piecing the parts back together. This motivated our search for useful ways to modify HOP.

Before our development of post-HOP processing methods, which will be detailed later in this report, we did come up with many "pre-HOP" processes that were intended to address some of these problems we encountered in our particular data. This usually entailed varying our choice of neighboring information and/or scalar function in order to better suit our goals.

6.1 EHOP

Our choice of inverse Voronoi volume as a scalar function for point density led to extreme local sensitivity and a large number of max classes containing few points. In a desire to smooth local density fluctuations, we considered a modification of the original HOP algorithm. In this section, we describe a method that we developed along these lines – called *EHOP*.

For a point d, we are already aware that its neighborhood is $N(d) = \{n_1, n_2, ..., n_k\}$ where n_i is defined as a Voronoi neighbor of point d and k is the number of neighbors of d. Recall in HOP+ (and analogously in HOP-), we would then consider $\max\big(\rho(n_1), \rho(n_2), \ldots, \rho(n_k)\big)$ to determine the HOP+ neighbor for d (if $\rho(d) < \rho(n_i)$, where n_i is the HOP neighbor; otherwise d is a local maximum).

In this variant of HOP, we expand the set of HOP neighbors for each point by including each point's neighbors as well as its neighbors' neighbors.

We write the neighborhood of a point n_i (which is neighboring point d) as $N(d) = \{n_{i,1}, n_{i,2}, \ldots, n_{i,k_i}\}$ and our extended neighborhood of our point d,

$$E(d) = N(d) \cup \bigcup_{i=1}^{k} N(n_i). \tag{4}$$

As in regular HOP, we may now consider

$$\begin{aligned}\max\big(&\rho(n_1),\rho(n_2),\ldots,\rho(n_k),\\&\rho(n_{1,1}),\rho(n_{1,2}),\ldots,\rho(n_{1,k_1}),\\&\rho(n_{2,1}),\rho(n_{2,2}),\ldots,\rho(n_{2,k_2}),\\&\rho(n_{k,1}),\rho(n_{k,2}),\ldots,\rho(n_{k,k_k})\big)\end{aligned} \tag{5}$$

and once again, point d is considered a maximum if its density is larger than any of its extended neighbors, or its HOP+ neighbor is the n_i or $n_{i,j}$ that has the highest density.

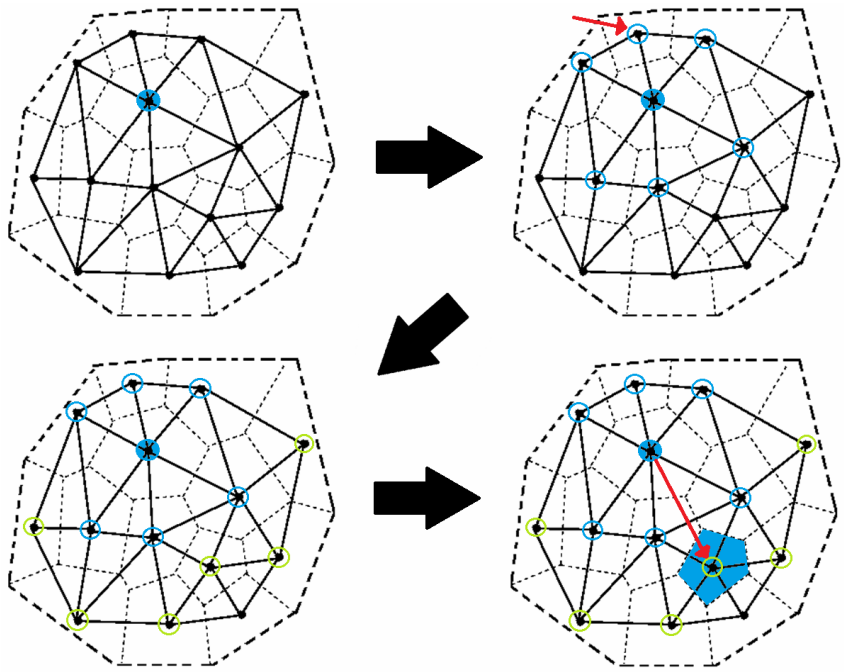


Figure 13: The EHOP procedure. Top left: The initial point, p, is highlighted. Top right: Points in $N(p)$ are circled in blue. The red arrow indicates the HOP+ neighbor of p. Bottom Left: Points in $E(p)$ are circled in green. Bottom right: Point p finds the point with the highlighted cell as its EHOP neighbor

The results of EHOP differ in various ways to those of regular HOP. First, it is common to find that a point's HOP neighbor differs from its EHOP neighbor as a direct result of EHOP's process. This implies that max classes differ between HOP and EHOP. The most noticeable discrepancy between HOP and EHOP is that there are significantly fewer max classes in EHOP than there are in HOP. This is because each point has a farther range in terms of the points they are able to HOP to. Conversely, this implies that a given maximum in EHOP has many more neighbors than a maximum in HOP, so more points are likely to converge to one maximum.

While EHOP yields fewer maxes than HOP, points that are maxima in EHOP must also be maxima in HOP. This is due to the fact that a max in EHOP still has the highest density in its local region. If a point p is a maximum in EHOP, but not in HOP, p would have a HOP+ neighbor q in its set of neighbors, $N(p)$ such that $\rho(q) > \rho(p)$. However, since q would also be in the extended neighborhood, $E(p)$, that would imply that p would hop to q in EHOP as well, which contradicts that p would be a maximum in EHOP. Thus, a maximum in EHOP must be a maximum in HOP as well.

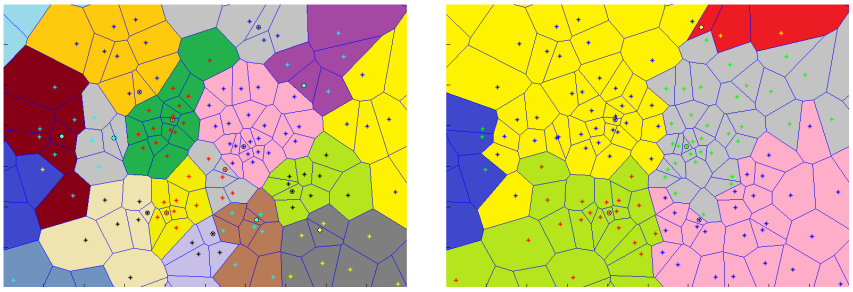


Figure 14: Colored max classes for HOP (left) and EHOP (right) on a 2d slice of the SDSS data. On another 2d slice of the data set, HOP yielded 439 maxima while EHOP yielded 219.

Though EHOP succeeds in smoothing our galaxy data by reducing the amount of max classes, it also produces some questionable results. One type of notable phenomena in regular HOP is the occurrence of points that appear to HOP across void-like regions due to the simplicity of the Voronoi neighboring information it uses. Extending the range through EHOP seems to exacerbate this questionable behavior.

For example, take points point p, q, r and s where q has the highest density in $N(p)$, r has the lowest density in $N(p)$ and s has the highest density in $E(p)$. Also, assume that s is not in $N(p)$, but it is in $N(r)$. Point p would hop to q through the HOP algorithm but p would hop to s through the EHOP algorithm. This is would be considered questionable behavior primarily because in EHOP, p would hop to s through passage of r, which has low density and a Voronoi cell with a relatively large volume. The following example demonstrates how a point can HOP through a low density cell which would intuitively indicate a void-like region.

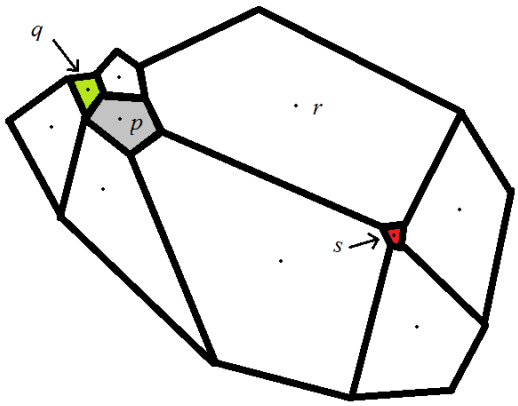


Figure 15: In HOP, p would HOP to q. In EHOP, p hops to s across the Voronoi cell of r, which appears to be a high-volume, low-density void-like region.

6.2 Smooth HOP

Since HOP depends strongly on the density function it uses, changing the density function can make a dramatic difference in the way HOP groups galaxies. The motivation behind the next variant, Smooth HOP, is to use a density function that would take into consideration the densities of its neighbors as well as itself to address the hyper-local nature of HOP. A logical density function to use that takes its local area's densities into account was to model it off of the fundamental physics density formula

$$\text{density} = \frac{\text{total mass}}{\text{total volume}}. \tag{6}$$

For each point p, the new density function would consider the point's mass and volume as well as those of all of its Voronoi neighbors, which are $N(p) = \{n_{i,1}, n_{i,2}, \ldots, n_{i,k_i}\}$. We

continue to use the assumption that the mass of each data point is 1. Thus, the total mass in this formula is $1 + k$ where k is the number of neighbors of p. Similarly, the total volume in the new formula would simply be the sum of the Voronoi volume of the point p and those of all its neighbors. Thus, our equation is

$$\rho(p) = \frac{1 + k}{\text{Vol}(p) + \sum_i \text{Vol}(n_i)} \tag{7}$$

This equation implies that the densities of all the points in $N(p)$ are equally important to the density of point p. To allow for variability, the equation may be further altered by attaching a weight parameter $a \in \mathbf{R}$ between 0 and 1 to both the mass and volume of the neighbors in the function. The parameter, a, may be varied to emphasize or understate the value of a point's neighbors while calculating density.

$$\rho(p) = \frac{1 + ak}{\text{Vol}(p) + a\sum_i \text{Vol}(n_i)} \tag{8}$$

If $a = 0$, the density function degenerates to $\rho(p) = 1/\text{Vol}(p)$, which is the original density function that does not take its neighbors' densities into consideration at all. If $a = 1$, point p's neighbors are considered equally with point p in determining the density at p.

We refer to the application of HOP using the new density function as Smooth HOP. Instead of comparing the densities of each individual point, the algorithm determines its results by comparing a measure that resembles the density of each individual point's local neighborhood. The contextualization of density effectively smooths the data since the the function is less sensitive to the variance of individual points, mitigating the effect of the algorithm's local nature.

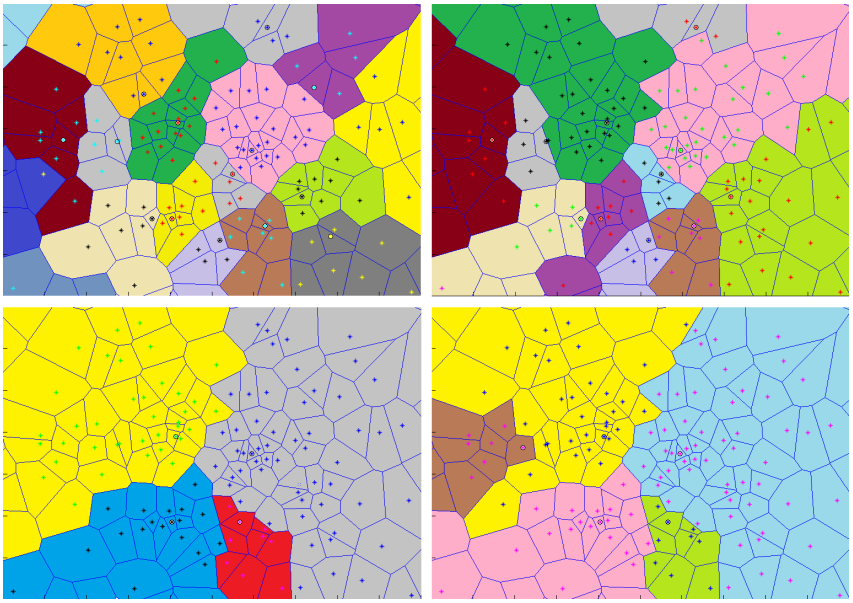


Figure 16: From left to right, top to bottom: Weight parameter a = 0, # max classes = 439; a = .15, # max classes = 355; a = .75, # max classes = 237; a = 1, # max classes = 224. Pictured here is a flattened 2d subset of the data illustrating the effect of varying the weight parameter on the size and shape of max classes.

The fundamental discrepancy between the results of regular HOP and Smooth HOP is the fact that it is very possible that a point's HOP neighbor is not the same as its Smooth HOP neighbor. This is a direct consequence of the change in density function. From this, it follows that it is possible that a point that is considered a maximum in HOP is not considered a maximum in Smooth HOP and vice versa. As a result, the topology, or shape, of the results of each algorithm may be different, where the extent of the difference depends on the weight parameter, a.

As the parameter a increases, the density function relies increasingly on the magnitude of the density of a point's neighbors. When implemented, the data yielded fewer max classes as a increased, indicating that the underlying density function for the HOP algorithm makes an impact on the results of the algorithm.

Thus, Smooth HOP is an effective treatment for galaxy data as it uses a density function that contextualizes a point's density with the density of its neighbors. As a result, it smooths the results of HOP somewhat depending on the weight parameter a, which determines how much the density formula considers a point's local neighborhood in calculating its density. Since the implementation of Smooth HOP only involves changing the density function, Smooth HOP can easily be implemented in any variant of HOP is compatible with any sort of post-processing method. The primary drawback for Smooth HOP is the fact that it requires a non-trivial parameter which may corrupt the simplicity of the HOP algorithm.

6.3 "JHOP" Method in 3D

JHOP is a method to partition not only the data points into max and min classes, but the data space itself. It does this by assigning each non-facet element of a simplicial complex (which itself effectively partitions the data space, excluding boundaries) to a particular HOP max class. It also attempts to potentially find critical simplices in ways that regular HOP might not be able to. This was done in semblance of Discrete Morse Theory, as previously described above and in Appendix A.

Note that, similar to the original HOP, the JHOP algorithm is easily run in both increasing ($JHOP^+$) and decreasing ($JHOP^-$) fashion.

Here, we assume a Delaunay Triangulation of the data with a well-defined density value at each vertex. Other neighboring techniques and scalar functions work similarly. Also, note that this method is perfectly applicable to 2D and 3D.

The basis of the JHOP algorithm is to assign each Delaunay tetrahedron to the max/min class of its densest vertex. In essence, the gradient of the entire tetrahedral interior is one big flow line toward the maximal-valued / minimal-valued vertex. One can also construct "JHOP paths", analogous to HOP paths, that provide information similar to the various HOP results. Described below is the algorithm to construct a single JHOP path.

6.3.1 $JHOP^+$: Max Classes (Descending Manifolds) and 2-Saddles

1. For each tetrahedron, identify the vertex of highest density (V_{dense}).
2. For each tetrahedron, pair the triangle face that is opposite V_{dense} with the tetrahedron itself. This will be defined as the "**gradient pair**".
3. Constructing a gradient path: From any tetrahedron, start from its respective V_{dense}, and HOP along a 1-segment to V_{dense}'s highest-valued neighboring vertex (V_{next}). All cofacent tetrahedra of this 1-segment (tetrahedra containing the 1-segment as an edge) are then added to the gradient path.

   Note: by construction, these added tetrahedra will contain V_{next} as their respective V_{dense}. Hence, all included tetrahedra do contain a gradient that points to V_{next}, so no ambiguity of JHOP paths will result.
4. Identifying critical simplices:
   - A maximum vertex V_{max} is defined when all cofacent tetrahedra have $V_{dense} = V_{max}$ (I.e. the gradient vectors of all cofacent tetrahedra point to V_{max}). The gradient path thus ends at this point.
   - For all other cases, a critical face (triangular 2-simplex) is any face that is a member of two gradient pairs. This will identify both minima and 2-saddles.
     - To differentiate between these minima and 2-saddles, there may be explicit ways to categorize based on the values of surrounding local vertices. However, do note that running $JHOP^-$ will return definite minima; our main concern here is the identification of 2-saddles.

5. You should now have a set of increasing gradient paths that partition the set of tetrahedra into classes of common V_{max}, which can hence be considered a max class (descending manifold).

6.3.2 *JHOP*$^{-}$: Min Classes (Ascending Manifolds) and 1-Saddles

1. For each tetrahedron, identify the vertex of least density (V_{small}).

2. For each tetrahedron, pair the triangle face that is opposite V_{small} with the tetrahedron itself. This will be defined as the "**reverse gradient pair**".

3. Constructing a reverse gradient path: From any tetrahedron, start from its respective V_{small}, and HOP along a 1-segment to V_{small}'s least-valued neighboring vertex (V_{next}). All cofacent tetrahedra of this 1-segment (tetrahedra containing the 1-segment as an edge) are then added to the reverse gradient path.

   Once again, note that these added tetrahedra will contain V_{next} as their respective V_{small}. Hence, all included tetrahedra do contain a gradient that points to V_{next}, so no ambiguity of JHOP paths will result.

4. Identifying critical simplices:

   - A minimum vertex V_{min} is defined when all cofacent tetrahedra have $V_{small} = V_{min}$ (I.e. the reverse gradient vectors of all cofacent tetrahedra point to V_{min}). The reverse gradient path thus ends at this point.
   - For all other cases, a critical face (triangular 2-simplex) is any face that is a member of two reverse gradient pairs. This will return both maxima and 1-saddles.
     - To differentiate between these maxima and 1-saddles, there may be explicit ways to categorize based on the values of surrounding local vertices. However, do note that running *JHOP*$^{+}$will return definite maxima; our main concern here is the identification of 1-saddles.

5. You should now have a set of decreasing gradient paths that partition the set of tetrahedra into classes of common V_{min}, which can hence be considered a min class (ascending manifold).

6.3.3 Comments on JHOP

Simply to reiterate: alternatively to the process described above, JHOP can be largely simplified to the process of first performing regular HOP on the 1-skeleton of the Delaunay tessellation and then assigning each tetrahedron to the max class of its densest vertex (analogously in the *JHOP*$^{-}$manner). Gradients can be constructed after the partitioning of data and will yield the same final result.

The main purpose of JHOP was to identify saddles in an easy and intuitive way: locating the point of gradient "separation" between two maxima or two minima, where a single face is assigned as a gradient pair within two separate max/min classes. However, the "gradient pair method" as described here proved incomprehensive at identifying saddles between all neighboring maxima/minima.

Lastly, JHOP does possess further potential for new information that we chose not to explore. For example, describing the JHOP gradient pairs in terms of vectors could provide additional information with which to post-process the data. Also, there may be other ways to identify critical values in fashions similar to regular HOP (boundary simplices, bond edges, etc.). However, we did not immediately recognize any significant pursuits within this line of thinking.

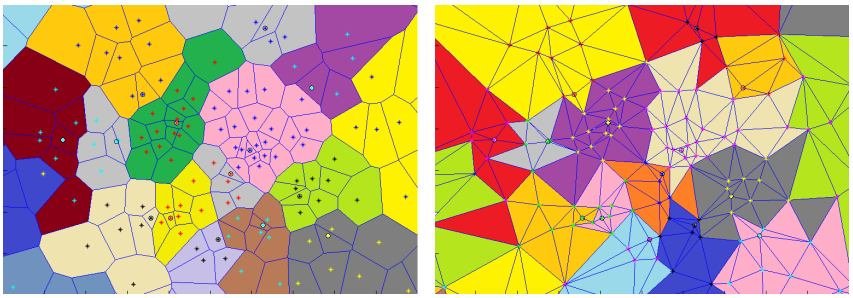


Figure 17: Comparison of HOP and JHOP.

6.4 Dimensional HOP

Although somewhat convoluted, and not by intention, Morse theory, and, by extension, discrete Morse theory, harbors a feature we wish to extract from the finite set of data points—the ability to return structures of all scales with respect to the parameter of topological persistence. While theoretically feasible, the requirement of gargantuan computational memory in the current implementation of discrete Morse theory renders its application to the SDSS data set utterly unattainable; in contrast, HOP is of little computational effort. Thus, we wish to further explore the various applications of HOP in hopes of the admission of a similar, albeit vastly simplified, version of the true result provided by Morse theory employing an analog of topological persistence.

Prior to attempting versions of topological persistence (well outside the scope of this survey), however, recall that the principal ingredients are the critical points of all orders and although we implemented algorithms in which maxima, saddles, and minima are relatively easily defined and computed, the nature of their definitions do not readily lend to well-behaved higher-dimensional generalizations, demonstrating the need for an alternative construction.

With the underlying assumption of the tessellation of the underlying data space by a simplicial complex, we noted that its atoms are among the least complex topological objects, each homeomorphic to B^n, with the behavior that whenever two intersect, the intersection must also be homeomorphic to B^i, for some $0 \leq i \leq n-1$, yielding quite well-behaved results in terms of critical points of various orders that are dependent upon the degree of the simplicial complex on which HOP operates.

6.4.1 Complete Simplicial Complexes

Definition 21. Define $B^n = \{(x_1, \ldots, x^n) \in \mathbb{R}^n : \|(x_1, \ldots, x_n)\| \leq 1\}$,
$B^n_+ = \{(x_1, \ldots, x_n) \in B^n : x_n \geq 0\}$, and $B^n_- = \{(x_1, \ldots, x_n) \in B^n : x_n \leq 0\}$.

Remark 1. Depending on the context, when we say that an $A \subseteq \mathbb{R}^n$ is a B^i, whether the A is explicitly mentioned or implicitly assumed, we mean that A is homeomorphically equivalent to B^i.

Definition 22. Define a relation $N \subseteq \mathcal{P}(\mathbb{R}^n) \times \mathcal{P}(\mathbb{R}^n)$ to be such that $(A, B) \in N$ if and only if

- A and B are both B^i,
- $A \cap B = \partial A \cap \partial B$, and
- $A \cap B$ is an $(i-1)$-dimensional manifold with boundary.

If $(A, B) \in N$, we call A and B *neighbors*. As an abuse of notation, we will define $N(A) = \{B \in \mathcal{P}(\mathbb{R}^n) : (A, B) \in N \wedge B \text{ is } B^i\}$.

Definition 23. Let $0 \leq i \leq n$. Let $S \subseteq \mathcal{P}(\mathbb{R}^n)$ be such that for all $t \in S$, t is B^i. The *restriction* of N to S, $N \restriction S$, is a subset of N such that $(A, B) \in N \restriction S$ if and only if $A, B \in S$ and $(A, B) \in N$. Similarly, $N \restriction S(A) = \{B \in S : (A, B) \in N \restriction S\}$.

Definition 24. An *N-path* $P = \{B_2^i, \ldots, B_{m-1}^i\}$ from B_1^i to B_m^i is such that for all $1 \leq j \leq m-1$, $(B_j, B_{j+1}) \in N$. Similarly, an *$(N \upharpoonright S)$-path* is such that for all $1 \leq j \leq m$, $B_j^i \in S$.

Definition 25. Let $P \subseteq \mathbb{R}^n$. Its *convex hull* is the intersection of all $t \subseteq \mathbb{R}^n$ such that t is connected and convex, and $P \subseteq t$. The *convex hull function* $CH : \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^n)$ is such that for each $P \subseteq \mathbb{R}^n$, $CH(P)$ is the convex hull of P.

Definition 26. Given $P = \{p_1, \ldots, p_i\} \subseteq \mathbb{R}^n$, its i-dimensional *simplex* is $CH(\{p_1, \ldots, p_i\})$.

Proposition 27. *Let $CH(P)$ be a simplex. Then for all $Q \subseteq \mathcal{P}(P)$, $CH(Q)$ is a simplex.*

Proof. Follows directly from definition. □

Definition 28. A *simplicial complex* is a collection of simplices $\{CH(P_1), \ldots, CH(P_m)\}$ such that for all $1 \leq i, j \leq m$, $i \neq j$, if $P_i = \{p_{i_1}, \ldots, p_{i_k}\}$ and $P_j = \{p_{j_1}, \ldots, p_{j_l}\}$ are such that $CH(P_i) \cap CH(P_j) \neq \emptyset$, then for some $1 \leq o_1, \ldots, o_q \leq k$ and $1 \leq p_1, \ldots, p_r \leq l$, $\{p_{i_{o_1}}, \ldots, p_{i_{o_q}}\} = \{p_{j_{p_1}}, \ldots, p_{j_{p_r}}\}$, and $CH(P_i) \cap CH(P_j) = CH(\{p_{i_{o_1}}, \ldots, p_{i_{o_q}}\})$.

Definition 29. An n-dimensional *complete simplicial complex* is a simplicial complex such that if, for some $0 \leq i \leq n$, $CH(\{p_1, \ldots, p_i\})$ is a simplex in it, then there exists a simplex $CH(\{q_1, \ldots, q_n\})$ in it such that $\{p_1, \ldots, p_i\} \subseteq \{q_1, \ldots, q_n\}$.

Remark 2. For complete simplicial complexes, since lower-dimensional simplices are already accounted for in the highest-dimensional simplices, we omit writing them when listing the individual simplices.

Definition 30. The boundary, ∂S, of an n-dimensional complete simplicial complex S, is the set of $(n-1)$-dimensional simplices such that $CH(\{p_1, \ldots, p_{n-1}\}) \in \partial S$ if and only if

- There exists a $CH(\{q_1, \ldots, q_n\}) \in S$ such that $\{p_1, \ldots, p_{n-1}\} \subseteq \{q_1, \ldots, q_n\}$, and
- For all $CH(\{q_1, \ldots, q_n\}), CH(\{r_1, \ldots, r_n\}) \in S$ such that $(CH(\{q_1, \ldots, q_n\}), CH(\{r_1, \ldots, r_n\})) \in N \upharpoonright S$, $\{q_1, \ldots, q_n\} \cap \{r_1, \ldots, r_n\} \neq \{p_1, \ldots, p_{n-1}\}$.

Lemma 31. *∂S is an $(n-1)$-dimensional complete simplicial complex.*

Definition 32. An n-dimensional complete simplicial complex S is *connected* if for all $s_1, s_2 \in S$, there exists an $(N \upharpoonright S)$-path from s_1 to s_2.

Remark 3. A path-connected (i.e., connected in the traditional sense, since simplicial complexes are compact) complete simplicial complex is not necessarily connected, e.g., take two tetrahedra such that they intersect at one (entire) edge only.

Remark 4. $(N \upharpoonright S)$-paths do not make sense when S is not a complete simplicial complex, e.g. a triangle joined to an edge at a vertex.

Remark 5. The boundary of a connected complete simplicial complex is not necessarily connected or path-connected (i.e., connected in the traditional sense).
E.g., take the unit square, divide into nine smaller squares in the traditional way. Remove the center square, then draw one diagonal line across each of the smaller squares.

Definition 33. Let S be an n-dimensional complete simplicial complex. A *partition* is a $\Pi \subseteq \mathcal{P}(S)$ such that

- For all $\pi \in \Pi$, for all $s_1, s_2 \in \pi$, there exists an $(N \upharpoonright \pi)$-path from s_1 to s_2,
- For all $\pi_1, \pi_2 \in \Pi$, $\pi_1 \cap \pi_2 \neq \emptyset$, and
- $\bigcup \Pi = S$.

Proposition 34. *π is a connected complete simplicial complex.*

Proof. Follows directly from definition. □

Definition 35. The boundary of a partition, $\partial\Pi$, is $\bigcup_{\pi\in\Pi}\partial\pi$.

Remark 6. The definition of the boundary of a partition is a little arbitrary. Another equivalent definition is to take $\partial S \cup \bigcup_{\pi_1,\pi_2\in\Pi}\bigcup_{s_1\in\pi_1,s_2\in\pi_2} s_1\cap s_2$.
E.g., take a triangle T, pick a $p\in\text{int}(T)$, then draw straight lines to each of the vertices to form three triangles T_1,T_2,T_3. Form T_4 by picking a q noncoplanar to T, then drawing straight lines to p and one vertex of T. $S=\{T_1,T_2,T_3,T_4\}$ is then a complete simplicial complex. Partition it with $\pi_1=\{T_1,T_2,T_3\}$ and $\pi_2=\{T_4\}$.
Note that we are essentially implicitly assuming that $\pi_1\cap\pi_2\subseteq\partial\pi_1$ and $\pi_1\cap\pi_2\subseteq\partial\pi_2$, i.e. in the example, we imply that $p\in\partial\pi_1$ when $p\in\text{int}(\pi_1)$, considering that π_1 is a complete simplicial complex itself and is homeomorphic to B^2, and we arrive at points that are seemingly simultaneously on the interior and the boundary.
In the stated definition, the boundaries of two elements in a partition are assumed to be of no relation to each other (although we still consider both to be part of the larger structure), thus these ambiguities are sidestepped.

Lemma 36. $\bigcup\partial\Pi = \bigcup\partial S \cup \bigcup_{\pi_1,\pi_2\in\Pi}\bigcup_{s_1\in\pi_1,s_2\in\pi_2} s_1\cap s_2$.

Proof. $\partial S = \bigcup_{\pi\in\Pi}\bigcup_{s\in\pi,\forall t\in S\setminus\{s\}((s,t)\notin N)}\subseteq\partial\Pi$ Let $s\in\partial S$. Then by definition of ∂S and Π, $s\in\partial\pi$ for some $\pi\in\Pi$, and thus $s\in\partial\Pi$.
Let $\pi_1,\pi_2\in\Pi$, $s_1\in\pi_1$, and $s_2\in\pi_2$ be such that $s_1\cap s_2\neq\emptyset$. Clearly, if $(s_1,s_2)\notin N\restriction S$, then either $s_1\cap s_2\in\bigcup\partial S$ and there exists an $s_3\in\partial S$ such that $s_1\cap s_2\subseteq s_3$, or there exists a $\pi_3,\pi_4\in\Pi$ and an $s_3\in\pi_3$ and $s_4\in\pi_4$ such that $s_1\cap s_2\subseteq s_3\cap s_4$ and $(s_3,s_4)\in N$ (could be that one of them is actually either s_1 or s_2).
The former is already shown, and in the latter, by definition $s_3\cap s_4\in\partial\pi_3$. Thus $\bigcup\partial S\cup\bigcup_{\pi_1,\pi_2\in\Pi}\bigcup_{s_1\in\pi_1,s_2\in\pi_2} s_1\cap s_2\subseteq\bigcup\partial\Pi$.
Let $\pi_1\in\Pi$, and let $CH(\{p_1,\ldots,p_{n-1}\})\in\partial\pi_1$. Then for some $CH(\{q_1,\ldots,q_n\})\in\pi_1$, $\{p_1,\ldots,p_{n-1}\}\subseteq\{q_1,\ldots,q_n\}$.
If there exists an $CH(\{r_1,\ldots,r_n\})\in S\setminus\pi$ such that $\{p_1,\ldots,p_{n-1}\}\subseteq\{r_1,\ldots,r_n\}$, then for some $\pi_2\in\Pi$, $CH(\{r_1,\ldots,r_n\})\in\pi_2$, and $\{p_1,\ldots,p_{n-1}\}=\{q_1,\ldots,q_n\}\cap\{r_1,\ldots,r_n\}$.
Otherwise, clearly, $CH(\{p_1,\ldots,p_{n-1}\})\in\partial S$.
Thus $\bigcup\partial\Pi\subseteq\bigcup\partial S\cup\bigcup_{\pi_1,\pi_2\in\Pi}\bigcup_{s_1\in\pi_1,s_2\in\pi_2} s_1\cap s_2$. □

Theorem 37. *$\partial\Pi$ is an $(n-1)$-dimensional complete simplicial complex.*

Proof. Follows directly from the equivalent definition in theorem 36. □

Remark 7. $\partial\Pi$ is not necessarily connected.

Theorem 38. *For all $0\leq i\leq n$, $\partial^i\Pi$ is an i-dimensional complete simplicial complex, where $\partial^0\Pi=\Pi$, and $\partial^i\Pi$ is $\partial\cdots\partial\Pi$ i times.*

Proof. Follows by repeated application of theorem 37. □

Remark 8. Theorem 37 can be easily extended to tessellation of compact polygonal or polyhedral (2- or 3-dimensional) manifolds via polygons or polyhedra, as well as higher-dimensional tessellations by polytopes, by first creating a simplicial decomposition of each of the shapes with the requirement that the union of the complete simplicial complexes of each of the shapes still forms a complete simplicial complex, then partitioning such that the simplices are grouped such that the simplices from one shape belong to one and only one partition, e.g. the set of compact Voronoi cells given a set of points and its corresponding Voronoi tessellation.

Remark 9. The requirement of a simplicial complex can be further extended to particular tessellations by B^n, by first decomposing each B^n into smaller B^n such that, as a whole, the intersection between any two of the (smaller) B^n is B^i, for some $0\leq i\leq n-1$ (along with the rest of the requirements of a simplicial complex translated over).
Essentially, this follows through because there exists a homeomorphism between the tessellation of smaller balls and a complete simplicial complex.

Theorem 39. *HOP returns a partition of S (see proposition 12.*

Remark 10. Note that when applying theorem 38 and theorem 39 recursively, we need not apply HOP to the 0-dimensional complete simplicial complex, as each of its elements is disconnected.

Remark 11. Together, theorem 38 and theorem 39 return critical simplices of all dimensions from n to 0, which we liken to the order of the critical simplices described in discrete Morse theory.

Remark 12. Occasionally, certain elements in a partition may not return useful information. E.g., let S be a tessellation of S^2, and let Π be a partition of a 2-dimensional complete simplicial complex such that $S \in \Pi$. If there exists a HOP function that returns this, then on S, although the maximum exists, there is no boundary. In other words, we need to either augment the definition of the boundary of a partition to incorporate these cases, or treat it separately and consider it as the equivalent of a manifold without boundary (with the rest equivalent to manifolds with boundary).
An extension of this problem is when S contains extraneous triangles, i.e. "fins" protruding from S^2.

6.4.2 Delaunay Tessellation

A Delaunay tessellation D is a complete simplicial complex, thus by theorem 38 and theorem 39, we may apply HOP to D, then recursively apply HOP to the results.

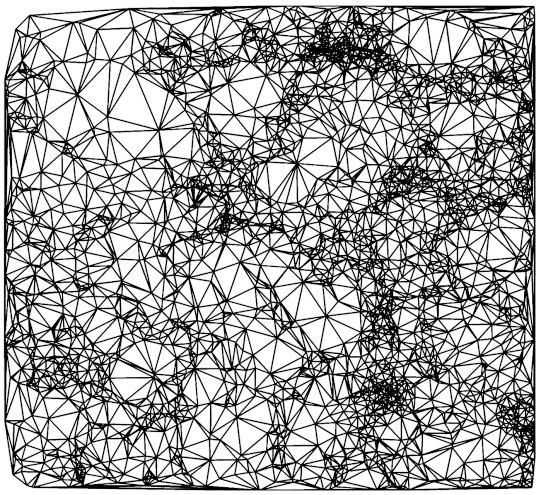


Figure 18: The 2-dimensional slice of the SDSS data, after Delaunay tessellation. Each triangle is outlined by the black lines.

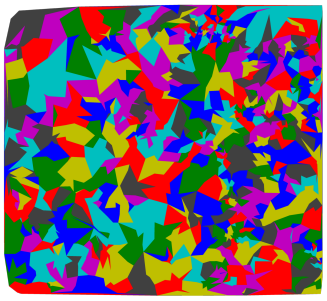


Figure 19: After performing HOP on the triangles, each element in the resulting partition is colored differently.

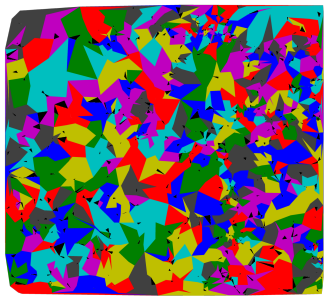


Figure 20: The same as figure 19, with the maximum of each of the classes colored in black.

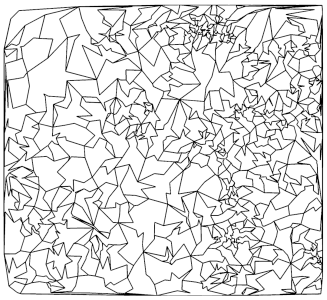


Figure 21: Each of the straight line segments here represents a 1-simplex in the boundary of the 2-dimensional partition.

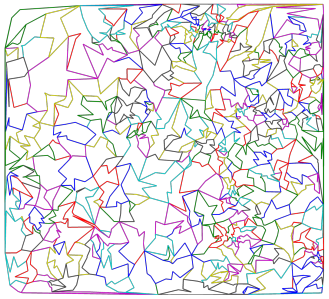


Figure 22: Performing HOP on the 1-dimensional simplicial complex returns a partition, each element in it colored differently.

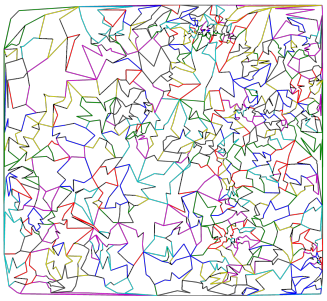


Figure 23: The maximum of each of the classes is colored in black.

Figure 24: The boundary of the partition is then a 0-dimensional simplicial complex.

Before applying this to the slice of the SDSS data in figure 18, note that HOP functions are initially defined on the vertices of the Delaunay tessellation only. Thus we first assign values to each of the triangles and edges linearly, to be able to perform HOP on the 2- and 1-dimensional complete simplicial complexes.

For the triangles, we obtain a partition formed of 2-dimensional complete simplicial complexes, pictured in figure 19, with the corresponding maxima in figure 20.

Note that although each of the neighboring classes should be colored differently,

occasionally they are colored the same.

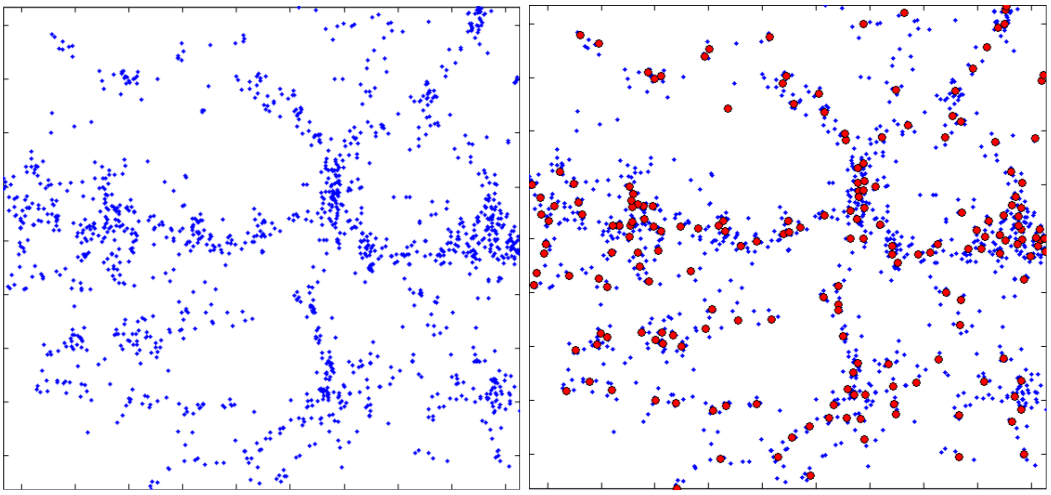
We then obtain a 1-dimensional complete simplicial complex as the boundary of the partition in figure 21. Performing HOP on it returns a partition, shown in figure 22, with the corresponding maxima colored in black in figure 23.

Finally, its boundary is a 0-dimensional complete simplicial complex, shown in figure 24.

7 HOP Geodesics and Alpha Complexes

7.1 HOP geodesics

The HOP algorithm provides an efficient way to locate local maximum density points, and it also determines the set of points which terminate at each maximum. This provides us with a meaningful partition of the data into max-classes. Furthermore, each max-class is represented by a single unique maximum. These max classes are analogous to partitioning the data space into descending manifolds in smooth Morse theory. But in order to analyze larger scale structures, we need a method of grouping these max classes in a way that is geometrically meaningful.



(a) A thin slice of the SDSS data flattened to two dimensions.

(b) All of the HOP maxima for this set of points are shown in red.

Figure 25

The motivation for this HOP geodesic idea came from observing the results from the HOP algorithm. In figure 25(a) we have a thin slice of the raw SDSS data which has been projected down to two dimensions. We ran HOP on this data set and all of the resulting HOP maxima are shown in figure 25(b). Each red point is a HOP maximum. One advantage to using the HOP algorithm on SDSS data is that the max-classes tend to be fairly small. So in the high density cluster regions many maxima are bunched close together. In the void regions very few (if any) maxima can be seen, which is not surprising. But what is interesting is that the maxima appear to trace out the significant filamentary structures. Our method described in this section attempts to connect neighboring maxima via paths in such a way that we can chain the maxima together along the filaments. These paths are built from vertices and edges in the Delaunay triangulation of the data (see figure 26) and we use the HOP algorithm to help generate the paths. Also, we assign a length to each path which takes into account the density of the vertices in the path. The overall goal is to trace out the "spine" of the data efficiently.

Our method for doing this was to consider each HOP maximum as the representative for its max class and connect neighboring maxima via paths that function like geodesics. (Recall that two maxima are "neighboring maxima" if their respective max classes have a bond edge in common.) We define a certain pseudo-metric on the set of HOP maximum

points to determine the distance between neighboring maxima and also define the geodesic path. We will of course define this more precisely below, but our idea was motivated by the following observation. For our purposes we want to group together HOP maxima if they are members of the same over-dense structure in the data set. If two max-classes are a part of the same cluster or filament, then we want a method for joining them together. So we consider two neighboring maxima to be "close" if there is dense chain of data points between them. The preferred path (the geodesic) will be the one that follows along the densest structure the best. Our hope was that this process would connect neighboring maxima by paths that follow along one dimensional paths of highest local density. That is, these paths would not be paths of shortest euclidean length, instead they would follow along the densest path (in a certain sense we define shortly). This, we hoped would connect neighboring maxima along one dimensional filamentary structures that tie them together.

For an example see figure 27(a) where we have drawn two neighboring max-classes: one above in blue and one below in red. The representative maxima for each max-class is enlarged and outlined in black. So all of the blue points HOP to the blue max and all of the red points HOP to the red max. Note that the two max classes appear connected via a filamentary chain of points on the right hand side of the picture. But compared to the point density of this filament, the region directly between the maxima has a larger gap or void region. We wish to find a path which connects the two maxima but runs through the densest structure that binds their max-classes together. In figure 27(b) you see two different paths connecting the two maxima. The solid green path is the shortest Euclidean length path connecting them. Note that this green path cuts right through the relatively void region. The dashed yellow and black path travels through the filamentary structure to connect the maxima. Note that the longest edge of the solid path is much longer than the longest edge of the dashed path. Any path which traverses a relatively void region must have longer edges to stretch across the void. Paths which move through denser regions have relatively shorter jumps from point to point.

This observation suggests the following general observation: among all the paths which connect the two neighboring maxima, we want to find the path which minimizes the longest edge in the path. There are practical problems with implementing this however. We cannot literally search *all* possible paths connecting two neighbor maxima, so we limit our search to the paths which go through the bond edges that tie the two neighbor max-classes together. (In figure 27(b), the two edges that have different color endpoints are bond edges, so these two paths would be considered in our search.) This means that we do not consider paths that travel through a third max class; the vertices in the path must be contained in the union of the two neighboring max-classes. The bond edge in the path is generally somewhere in the middle of the path and occurs at a pair of points on the boundary of the two max-classes, hence they are lower density points (relative to the maxima). We further simplify our

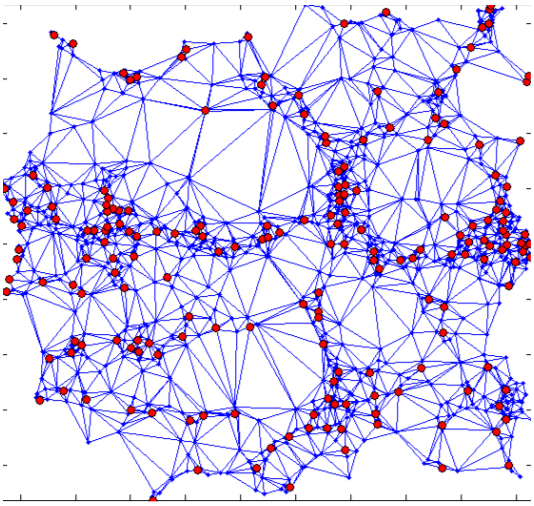
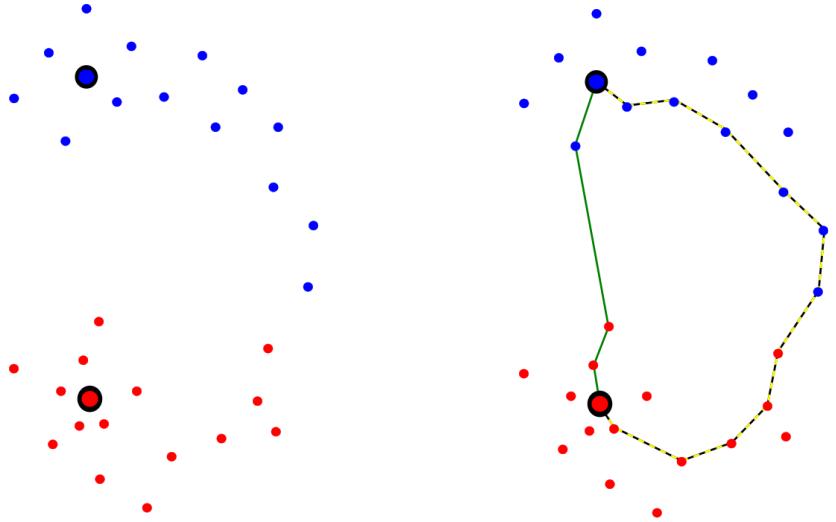


Figure 26: The raw data and the HOP maxima are shown with the Delaunay triangulation. We use paths in the Delaunay graph to connect neighboring maxima.



(a) Two neighboring max-classes are shown. The representative maxima are the larger points outlined in black.

(b) The shortest *Euclidean* length path connecting neighbor maxima is shown in green. The black and yellow dashed path (the HOP geodesic) connects the maxima through a filamentary structure.

Figure 27: Note that the longest edge in the green path is much longer than the longest edge in the black and yellow path. Our algorithm finds the path that minimizes the longest edge in the path and hence favors the dashed path. In our metric the dashed path is shorter than the green path.

search with the following observation: the two endpoints of a bond edge are, by definition, in different neighboring max-classes. So since we want the path connecting the two maxima to go through dense structures, the most direct route from each endpoint of this bond edge to the representative max is to use the paths determined by HOP from these endpoints to the maxima. The HOP paths from the vertices of the bond edge will HOP along points of increasing density to the maxima. So if we think of point density as altitude of a mountain peak, the path that we desire which connects the two neighboring maxima will be like a mountain trail connecting two neighboring mountain peaks which travels through the tallest ridge line between the peaks.

Now we more precisely define the HOP geodesic and the algorithm to find it. We begin by defining the paths connecting neighboring maxima and then define a pseudo-metric on these paths. Let $\widetilde{M}_a$ and $\widetilde{M}_b$ be two neighboring max-classes. So M_a and M_b are the representative maximum points for the two max-classes (see figure 28). Since, $\widetilde{M}_a$ and $\widetilde{M}_b$ are neighboring max-classes, then there is a bond edge, $e = (v_a, v_b)$, which connects the two neighboring max classes (see figure 29(a)). The endpoints of e, namely v_a and v_b, are members of $\widetilde{M}_a$ and $\widetilde{M}_b$ respectively. This means that there is a unique ascending HOP path (via the HOP$^+$ algorithm) from v_a to M_a, similarly from v_b to M_b. We denote these HOP paths $v_a \nearrow M_a$ and $v_b \nearrow M_b$. HOP paths are, of course, directed paths, so by removing the direction of the arrows of these two HOP paths, we obtain two *undirected* paths: $v_a \leftrightarrow M_a$ and $v_b \leftrightarrow M_b$. Thus we can connect these two paths by the edge $e = (v_a, v_b)$ to create a single, undirected path connecting M_a and M_b (see figure 29(b)).

$$P(e, M_a, M_b) = M_a \leftrightarrow v_a \overset{e}{-} v_b \leftrightarrow M_b \tag{9}$$

We call the path in equation (9) a ***HOP neighbor-max path***. The notation $P(e, M_a, M_b)$ emphasizes the fact that the path connects M_a to M_b via their common bond e, but this notation can be simplified. Note that the edge $e = (v_a, v_b)$ *uniquely* determines the path $P(e, M_a, M_b)$ since the HOP paths from the endpoints of e are unique. (See the section on HOP for a proof of the uniqueness of HOP paths.) Thus the notation on the left hand side of (9) can be simplified to $P(e)$ as long as it is clear what max-classes the endpoints of e belong to. Furthermore, it should be noted that *every* bond edge determines a unique

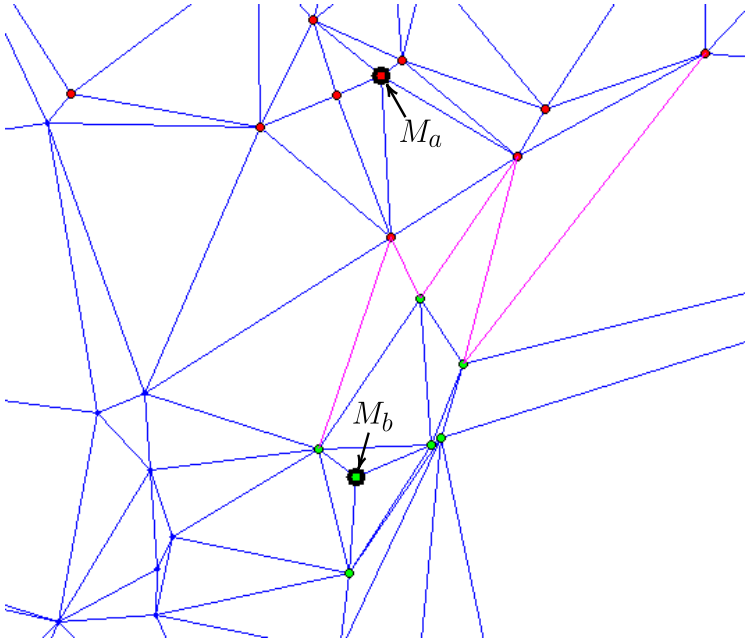
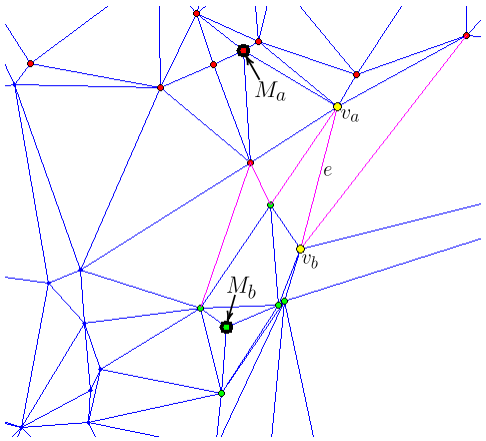


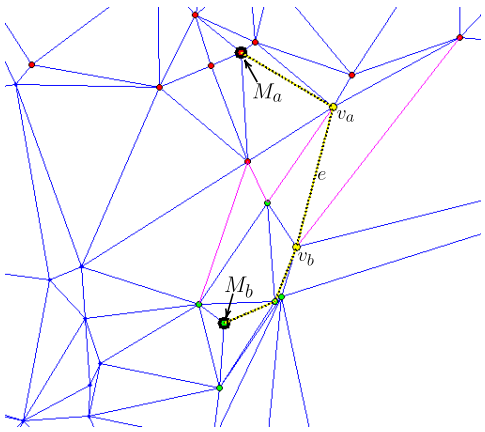
Figure 28: Two neighboring max-classes are shown. $\widetilde{M_a}$ consists of the red points. $\widetilde{M_b}$ consists of the green points. The representative maxima, M_a and M_b, are outlined in black. The five bond edges connecting these two max-classes are shown in magenta.

HOP neighbor-max path. Also note that although each neighbor-max path is unique, two different HOP neighbor-max paths may have some edges in common (see figures 30 and 31).

Now that we have defined the paths that connect neighboring maxima, we can define a notion of length of these paths. Consider two neighboring maxima M_a and M_b and let $P(e)$ be a neighbor-max path connecting them. The key to our approach is the following: we define the length of $P(e)$ to be the Euclidean length of the longest edge in $P(e)$. More formally, let f represent an arbitrary edge in the Delaunay graph and denote the Euclidean length of f by $|f|$. Since $P(e)$ is a path in the Delaunay graph, it has an edge set $\{f_1,\ f_2 \cdots,\ f_k\}$.



(a) The bond edge e has endpoints v_a and v_b (shown in yellow). Each bond edge determines a unique path from M_a to M_b.



(b) The dashed yellow path is the HOP neighbor-max path determined by e. Each bond edge determines a unique path.

Figure 29: The HOP algorithm determines a HOP path from v_a to M_a with one edge. The HOP path from v_b to M_b has two edges. Concatenating these two HOP paths with e in the middle (and removing the edge directions), produces the *HOP neighbor-max path* with four edges (the yellow dashed path). We define the length of this yellow path to be the Euclidean length of the longest of the four edges in the path.

We define the length of $P(e)$ to be

$$|P(e)| = \max\{|f_1|,\ |f_2|, \cdots,\ |f_k|\}. \tag{10}$$

(This definition of length is technically not a metric. This is because the definition of the length of a path is not defined for any path between any arbitrary pair of maxima; it is only defined for a pair of *neighboring maxima*. Because of this, the triangle inequality does not hold in general. We could define the length between non-neighboring maxima to be infinity, and work to make this a legitimate metric on the set of HOP maxima, but we don't require this for our purposes.)

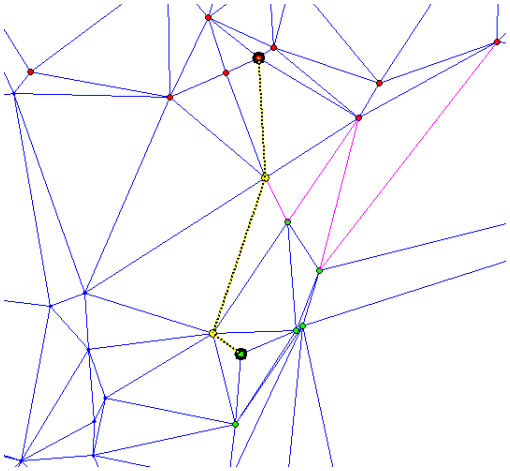


Figure 30

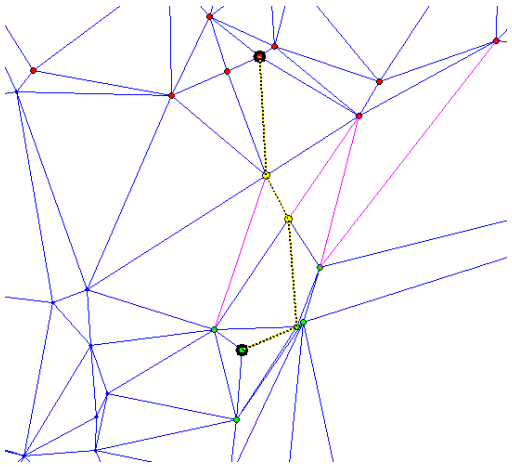


Figure 31

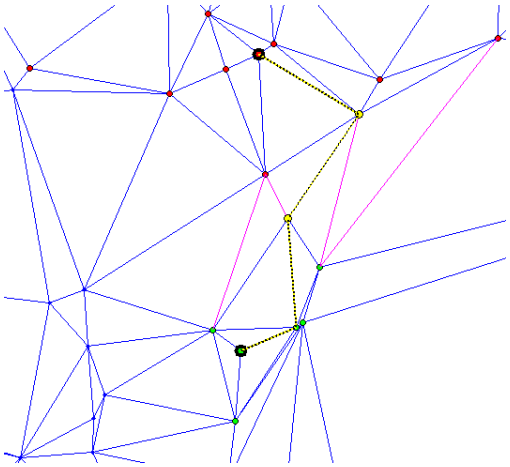


Figure 32

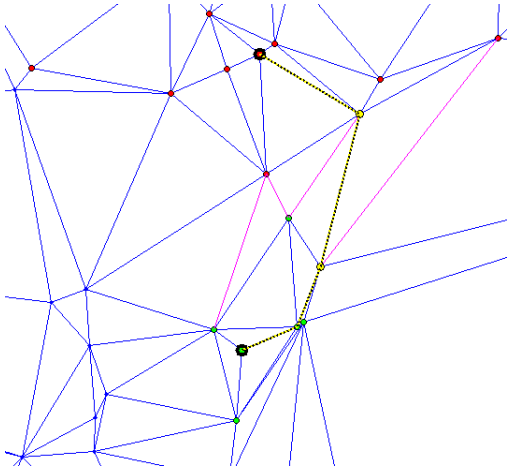


Figure 33

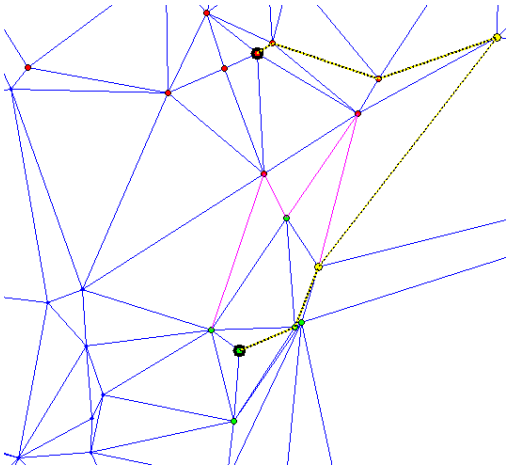


Figure 34

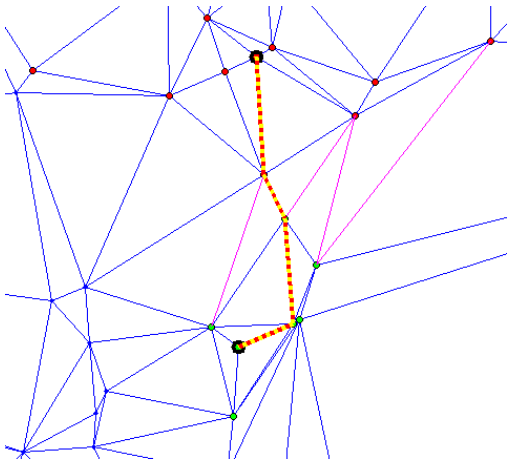


Figure 35: The HOP geodesic which connects these two neighboring maxima.

Now that we have defined the length of a neighbor-max path, we can easily define the distance between M_a and M_b. Because they are neighboring maxima, they share at least one bond edge and the set of all bond edges they share is finite: $\{e_1, e_2, \cdots, e_n\}$. These n bond edges determine n unique HOP neighbor-max paths connecting M_a to M_b:

$$\big\{P(e_1),\ P(e_2), \cdots,\ P(e_n)\big\}. \tag{11}$$

We define the distance between M_a and M_b to be the length of the shortest of all the

neighbor-max paths connecting them (where "length" here is in the sense of equation (10) above). So the distance is

$$d(M_a, M_b) = \min\Big(\big|P(e_1)\big|,\ \big|P(e_2)\big|,\ \cdots, \big|P(e_k)\big|\Big). \tag{12}$$

Now, it would seem natural to define the resulting geodesic to be the path which attains the minimum length in (12), but although the minimum length is well defined, it is certainly possible that more than one path attains this minimum length since two paths can share their minimum length edge. (Remember that two different max-paths can share some edges.) In order to choose the geodesic in this case, we compare the second longest edge among the minimum length paths, then the third longest edge, etc. In practice this process will terminate at a single well-defined geodesic since the paths which attain the minimum length in (12) cannot share every edge, and double precision floating point calculations ensure that unique edges almost surely have different length. This process selects which of the $P(e_j)$ will be the ***HOP geodesic*** connecting M_a to M_b.

On page 31 you can see the process of finding the HOP geodesic. Figures 30 through 34 show the five unique paths determined by the five bond edges connecting the two max-classes. Note that figure 30 and 31 are different paths which share one edge. Among the five paths, the one in figure 31 has the shortest maximum edge length. It is the HOP geodesic shown in figure 35.

With these definitions in hand, every neighboring pair of maxima has a unique geodesic path connecting them. This also allows us to measure the distance between two maxima. Two neighboring maxima are "close" if the geodesic connecting them consists of relatively short edges. And neighboring maxima are "far apart" if the geodesic connecting them has at least one long edge (so it has to traverse a larger void region). Maxima that are in dense cluster regions will have very short geodesics because the points are denser there.

We can use these geodesics to connect the maxima together that we saw in figure 25(b) above. Below in figures 36 through 45, we begin by plotting the shortest geodesics and plot progressively longer ones. We simply increase a length threshold and plot any geodesic that is shorter than the threshold. Each figure shows a pair of pictures that go together. On the left you see the raw data points along with the geodesics at that threshold level, and on the right you see the same geodesics but with only the HOP maxima. The maxima are slightly larger black dots than the raw data on the left. The threshold length is printed above the picture on the right in each figure.

Note that in figure 36 and 37, even though we are only plotting the shortest geodesics, the dense cluster regions are already very clear. But the most interesting part is how the filamentary structures appear to be traced out by the geodesics as we increase the threshold. See for example figure 39 where the long horizontal filament in the middle of the data has been traversed by a chain of geodesics. As the threshold increases larger distances are

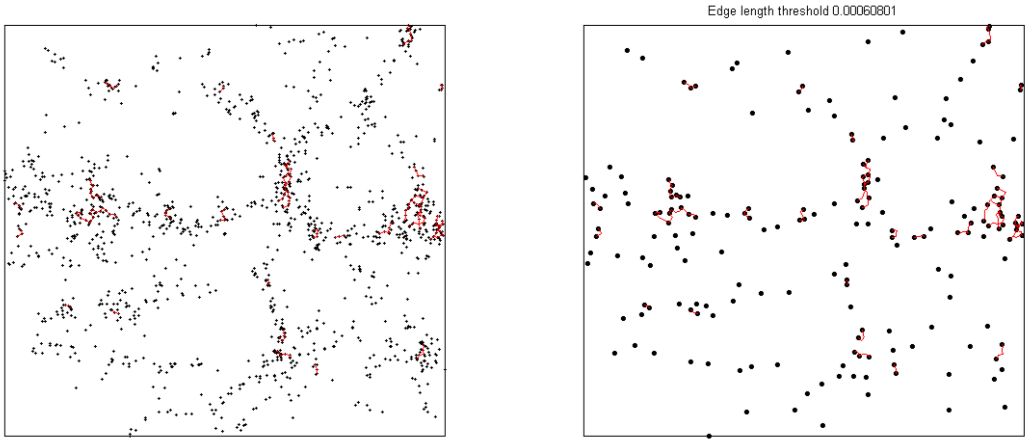


Figure 36: On the left we see the complete raw data along with the geodesics that are shorter than the given threshold. On the right we see the same geodesics but only the HOP maxima are shown.

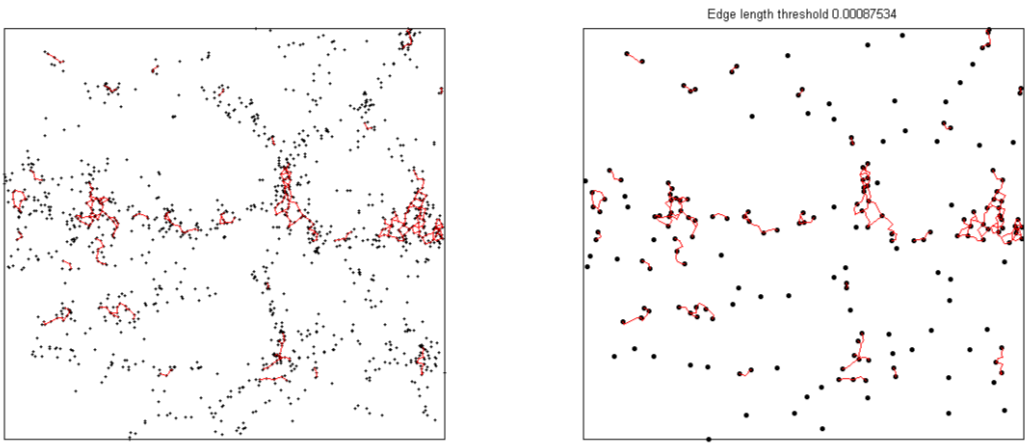


Figure 37

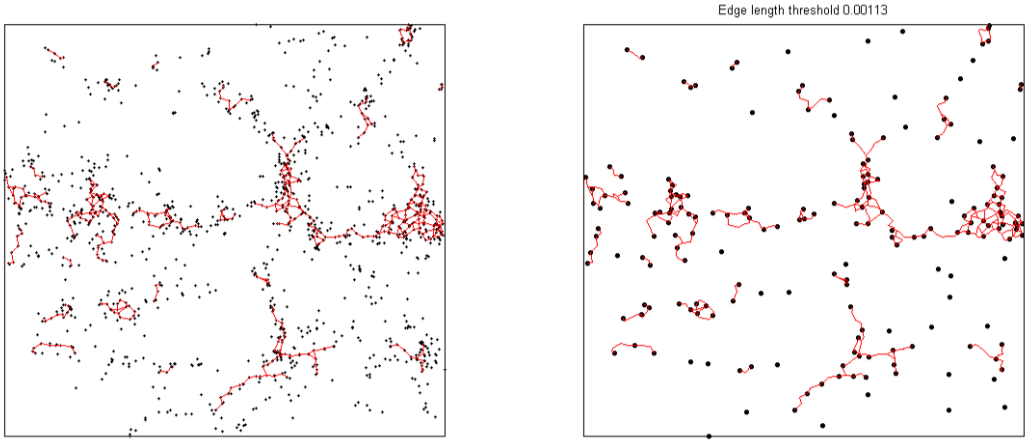


Figure 38

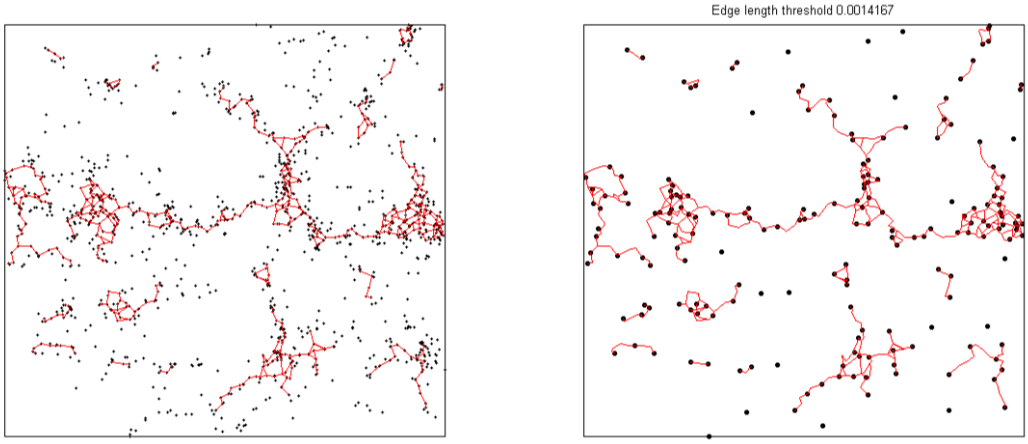


Figure 39

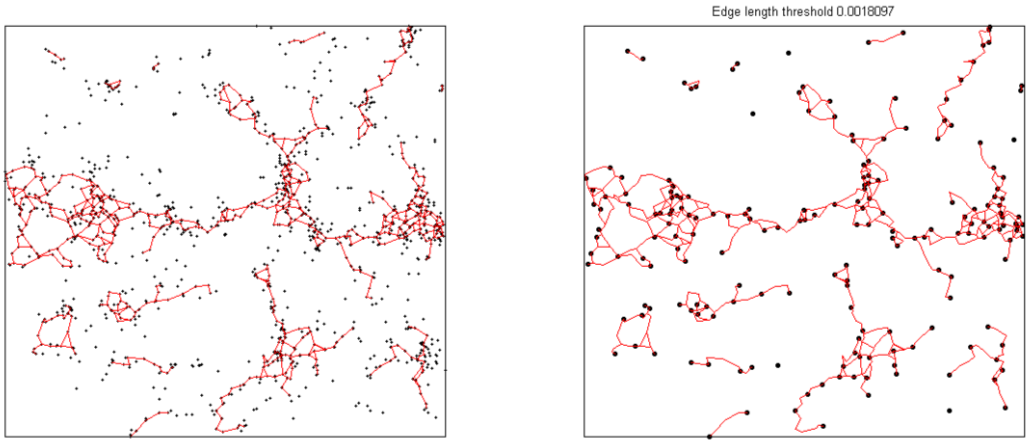


Figure 40

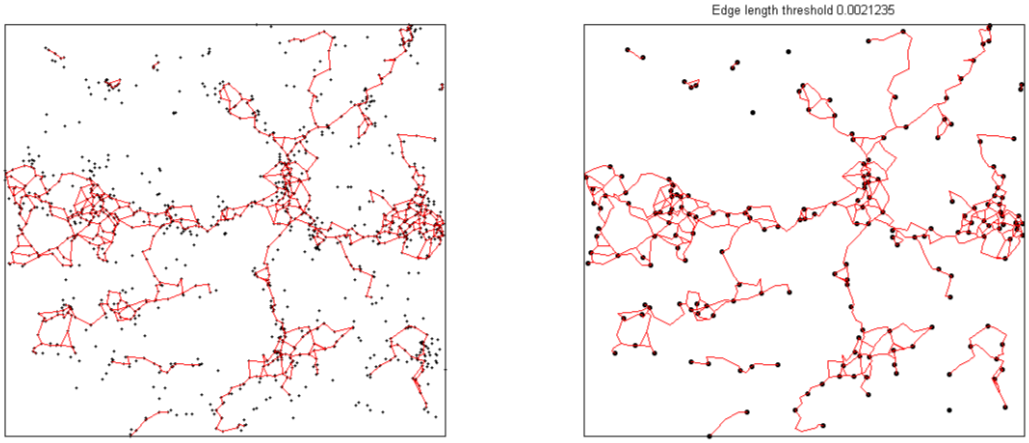


Figure 41

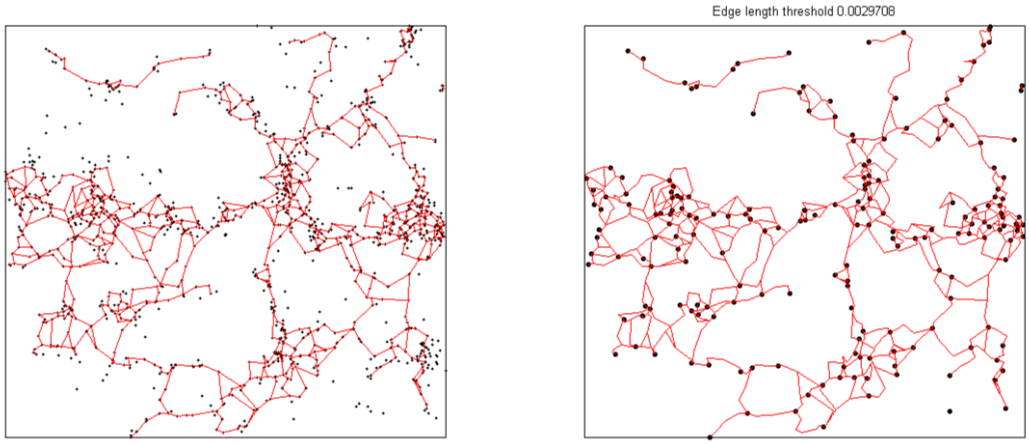


Figure 42

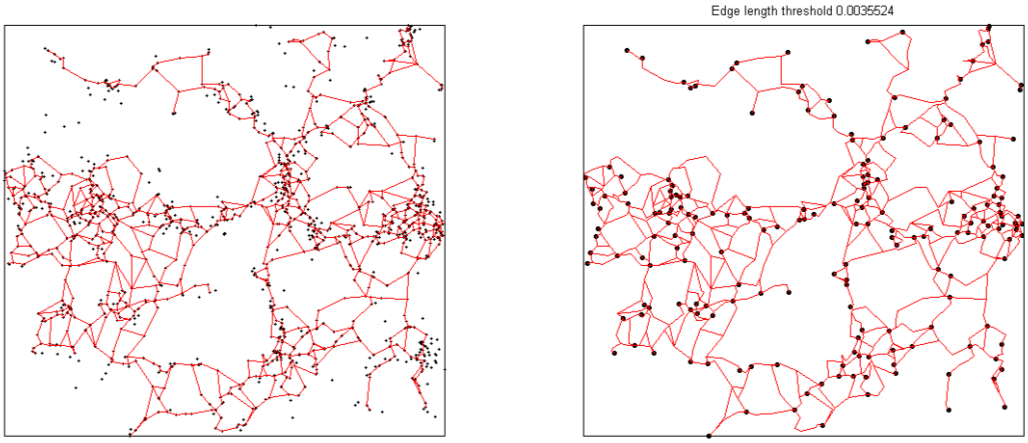


Figure 43

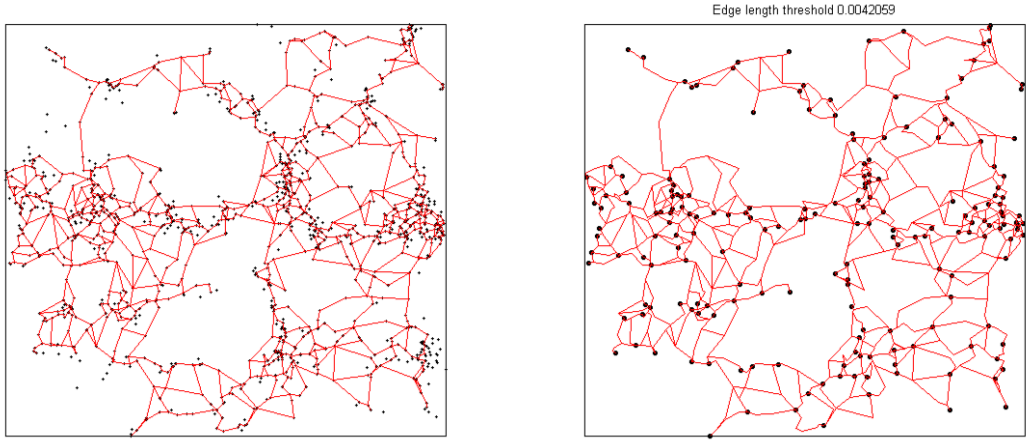


Figure 44

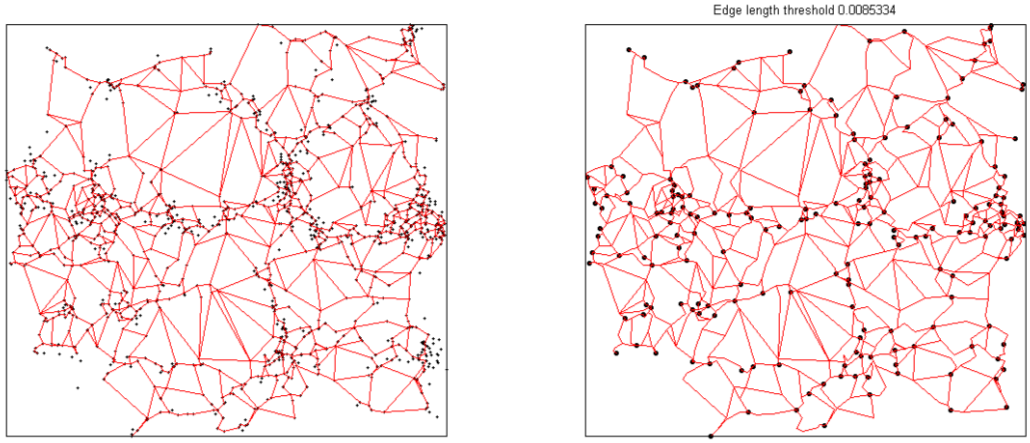


Figure 45

spanned, but even in figure 44 where almost all of the geodesics are plotted, the larger void areas are still clear. Figure 45 shows all geodesics for every pair of neighbors.

In figure 46 we show a close-up at a threshold level very close to that of figure 41. The region shown in figure 46 is in the upper part of figure 41 (slightly to the right of center). This close-up shows two filamentary structures: the denser, thicker one starts at the bottom and goes toward points a and b in the upper left hand corner, and the other goes through d toward c. The green circles in this picture are HOP maxima. At this threshold level, we see a geodesic connecting d to c. Although a and b are neighbor maxima, the geodesic is longer than the threshold so it is not plotted. Note that the Euclidean distance from a to b is clearly shorter than the Euclidean distance from c to d, but the geodesic from a to b is longer because of the larger gap in the data points connecting a to b. This is precisely the effect we intended with our definition of length of a neighbor-max path. This effect is precisely why, as we increase the geodesic length threshold, you see the filamentary structures detected quickly and the larger void regions persist until the threshold is very large.

Although we did not develop this geodesic scheme any further, we feel this could potentially be used as a tool to efficiently detect filamentary structures and also detect voids and void boundaries. For example, neighbor maxima which have a relatively long geodesic connecting them must have a large gap in the data between them, thus these two maxima are likely on opposite boundaries of a void region. A maximum is likely to be in the interior of a cluster region if all of the geodesics connected to it are relatively short, *i.e.* it is close to all of its neighbors. Boundaries of cluster regions as well as filaments could be found by noting that *some* of their attached geodesics are short and others are long. Hence the maximum is on the boundary between a dense region and a relatively void region.

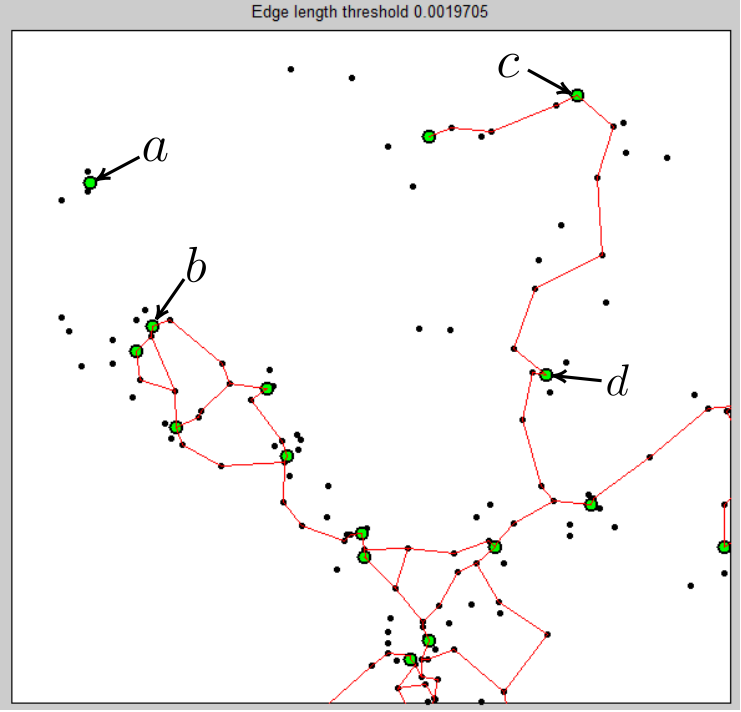


Figure 46: A close-up of some geodesics at a threshold level similar to figure 41. The geodesics under this threshold connect maxima through the dense structures but cannot traverse the void regions.

Another possible use of these geodesics is to merge neighboring max-classes based on the distance between the representative maxima. If the geodesic connecting two maxima is relatively short, then that means there is a dense chain of data points connecting them. Therefore these two max-classes could be merged together into one larger structure. This use of the geodesics would allow one to study the data at various scale levels. If the threshold for merging is low, then the data is partitioned more finely and smaller scale structures could be studied. Increasing the threshold causes more max-classes to be merged into larger scale

structures.

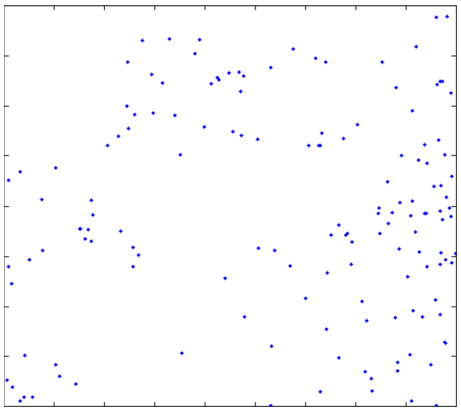
Because the algorithm to find these geodesics relies on the HOP algorithm, it is very simple and runs very quickly on large data sets. We tested this in three dimensions on roughly 135,000 galaxies from the SDSS data and the program finished in less than 30 minutes (and this process was programmed in MATLAB, hence speed could be greatly improved). Plots of geodesics in three dimensions appear similar to the two dimensional examples above.

7.2 A problem with HOP

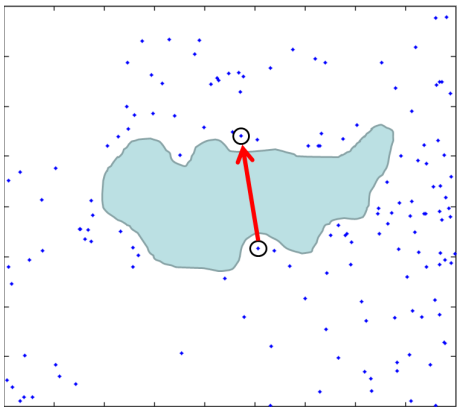
In testing we found that HOP has one potentially significant problem. To explain this problem we show an example. In figure 47(a) we have an example set of raw two dimensional data points. Note that the data appears to have a void region in the middle. In figure 47(b) we shaded the void region blue and we circled two neighboring data points (meaning they share a Voronoi cell wall). It turns out that the HOP algorithm "HOPs" from the lower circled point to the upper circled point because it is the lower point's densest neighbor. This causes a HOP path to traverse a void region, and more importantly, it causes the circled point below the void to be grouped together with points above the void. To clearly see why this happens and what the effect is, see figure 48. In figure 48(a) we see the raw data with the Voronoi cells. The red point is shown with its six neighbors. The densest of these six neighbors is at the point of the red arrow. So the HOP algorithm creates a HOP path which traverses this void region. The resulting max-classes are encircled in figure 48(b). While many of the max-classes are reasonably shaped, the thin vertical max-class that cuts across the void is the problematic result from HOP. In the raw data it appears that the point on the lower side of the void should be grouped with the points below the void.

This sort of "unnatural" grouping is a product of the fact that we are using HOP on the Delaunay graph (which is determined by the Voronoi cells). Many points on the boundary of a void will have Voronoi neghbors on the opposite side of the void, hence these points will have a Delaunay edge connecting them together. Thus it is possible that HOP will choose this edge if the densest neighbor of one of these points happens to be its neighbor on the other side of the void.

To avoid this possibility we considered methods for not allowing HOP to traverse a void like this. One solution we preferred is to use a tool that is in common use in topological data analysis called an alpha complex. An alpha complex is simply a subset of the Delaunay graph where the inclusion or exclusion of any particular edge depends on a single scale parameter α.

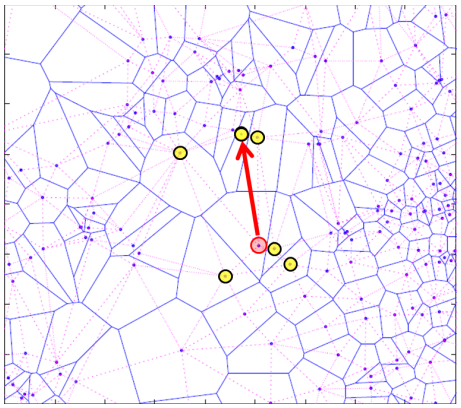


(a) A two dimensional set of raw data points.

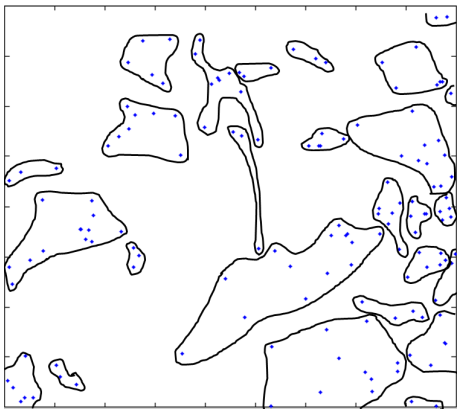


(b) The void region is shaded blue. The lower circled point "HOPs" to the upper circled data point.

Figure 47: The HOP algorithm causes the lower circled data point in figure (b) to be grouped with points that are across the void. This problem is a common effect of HOP.



(a) The raw data is shown with its Voronoi cells. The neighbors of the red point are shown in yellow.



(b) The max-classes resulting from HOP are encircled.

Figure 48: In (a) we see that the densest neighbor to the red point is the one at the head of the red arrow. So HOP jums across this void region. In (b) we see the resulting HOP partition into max-classes.

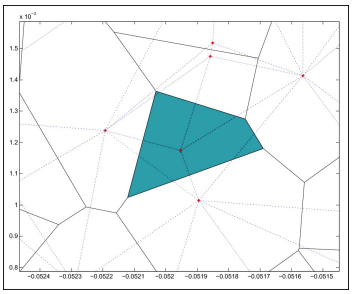


Figure 49: Voronoi Cell

7.3 Alpha complexes

The dual of the Voronoi diagram, as discussed before, is the Delaunay tesselation. A Delaunay edge, or a 1-cell, is automatically added when two Voronoi cells share a common wall (see figure 49).

Given the nature of our density function, when we apply the HOP algorithm, we sometimes get max class groupings that are unnatural upon visual inspection. To overcome such bias, which may be the direct result of our chosen density function, we implement an approach from computational geometry, called an α-complex to mitigate this unnatural grouping.

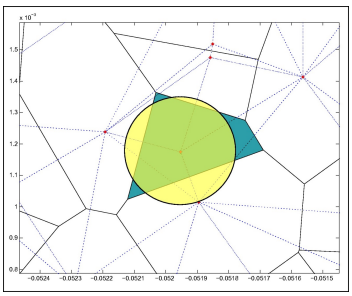


Figure 50: A Voronoi cell in blue and a disk of Radius α in yellow. The intersection of the two shapes is the α-shape.

To begin, we start with a Voronoi cell (the blue cell in figure 50). We then create a disk

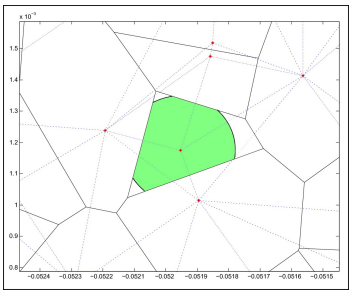


Figure 51: An α shape

of radius, α, centered at the vertex of the Voronoi cell (the yellow disk in figure 50). The α-shape is the resulting shape that is created from the intersection of the Voronoi cell and the disk of radius α.

We use α-shapes to build a simplicial complex called an α-complex. To create this complex, one starts with a Voronoi tesselation of point cloud data (Fig. 52).

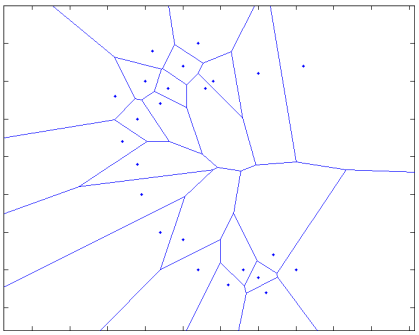


Figure 52: Voronoi Tesselation of Point Cloud Data

We then choose a particular value for α and form an alpha shape around *every* point in the data set. The value of α must be the same for every data point (an example with a small value of α is shown in figure 53).

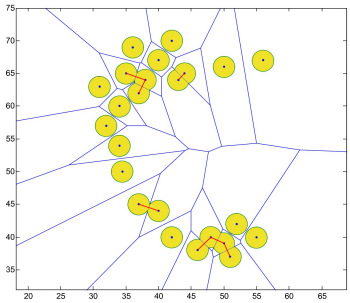


Figure 53: Disk of Radius α Created For All Voronoi Cells

When two α-shapes intersect (*i.e.* the share a wall) we add a 1-cell (an edge) connecting the two representative points for those α-shapes (see figure 54). If three α-shapes intersect (*i.e.* they have a non-empty pairwise intersection) then a 2-cell (triangle) is added with the three representative points as vertices. In general (in higher dimensions) if k α-shapes have pairwise non empty intersection (meaning they all share at least one point), then we add a $(k-1)$-cell with the k points as vertices. This process produces a complex of cells. In two dimensions the complex is made up of points, line segments and triangles; in three dimensions the complex is made up of points, line segments, triangles and tetrahedra, etc.

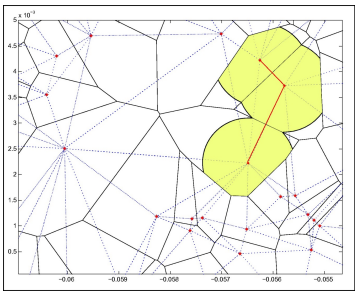


Figure 54: Alpha Complex (Note the Red Delaunay Edge Added to Intersecting α-shapes)

Note that although the radius of the disk must be the same for each Voronoi cell, the choice of radius can be varied. For $\alpha = 0$ the complex only consists of the isolated data points. As α increases more cells are added. Thus for different values of radius α, resulting α-complexes will be different from each other based on the initial value of our radius.

As the value of α is increased more cells are added. This comes from the fact that as the value of α (the radius of the disk) is increased, more α-shapes will intersect each other, thus automatically creating more Delaunay edge connections. If the value of α was set to ∞, the full Delaunay triangulation would be recovered.

Thus, the α-shape approach takes a collection of point cloud data and turns it into a subset of the full Delaunay triangulation, which we can analyze geometrically, noting that this subset gives different, and possibly useful, groupings than the full Delaunay triangulation.

7.4 HOP on alpha complexes

With the alpha complex tool in hand, we can revisit the problem with HOP that we discussed in section 7.2. In figures 55 through 62 you see the same data we saw in figure 47(a) earlier, but in this new series of pictures we see the growth of the α-complex on that data as α increases. For $\alpha = 0$ the α-complex would consist of only raw data points and none of the Delaunay edges would be added to the complex. But in figure 55 we see the resulting α-complex for a very small α. Only the shortest edges and smallest triangles are included. As α increases longer edges and larger triangles are added to the complex and eventually, if α is large enough, we recover the full Delaunay triangulation which you can see in figure 62.

Remember the problem was that HOP grouped together points that were on opposite sides of the void. The problem points that we discussed in section 7.2 can be seen in figure 55 where the red point is below the void region and the yellow point is above the void region. Remember that HOP will go from the red point to the yellow point if we apply the HOP algorithm to the *complete* Delaunay triangulation. But as you can see in figure 55, for small α, there is no edge between these two points. Note also that the edge between the red and yellow points does not appear until a relatively large value of alpha in figure 61. So if we apply HOP to any of the first six α complexes, the red point cannot HOP to the yellow point.

The HOP algorithm doesn't require the triangle 2-cells at all even though we are showing them in the figures. In order to do HOP we simply need the vertices and edges. So we only apply HOP to the 1-skeleton of the α-complex (and the same is true even for three dimensional data).

One possibility that we considered (which needs further consideration) is to use HOP on the α-complex for various values of α in order to study the results of HOP as α varies. HOP results for one such value of alpha is shown in figure 64. In figure 63(a) we see the full Delaunay triangulation which is equivalent to the 1-skeleton of an α-complex for arbitrarily large α. In figure 63(b) we see a subset of the Delaunay triangulation for a particular choice of α. The results of applying HOP to this complex is shown in figure 64. One interesting result is that the HOP minima, tend to fall on the boundary of the void regions. We feel that analyzing the HOP minima and maxima over all possible values of α may lead to methods that can identify which significant structures persist as α varies.

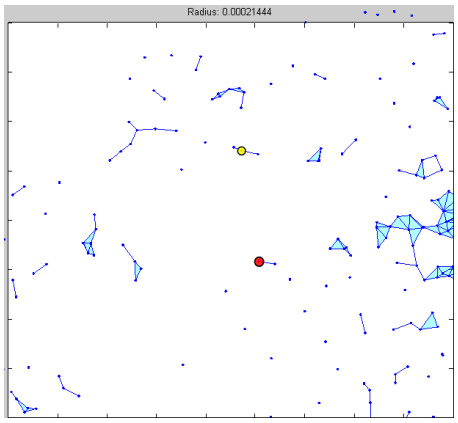


Figure 55

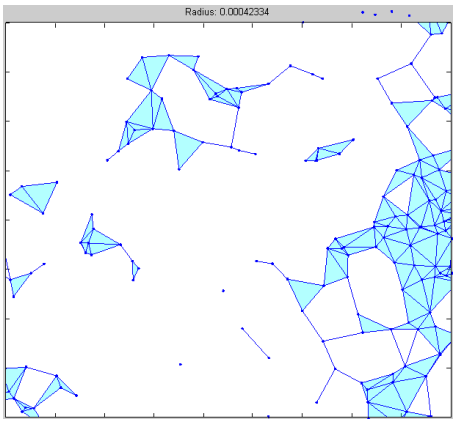


Figure 56

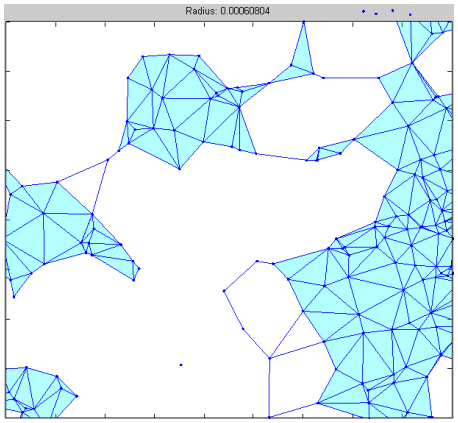


Figure 57

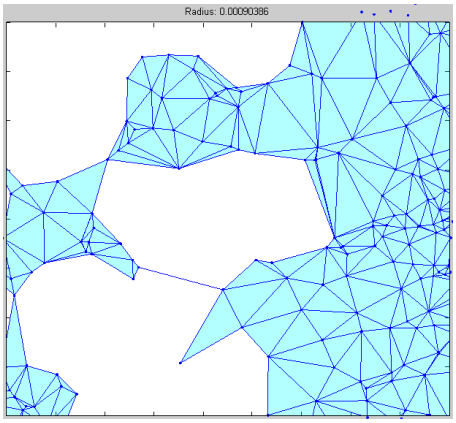


Figure 58

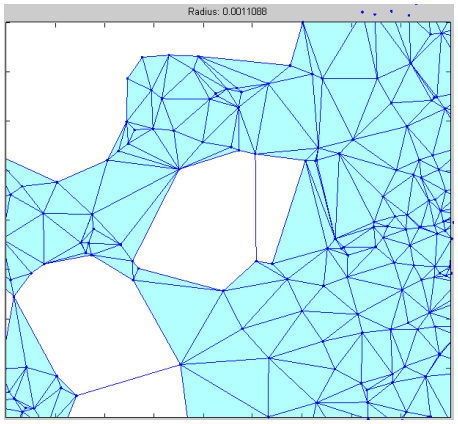


Figure 59

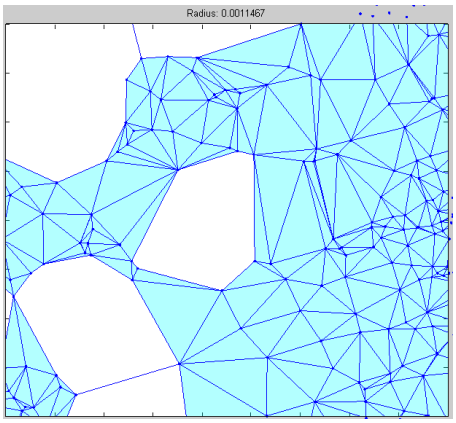


Figure 60

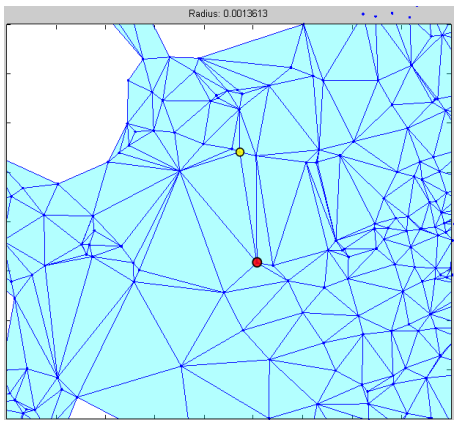


Figure 61

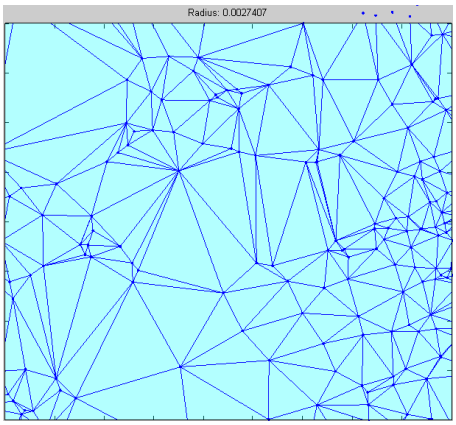
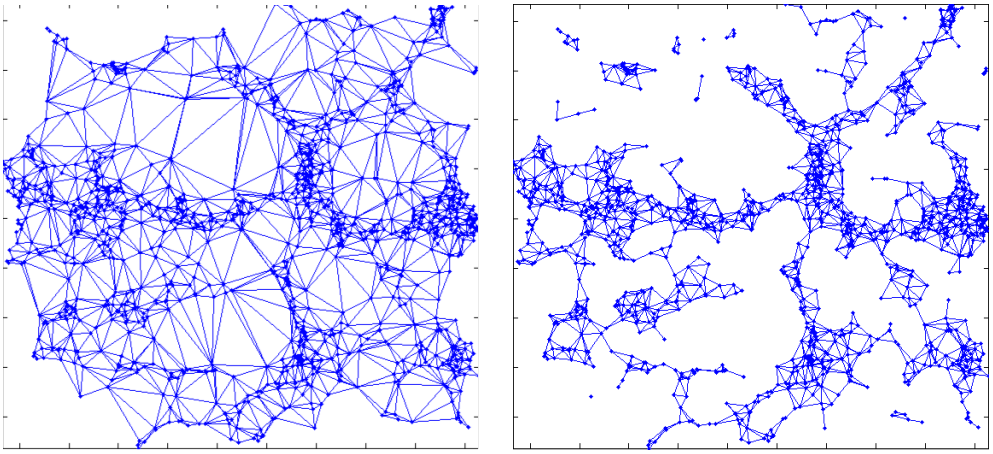


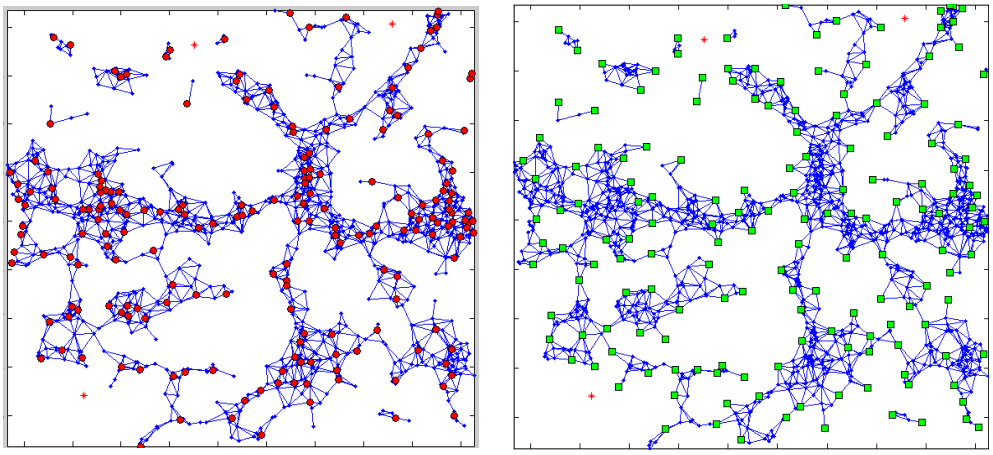
Figure 62



(a) Raw data is shown with its the full Delaunay triangulation.

(b) The 1-skeleton of an α-complex. This is a subset of the Delaunay triangulation.

Figure 63:



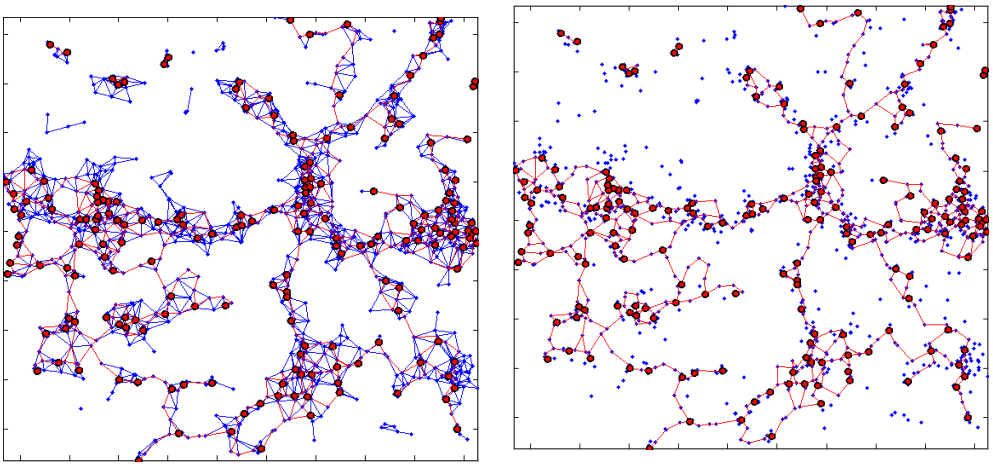
(a) HOP maxima are shown after applying hop to this α-complex. The red asterisks are isolated data points in the complex.

(b) HOP minima are shown on the same complex. Note that the minima primarily outline the boundary of the complex.

Figure 64: HOP minima and maxima on an α-complex for a particular value of α.

Finally, we can also apply the same technique of attaching neighboring maxima together via geodesics on the maxima seen in figure 64(a). The resulting geodesics are shown in figure 65.

Note that the results in figure 65(b) look very similar to what we saw in section 7.1. For example, see the geodesics in figure 43 on page 35. But there is a key difference between *how we got the results*. In the earlier example, we were calculating all geodesics on the *full Delaunay complex* and we were only plotting the geodesics that fell below a particular length threshold. But in the example in this section, we are plotting *all* geodesics (regardless of length) but the geodesics are limited to the edges in the α-complex. Since the alpha complex is a subset of the Delaunay complex, the geodesic paths must be contained in this subset. We feel that more study needs to be done to investigate the interplay between α scale parameter and the threshold parameter on the length of the geodesics. Although the results are similar in the example that we have shown in this section, we have not tested this enough to see if the results are similar in general.



(a) HOP geodesics on an α-complex.

(b) The same geodesics as in figure (a) but with the complex removed.

Figure 65: Geodesics on an α complex.

8 Centroidal Nudges

In most of the described approaches, we utilized a very specific type of tessellation, the Voronoi tessellation, to the extent that in addition to assuming the data to be representative of the underlying rules dictating the formation of our observation points, we, quite faithfully, trust the Voronoi tessellation to depict a proper protrayal of the region of space the data is observed to be embedded in. Although common sense dictates that the Voronoi tessellation, as well as the derivative assumption of the Voronoi density being a decent approximation of the appropriate HOP function, somewhat reflect the mechanics of the gravitational potential, which the data points are accepted to follow, we have not located satisfactory evidence, be it theoretical or physical, confirming the conclusion. In fact, certain phases in the procedure in its construction suggest the contrary, one being the spherical symmetry, another the universality that the Voronoi cells lack.

Therefore, if we desire enlightenment upon the problem of creating (via tessellation), grouping and classifying structures whose sole influences are the data points and a set of rules the data is believed to follow, the complete procedure should be

1. Create a tessellation.
2. Verify the tessellation.
3. Partition the tessellation.
4. (Possibly) group the partitions.
5. Classification.

Before describing a scheme for verifying the validity of the assumption of the Voronoi tessellation, consider the following conjectures.

Definition 40. Let V be the Voronoi tessellation of some finite $D \subseteq \mathbb{R}^n$ in which for all $d \in D$, $v_d \in V$ is a Voronoi cell based at d.
Let $\text{int}(V) = \{v \in V : v \cap CH(D) = v\}$. Let $\partial V = V \setminus \text{int}(V)$. Let $\text{int}(D) = \{d \in D : v_d \in D\}$. Let $\partial D = \{d \in D : v_d \in \partial V\}$.
Let $f_V : V \to \mathbb{R}^n \cup \{\infty\}$ map each Voronoi cell to its centroid.

Conjecture 41. *For all finite $D, E \subseteq \mathbb{R}^n$ with corresponding Voronoi tessellations V and W, if $\partial V = \partial W$ and $f_V[V] = f_W[W]$, then $V = W$.*

Conjecture 42. *Let $D, E \subseteq \mathbb{R}^n$ be finite with V and W their corresponding Voronoi tessellations. If*

- $\partial D = \partial E$, *and*
- $V = W$,

then $D = E$.

Conjecture 41, seemingly true, states that if we are given a data set, and draw its corresponding Voronoi tessellation, if we calculate the centroids of each of the Voronoi cells, and then somehow forget the Voronoi tessellation is formed on the interior of the data space, we can recover it completely by using the boundary and the centroids. This implies that in using the Voronoi tessellation, we may identify the cells with the centroids. Conjecture 42, appearing to have specific counterexamples that we have yet to encounter, states that if we are to forget the coordinates of the data points whose Voronoi cells are interior to the data space, we can recover them by using the Voronoi cells alone. This implies that we may identify the interior data points with the Voronoi cells.

Together, these provide us a well-defined notion of tracing the trajectories from data points to their Voronoi centroids, called Lloyd's algorithm. Although its general use in smoothing data is not a goal we wish to achieve, it bears a feature we covet in the verification for validity, the construction of a series of well-defined paths that exhibits resemblance to dynamical systems, which the underlying rules are a member of, when we recursively apply the procedure of

1. Nudge the points towards the centroids, and
2. Re-tessellate using the nudged points.

The resulting set of trajectories have the terminal (at infinitely many iterations) set of points being evenly spaced in the sense that if we Voronoi tessellate it, the resulting cells are regular hexagons in in $\mathbb{R}^2$. Note that in this process, we introduce an additional dimension, which we liken it to time to attempt to depict a more illustrative aspect. In $\mathbb{R}^3$, for example, we cast the data into $\mathbb{R}^4$, and we may think of the galaxies in the SDSS data devolving in time towards a more evenly-distributed set.

These trajectories are not very smooth by nature, so rather than nudging completely from data points to centroids, we may nudge them in smaller fractions, say, 0.01 of the distance between data points and their centroids. Since this appears to extend Lloyd's algorithm, we use a more illuminating name, centroidal nudges.

Utilizing these trajectories, the verification process is as simple as taking the coordinates of the points near equilibrium (when there is only one maximum) and feeding them into a computer simulation (with scales adjusted such that the size of the universe remains constant), then checking the degree of topological conjugacy between the centroidal trajectories and the trajectories output by the simulation (assuming that it is deterministic, as is the set of underlying rules regarding the evolution of the universe, in general).

Briefly sidetracking, a method that allows the circumvention of the remaining steps of the complete procedure in the interpretation of the data is to employ Morse theory to analyze this "evolution" via this artificial construct of Voronoi tessellation.

In the partitioning of the Voronoi cells, there are two ways we can progress while still maintaining the usage of centroidal nudges.

- If we can discover a consistent way to trace opposite of, rather than towards, the centroids, the resulting behavior is that points naturally cluster together. Thus, for partitioning, all that we are required is to perform such nudges, stop after a certain time, then determine the points to be in the same partition if they have coalesced into one. Note that to retain multiple points overlaying each other, we may require a weighted Voronoi tessellation.
- Determine the critical points via (an approximation of) the Brouwer fixed-point theorem, find the maxima, then use an iterative technique of both successively smaller balls and gradients provided by centroidal trajectories (an adaptation of stability in dynamical systems) to determine the maxima each point belongs to.

If we are to desire larger partitions for larger-scale structures, it is necessary to first determine which of the techniques of grouping is more desirable, complete or partial groups, i.e. completely merging one entire insignificant structure into only one more significant neighboring structure, or splitting up the insignificant structure into smaller pieces (in a way that makes physical sense), and merge each piece with its neighboring, more significant, structure. Here, we make the argument for partial groups.

Suppose that we have, an example in $\mathbb{R}^1$, with two extremely-defined clusters on two ends, and one sparser cluster, in the center. Suppose that we are operating under the rules of the universe, assuming that the underlying force is overall attractive. Then, in the scale of time, at an earlier period, the clusters were not well-defined, i.e. the points are further scattered apart. In particular, the center cluster remains less defined than the two on the sides. Eventually, after sufficient progression in reverse time, the middle set of data stops behaving (under HOP) as a cluster. Therefore, it makes little sense to completely group the entire middle set into only one of either left or right clusters.

If we are to accept the latter type of grouping, via partial groups, then centroidal nudges already has it built-in, by simply using the data set obtained upon performing a few iterations of the procedure, then perform the partitioning as before.

Lastly, to determine the type of structure a particular partition of data is, we take the set of points, then surround it by sufficiently-dense points lying on a sphere, evenly distributed. We run centroidal nudges on the data partition along with the spherical set of points until equilibrium, then compare the volume of the Delaunay simplices entirely in the interior of the sphere at equilibrium to the volume of the Delaunay simplices entirely in the interior of the sphere at the original data set. If sufficiently close to zero, then we flatten the data about its least-squares plane, then proceed the same as before, except using one-lower-dimensional sphere, progressing until we obtain a value sufficiently far from zero, or until zero dimensions. We can then cluster accordingly, with stopping at n-dimensions the n-clusters, stopping at $(n-1)$-dimensions the $(n-1)$-clusters, etc., down to 1-dimension, corresponding to filaments, and 0-dimensions, corresponding to voids.

A similar idea is that we can take the partition, along with all of its Delaunay simplices, put a ball of points around it, nudge the inside, then observe the simplices as they get formed or destroyed to move from the equilibrium to our data set.

9 Concluding Remarks

The α-geodesics approach combines two well-known computational topology algorithms – HOP and alpha-shapes – along with the novel notion of geodesics on HOP Max Classes. In doing so, we provide a computationally efficient generative model for the geometric topology of galactic point cloud data, that seems to correspond to existing research on the topology of real and simulated data on the formation of the cosmic web. In addition, the results of our approach have much visual and intuitive appeal. However, since the procedure uses HOP, we also inherit the drawbacks of the HOP algorithm.

9.1 Future directions

A possible future direction for this approach could involve assigning a code to each galaxy based on how many tetrahedra, triangles, and edges are connected to that galaxy, as α is varied α-complex. These codes could then be used to identify whether the galaxy lives in a cluster, filament, or void.

Another interesting question for research involved finding a way to compare and contrast the various algorithms to evaluate their output in a meaningful way.

Finally, other real and simulated data can be used to establish the usefulness and general applicability, or lack thereof, of the methods covered in this report.

Appendices

A DMT: Definitions & Theorems

In this appendix, we describe basic terminology and definitions in Morse theory.

Definition 43. A Morse function is a smooth underlying scalar function whose critical points are non-degenerate.

Definition 44. An integral line is a curve tangent to the gradient vector field of the Morse function.

Definition 45. A ascending manifold is the set of all integral lines emanating from a common minimum. The boundary of an ascending manifold in 2d contains maxima and saddles.

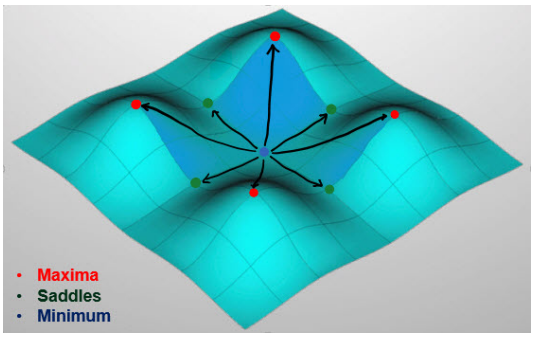


Figure 66: An ascending manifold

Definition 46. A descending manifold is the set of integral lines with a common maximum.

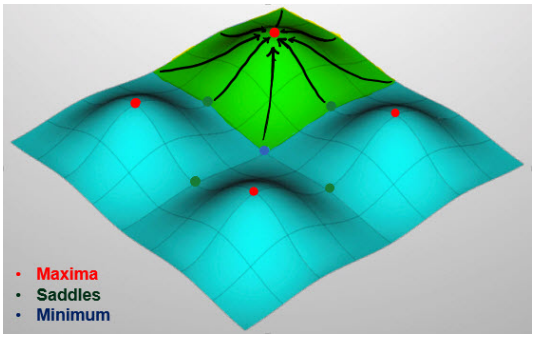


Figure 67: A descending manifold

Definition 47. A Morse-Smale cell is the intersection of an ascending manifold and a descending manifold. The Morse-Smale complex is the set of all Morse-Smale cells.

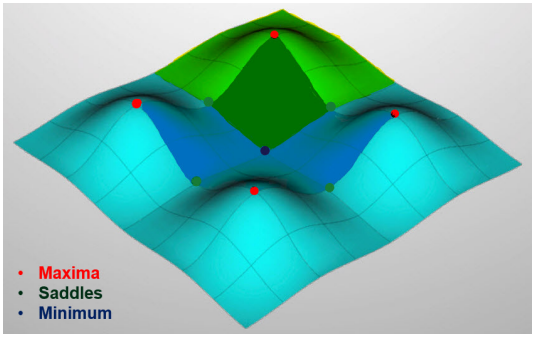


Figure 68: An intersecting ascending and descending manifold resulting in a Morse-Smale cell in green

Definition 48. A (finite) simplicial complex is a pair (V, K), where V is a finite set of verices and K is a collection of subsets of V such that:

1. $V \subset K$
2. If $\beta \in K$ and $\alpha \subset \beta$, then $\alpha \in K$.

K is called a simplicial complex and each element of K is called a simplex. If $\alpha \in K$ has $(p+1)$-vertices, we call α a p-simplex and $dim\alpha = p$. Geometrically, a vertex is a 0-dimensional simplex, an edge is a 1-dimensional simplex, a triangle is a 2-dimensional simplex and a tetrahedron is a 3-dimensional simplex.

Definition 49. Let K be a simplicial complex. A function $f : K \to \mathbb{R}$ is called a discrete Morse function if for every simplex $\alpha \in K$, we have:

1. There is **at most one** simplex $\beta \in K$ with $\alpha < \beta$ such that $f(\beta) \leq f(\alpha)$.
2. There is **at most one** simplex $\gamma \in K$ with $\gamma < \alpha$ such that $f(\gamma) \geq f(\alpha)$.

In other words, f assigns higher values to higher dimemsional simplices with at most one exception locally at each simplex.

Definition 50. A discrete vector field on a simplicial complex K is a set $X \subset K$ x K of pairs of simplices, such that for all $(\alpha, \beta) \in X$:

1. $\alpha < \beta$, and
2. Each simplex is in at most one pair.

If $(\alpha, \beta) \in X$, then $dim\beta = dim\alpha + 1$. We use an arrow from α to β to represent this discrete vector field X when $(\alpha, \beta) \in X$. We can then define a path analogous to the integral cruve of a smooth vector field in smooth Morse theory as follows

Definition 51. If X is a discrete vector field on K, then an X-path is a sequence

$$\mathscr{P} = (\alpha_0, \beta_0, \alpha_1, \beta_1, ..., \alpha_r, \beta_r, \alpha_{r+1})$$

of simplices such that:

1. $(\alpha_i, \beta_i) \in X$ for all $0 \leq i \leq r$;
2. $\alpha_{i+1} < \beta_i$ and $\alpha_{i+1} \neq \alpha_i$ for all i

This means that all α_i's have the same dimension and all β_i's have the same dimension, one greater than that of the α_i's. The simplices $\mathscr{P}$ alternate in dimension, starting and ending with a simplex of lower dimension. Note that every discrete Morse function $f : K \to \mathbb{R}$ defines a unique discrete vector field called the discrete gradient X_f of f.

Definition 52. A critical simplex is a simplex $\alpha \in K$ that is not paired with any other simplex. If α is not critical, then it eith has an adjacent simplex of higher dimension with lower density, or an adjacent simplext of lower dimension with higher density.

Theorem 53. *If $f : K \to \mathbb{R}$ is a discrete Morse function, then $\mathscr{P} = (\alpha_0, \beta_0, \alpha_1, \beta_1, ..., \alpha_r, \beta_r, \alpha_{r+1})$ is an X_f-path if and only if*

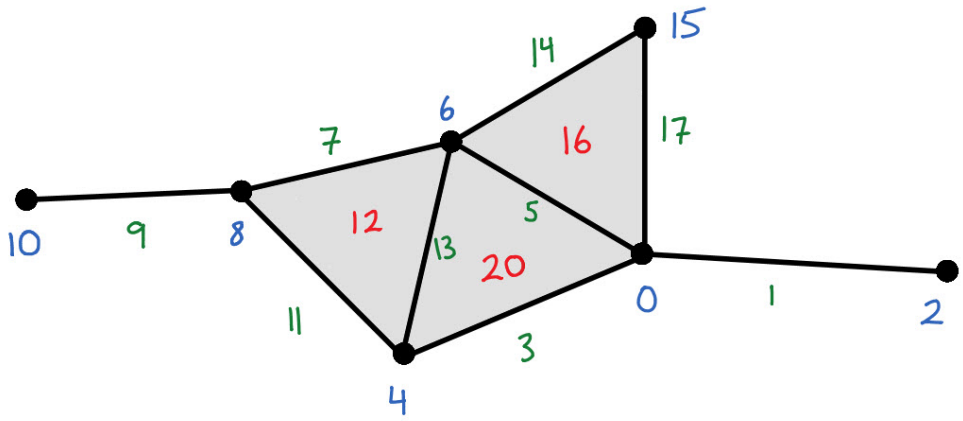
1. *$\alpha_i < \beta_i$ and $\alpha_{i+1} < \beta_i$ for all i; and*
2. *$f(\alpha_0) \geq f(\beta_0) > f(\alpha_1) \geq f(\beta_1) > ... \geq f(\beta_r) > f(\alpha_{r+1})$.*

A sequence of simplices $\mathscr{P}$ is a gradient path if and only if f decreases along $\mathscr{P}$.

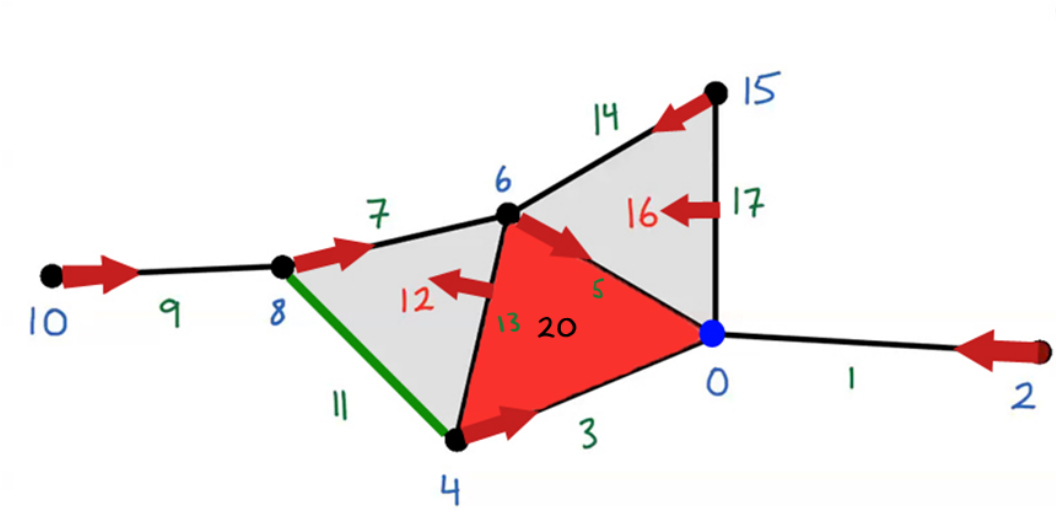
Theorem 54. *Let X be a discrete vector field on a simplicial complex K. Then X is the discrete gradient of some discrete Morse function if and only if X has no closed paths.*

Theorem 55. *Let $f : K \to \mathbb{R}$ be a discrete Morse function. If α and β are critical simplices with $dim\beta = dim\alpha + 1$, and there is exactly one gradient path from a simplex on the boundary, then α and β can be cancelled in the sense that there exists another discrete Morse function $\widetilde{f} : K \to \mathbb{R}$ which has the same critical simplices as f except that both α and β are no longer critical for $\widetilde{f}$.*

Example. Given simplices as shown in the picture below



Each simplex (vertex, edge, or triangle) is assigned with a number corresponding to its density. We can construct gradient paths that flow from high-density simplices to low-density simplices as described in the definitions. As the result, we obtain X_f-paths shown in the figure below



Recall that a critical simplex is a simplex that is not paired with any other simplex. Therefore, the critical simplices in this example include the blue vertex (0-dimension simplex), the green edge (1-dimension simplex) and the red triangle (2-dimension simplex). Note that $X - f$-paths are not unique. For instance, both paths (17,16,5) and (17,16,14) are allowed.

B Graph theory: a primer

In this appendix, we recall basic terminology, defintions and theorems in graph theory.

A graph can be defined in a variety of equivalent ways. A general graph can be defined theoretically as follows.

Definition 56 (general graph and related terms)**.** A ***graph***, G, consists of a countable ***vertex set***, $V(G) = \{v_1, v_2, v_3, \ldots\}$ and a countable ***edge set***, $E(G) = \{e_1, e_2, e_3, \ldots\}$, along with a relation, ϕ, which assigns to each edge $e \in E(G)$ a non-ordered pair of vertices $v_i, v_j \in V(G)$. We call v_i and v_j the ***endpoints*** of e. Note that it is possible for the endpoints of e to be equal (that is, we allow $v_i = v_j$). Also $V(G)$ and $E(G)$ are allowed to be empty. If a vertex v is the endpoint of e, then we say that v and e are ***incident***. If $V(G)$ and $E(G)$ are finite sets then G is a ***finite graph***. The ***order*** of a graph is the number of vertices in G. Thus the order of G is equal to $|V(G)|$. The ***size*** of G is often denoted $|G|$. (sometimes we denote the size by $|V|$ when it is clear which graph contains V). The ***size*** of a graph is equal to the number of edges in G. So the size of G equals $|E|$).

For our purposes we will only be working with finite graphs. Furthermore, in practice, one rarely refers to the relation ϕ since it is usually obvious in context. Instead we simplify the notation and think of a graph $G = (V, E)$ as a pair consisting of it's vertex set and edge set. Also, one often further simplifies (and abuses) notation by stating things like: "Let V be a subset of G," or "let $E \subset G$," or "consider the vertex $v \in G$." These are technically incorrect but the intent should be clear in context.

This set theoretic definition is needed for proving things about graphs, but we usually visualize a graph by drawing. When a drawing is feasible, we often draw a graph by representing vertices with dots and drawing line segments (or curves) to represent the edges. These segments or curves terminate at the endpoints of the edge it is representing.

Definition 57 (simple graph)**.** A ***loop*** is an edge that is incident to only one vertex (that is, both of its endpoints are attached to the same vertex). ***Multiple edges*** are two or more edges which all share the same endpoints. A ***simple graph*** is a graph that does not have any multiple edges or loops.

Definition 58 (adjacency)**.** When two vertices, v_1 and v_2, are endpoints of the same edge, e, we say that these vertices are ***adjacent*** or that they are ***neighbors***. We will denote this adjacency pair $v_1 \leftrightarrow v_2$. The ***neighborhood*** of a vertex v, denoted $N(v)$, is the set of all vertices that are adjacent to v.

When a graph does not have any multiple edges the vertex pair determines the edge. So in that case, we can unambiguously write $e = v_1 \leftrightarrow v_2$. The choice of notation depends on how much we want to emphasize the endpoints of e.

A particular graph can be drawn in many different ways. But regardless of how you *draw* a graph, the graph itself is unchanged. That is, the vertices and edges, as defined in definition 56 above, are unchanged. *Also* note that the labels we give for the vertices and edges do not matter. It also doesn't matter how we depict vertices and edges in our picture. The key information that one needs to know in order to determine a graph are:

1. A list of vertices.
2. The adjacency information of each vertex in the list.

Of course to clearly define the second fact we need to label the vertices. But this labeling is arbitrary as long as each vertex has a unique label. So, for example, if you start with a particular graph G, any relabeling or redrawing of G, which preserves the adjacency information, results in a graph that is the same as G. Thus we have our notion of "sameness" in graph theory.

Definition 59. (isomorphism of graphs) Let G and H be graphs. An ***isomorphism*** from G to H is a bijective map, $f : G \to H$, which maps $V(G) \mapsto V(H)$ and maps $E(G) \mapsto E(H)$ and also preserves the adjacency relationship in the following way: if $e \in E(G)$ with endpoints u and v, then $f(e) \in E(H)$ with endpoints $f(u)$ and $f(v)$.

One can think of this definition as simply a careful set theoretic definition of of a very simple idea: relabeling vertices and/or redrawing a graph doesn't change the graph. For simple graphs (no loops, no multiple edges) isomorphism is even easier to define since a pair of vertex endpoints determine their edge.

Definition 60. Let G and H be *simple* graphs. An ***isomorphism*** from G to H is a bijective function, $f : V(G) \to V(H)$ such that if $u \leftrightarrow v \in G$ then $f(u) \leftrightarrow f(v) \in H$.

Definition 61 (degree)**.** For a simple graph, the ***degree*** of a vertex v, is the number of edges incident to v. The degree of v is denoted $d(v)$. If $d(v) = 0$ we say that v is an ***isolated vertex***.

Definition 62 (subgraph)**.** A ***subgraph*** of G is a graph H that is a subset of G. This subgraph is denoted $H \subseteq G$ and we sometimes say that "G contains H."

Note that we cannot define a subgraph as simply a subset of G. Since we demand H to be a *graph* which is *also* a subset of G, then $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ with the further stipulation that if $e \in H$, then the endpoints of e are also in H (or else H would not be a graph at all). So even though we use the notation $H \subseteq G$, H is more than just a subset. So if you include an edge in a subgraph H, you are forced to include the vertices in G at it's endpoints. But a subgraph need not have any edges at all.

For many applications and theorems we need a more restrictive notion of subgraph. We are often interested in graphs obtained by deleting vertices from a graph. But let us first discuss deleting edges which is simpler. When we delete an edge e from a graph G we denote the resulting graph $G - e$. If we instead wish to delete a collection of edges, say $F \subseteq E(G)$, we denote the result $G - F$. Note that deleting edges from a graph does not in any way alter the elements in $V(G)$. But the same is not true for deleting vertices. When we delete v from G, denoted $G - v$, we are forced to delete any edges which are incident to v or else $G - v$ would not be a graph. An equivalent statement can be made if we delete a set of vertices, W, from G to obtain the subgraph, $G - W$. So deleting a vertex from G *does* alter the edge set $E(G)$ (unless v was an isolated vertex in which case E is unchanged).

Another basic subgraph of G is an edge e, along with it's endpoints. We are often interested in traversing a graph by hopping from vertex to vertex along edges, that is, hopping from one vertex to an adjacent vertex, and so on. This is essentially equivalent to stringing together edges which is also a useful subgraph.

Definition 63. A ***walk*** which starts at a vertex, v_0, and ends at a vertex v_k is a subgraph, $W \subseteq G$, that can be written as an ordered alternation of vertices and edges:

$$W = (v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k),$$

where each e_i is incident to v_{i-1} and v_i, (which also means that v_i is adjacent to v_{i+1}). For a simple graph, a walk is completely determined by the sequence of vertices alone (since if v_i is adjacent to v_{i+1} in the walk, then there is a unique edge incident to both of them). So, in this case, we may simply write:

$$W = (v_0, v_1, v_2, \ldots, v_k),$$

The ***length*** of a walk, denoted $\ell(W)$, is the number of edges in the walk. Thus, $\ell(W) = k$ above. A ***closed walk*** is a walk where the starting vertex equals the ending vertex. That is $v_0 = v_k$.

Note that vertices and edges in a walk need not be unique. One is allowed to repeatedly traverse the same edge any number of times. Also, when we want to emphasize the starting point, u, and the endpoint, v, of a walk, one refers to the particular walk as a u, v-walk.

Definition 64. A ***trail*** is a walk where no edge is repeated (but vertices can be repeated). To emphasize the starting point and endpoint by calling it a u, v-trail.

Definition 65. A ***path*** is a walk where no vertex is repeated. To emphasize the starting point and endpoint by calling it a u, v-path.

Since no vertex is repeated in a path, then no edge is repeated either. Thus a path is a special case of a trail.

Definition 66. A ***cycle*** is a closed walk where no interior vertex is repeated.

So a loop is a cycle of length one. A cycle of length two is a multi-graph with two vertices and two distinct edges incident to these two vertices. Any cycle, C, with $\ell(C) \geq 3$ can be drawn in the plane as the outline of a polygon. Also note that if C is a cycle of length k, and e is an edge in C, then $C - e$ is a path of length $k - 1$.

Paths are essentially the graph theoretic analog of a curve and the length of a path is essentially the analog of measuring distance on a graph. So not surprisingly, paths are precisely the right tool for defining connectedness of graphs.

Definition 67. A graph, G, is ***connected*** if for every pair of vertices $u, v \in G$ there exists a u, v-path. A graph is ***disconnected*** if there exists a pair u and v for which there is no u, v-path. A ***component*** of a disconnected graph, G, is a maximal connected subgraph of G, and G is the union of its components.

Proposition 68. *Define a relation on pairs of vertices $u, v \in V(G)$ by declaring that $u \sim v$ if there exists a u, v-path. Then this relation is an equivalence relation on V which partitions G into its connected components.*

Proof. **Reflexive** Let $v \in V(G)$.

Consider a walk W = (v).

v is the only vertex in W, and it is not repeated.

$\therefore$ W is a walk with no vertices repeated; thus W is a path, and $v \sim v$

-

Symmetric Suppose $v_0 ... v_k \in V(G)$, $v_0 \sim v_k$

So there exists a path P = (v_0, e_0, v_1, ... e_k, v_k).

P is a path, so $\forall\ i \in$ [0,k], v_i is adjacent to v_{i+1}.

Edge relations are symmetric, so $\forall i \in$ [0,k], v_{i+1} is adjacent to v_i.

Consider the walk Q = (v_k, e_k, v_{k-1}, ... e_0, v_0).

By definition of path, no vertices are repeated in P = (v_0, e_0, v_1, ... e_k, v_k), so no vertices are repeated in Q = (v_k, e_k, v_{k-1}, ... e_0, v_0) since they share the same elements.

$\therefore$ Q is a walk with no repeated vertices, so Q is a path, and thus $v_k \sim v_0$

-

Transitive Suppose $t, u, v \in V(G)$, $t \sim u$, $u \sim v$.

So there exists paths P = $(t...u)$ and Q = $(u...v)$.

Case 1: P $\cap$ Q = {u} (i.e. Excluding u, P and Q are disjoint).

P repeats no vertices, and Q repeats no vertices, thus the walk R = $(t...u...v)$ repeats no vertices.

$\therefore$ R = $(t...u...v)$ is a path, thus $t \sim v$.

Case 2: P$\backslash\{u\}$ $\cap$ Q$\backslash\{u\} \neq \emptyset$ (i.e. Excluding u, P and Q still share at least one vertex).

Let $x \in$ P $\cap$ Q be the first element of Q = $(u, ...x, ..., v)$ to appear in P = $(t...x...u)$.

Since Q is a path, no elements are repeated in $(x, ..., v) \subset Q$; similarly, P is a path, so no elements are repeated in $(t, ..., x) \subset P$.

Since x is the first element of Q to appear in P, then $(t...x) \cap (x, ..., v) = \{x\}$.

$\therefore$ R = $(t, ..., x, ..., v)$ is a connected walk with no elements repeated: so R is a path, and $t \sim v$.

□

In our work we will also need the following two definitions regarding simple graphs (*i.e.* no loops and no multi-edges).

Definition 69 (complete graph)**.** A ***complete graph*** on n vertices is a simple graph of order n such that every pair of vertices are adjacent. A complete graph is denoted K_n.

Definition 70. (complement graph) Given a simple graph $G = (V, E)$ of order n (meaning $|V| = n$), the ***complement*** of G is the graph $\overline{G} = K_n - E$. It is equivalent to say that $\overline{G}$ has the same vertex set as G but every adjacent pair of vertices in G are not adjacent in $\overline{G}$ and every non adjacent pair in G are adjacent in $\overline{G}$.

Thus, you obtain $\overline{G}$ from G by deleting all of the edges that are in G and adding all of the possible edges that were not in G.

Often, while studying a graph G we are interested in a certain special subgraph of G which contains a sub-collection of the vertices of G. Suppose we have a collection of vertices in G which we will call W. Thus, $W \subseteq V(G)$. Furthermore, suppose we are interested in the subgraph of G which contains W and any edge in G that has endpoints in W.

Now we have what we need to define an important type of subgraph that we will need for our work.

Definition 71 (induced subgraph)**.** An ***induced subgraph*** of a graph G, is a subgraph obtained by deleting a set of vertices from G. But we typically determine an induced subgraph in the following way: Let $G = (V, E)$ and let $W \subseteq V$. (So W is any subset of the vertices of G.) The ***subgraph induced by*** W is the graph $G[W] = G - W'$ where W' is the complement vertex set $W' = V(G) \setminus W$.

Definition 72. Let G be a connected graph. A ***cut edge*** is a single edge $e \in E(G)$ such that $G - e$ is disconnected. A ***disconnecting set*** for G is any set of edges $F \subseteq E(G)$ such that $G - F$ is disconnected.

Definition 73. An ***edge cut*** F for the graph $G = (V, E)$, is a set of edges of the form $[S, S']$ where $S \subset V$ and S' is the complement $S' = V(G) \setminus S$.

Clearly if we delete $F = [S, S']$ from G, then the induced subgraph $G[S]$ is disconnected from $G[S']$.

Definition 74. A ***bond*** is a minimal non-empty edge cut. One edge in this bond is called a ***bond edge***.

So a bond is the smallest set of edges you need to delete from G in order to separate a partition of the vertices of G.

Definition 75. A ***directed graph*** (or ***digraph***) G is a graph where the edges are ordered pairs (instead of unordered pairs as in definition 56). A ***directed edge***, e, with endpoints u and v is an ordered pair (u, v) where we will denote the directed edge $e = u \to v$. Since u points to v, then we call u the ***tail*** and v the ***head*** of e. The ***underlying*** graph of G is the undirected graph obtained by removing the directions on the edges, that is, replacing ordered pairs with unordered pairs in $E(G)$.

The definitions for ***directed walk***, ***directed trail***, ***directed path***, and ***directed cycle*** are the same as before except that one must now follow the direction of the edges, tail-to-head, while traversing the walk, trail, path, or cycle.

Definition 76. A ***digraph*** G is ***strongly connected*** if for each pair $u, v \in V(G)$ there exists a *directed* u, v-path from u to v. G is ***weakly connected*** if its underlying graph is connected (in the sense of definition 67).

References

[1] J Richard Bond, Lev Kofman, and Dmitri Pogosyan. How filaments are woven into the cosmic web. *arXiv preprint astro-ph/9512141*, 1995.

[2] D. Eisenstein and P. Hut. Hop: A new group-finding algorithm for n-body simulation. *Astrophysics Journal*, 498:137–142, 1998.

[3] Donald G York, J Adelman, John E Anderson Jr, Scott F Anderson, James Annis, Neta A Bahcall, JA Bakken, Robert Barkhouser, Steven Bastian, Eileen Berman, et al. The sloan digital sky survey: Technical summary. *The Astronomical Journal*, 120(3):1579, 2000.

[4] T. Sousbie. The persistent cosmic web and its filimentary structure i: Theory and implementation. *Mon. Not. R. Astron. Soc.*, 2010.

[5] JP Huchra and MJ Geller. Groups of galaxies. i-nearby groups. *The Astrophysical Journal*, 257:423–437, 1982.

[6] William H Press and Paul Schechter. Formation of galaxies and clusters of galaxies by self-similar gravitational condensation. *The Astrophysical Journal*, 187:425–438, 1974.

[7] Chung-Hsing Hsu et al. Haloes gone mad: The halo-finder comparison project. *Monthly Notices of the Royal Astronomical Society*, 415(3), 2011.

[8] Mark C. Neyrinck. Zobov: a parameter-free void-finding algorithm, 2008.

[9] Erwin Platen, Rien Van De Weygaert, and Bernard JT Jones. A cosmic watershed: the wvf void detection technique. *Monthly notices of the royal astronomical society*, 380(2):551–570, 2007.

[10] Jörg M Colberg, Frazer Pearce, Caroline Foster, Erwin Platen, Riccardo Brunino, Mark Neyrinck, Spyros Basilakos, Anthony Fairall, Hume Feldman, Stefan Gottlöber, et al. The aspen–amsterdam void finder comparison project. *Monthly Notices of the Royal Astronomical Society*, 387(2):933–944, 2008.

[11] Miguel A Aragón-Calvo, Bernard JT Jones, Rien Van De Weygaert, et al. The multiscale morphology filter: identifying and extracting spatial patterns in the galaxy distribution. *arXiv preprint arXiv:0705.2072*, 2007.

[12] John Willard Milnor. *Morse theory*. Number 51. Princeton university press, 1963.

[13] R. Forman. A user's guide to discrete morse theory. *Seminaire Lotharingien de Combinatoire*, 48, 2002.

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

[15] Willem Egbert Schaap. *The delaunay tessellation field estimator*. PhD thesis, Ph. D. thesis, Groningen University, 2007.

[16] Thomas Lewiner, Helio Lopes, and Geovan Tavares. Applications of forman's discrete morse theory to topology visualization and mesh compression. *Visualization and Computer Graphics, IEEE Transactions on*, 10(5):499–508, 2004.

[17] Caren Marzban and Ulvi Yurtsever. Baby morse theory in data analysis. In *Proceedings of the 2011 workshop on Knowledge discovery, modeling and simulation*, pages 15–21. ACM, 2011.

[18] Attila Gabor Gyulassy. *Combinatorial construction of Morse-Smale complexes for data analysis and visualization*. ProQuest, 2008.

[19] Stephen Skory, Matthew J Turk, Michael L Norman, and Alison L Coil. Parallel hop: A scalable halo finder for massive cosmological data sets. *The Astrophysical Journal Supplement Series*, 191(1):43, 2010.

[20] Brad Jackson, Jeffrey D Scargle, David Barnes, Sundararajan Arabhi, Alina Alt, Peter Gioumousis, Elyus Gwin, Paungkaew Sangtrakulcharoen, Linda Tan, and Tun Tao Tsai. An algorithm for optimal partitioning of data on an interval. *Signal Processing Letters, IEEE*, 12(2):105–108, 2005.

[21] Henry King, Kevin Knudson, and Neža Mramor. Generating discrete morse functions from point data. *Experimental Mathematics*, 14(4):435–444, 2005.

[22] Oliver Hahn, Cristiano Porciani, C. Marcella Carollo, and Avishai Dekel. Properties of dark matter haloes in clusters, filaments, sheets and voids, 2006.

[23] Rien van de Weygaert, Gert Vegter, Herbert Edelsbrunner, Bernard JT Jones, Pratyush Pranav, Changbom Park, Wojciech A Hellwing, Bob Eldering, Nico Kruithof, EGP Bos, et al. Alpha, betti and the megaparsec universe: On the topology of the cosmic web. In *Transactions on Computational Science XIV*, pages 60–101. Springer-Verlag, 2011.

[24] Bridget L Falck, Mark C Neyrinck, and Alexander S Szalay. Origami: Delineating halos using phase-space folds. *The Astrophysical Journal*, 754(2):126, 2012.

[25] Dominique Aubert, Christophe Pichon, and Stephane Colombi. The origin and implications of dark matter anisotropic cosmic infall on haloes. *Monthly Notices of the Royal Astronomical Society*, 352(2):376–398, 2004.

[26] Erwin Platen, Rien van de Weygaert, Bernard JT Jones, Gert Vegter, and Miguel A Aragón Calvo. Structural analysis of the sdss cosmic web–i. non-linear density field reconstructions. *Monthly Notices of the Royal Astronomical Society*, 416(4):2494–2526, 2011.