Level Set Tree Methods

Jussi Klemelä*

Article Type:

Advanced Review

Abstract

Level set trees provide a tool for analyzing multivariate functions. Level set trees are particularly efficient in visualizing and presenting properties related to local maxima and local minima of functions. Level set trees can help statistical inference related to estimating probability density functions and regression functions, and they can be used in cluster analysis and function optimization, among other things. Level set trees open a new way to look at multivariate functions, which makes the detection and analysis of multivariate phenomena feasible, going beyond one- and two-dimensional analysis.

*jussi.klemela@gmail.com

GRAPHICAL TABLE OF CONTENTS

INTRODUCTION

Level set tree methods provide tools to study properties of a multivariate function $f : \mathbf{R}^d \to \mathbf{R}$ with the help of mapping

$$\lambda \mapsto \Lambda(f, \lambda),$$

which assigns level sets

$$\Lambda(f, \lambda) = \{x \in \mathbf{R}^d : f(x) \geq \lambda\}$$

to the levels $\lambda \in \mathbf{R}$. Sometimes "level set" is used to denote the contour $C(f, \lambda) = \{x \in \mathbf{R}^d : f(x) = \lambda\}$. Set $\Lambda(f, \lambda)$ is sometimes called "superlevel set".

A level set tree of function $f : \mathbf{R}^d \to \mathbf{R}$ is a tree whose nodes are identified with the connected components of the level sets $\Lambda(f, \lambda)$ of f. A child-parent relation of a level set tree corresponds to the set inclusion. The root of the tree is the support of f. A finite number of levels λ is necessary in order to obtain a tree with a finite number of nodes, although further regularity conditions are needed to guarantee a finite number of nodes. (For example, the indicator function of rationals leads to an infinite number of nodes.

A level set tree can be defined recursively: (1) The root of the tree corresponds to the support of the function. If the support is not a connected set, then there are several roots and we define a separate level set tree for each root. (2) The child nodes of a given node m are obtained from the level set whose level is one step above the level of the node m. We need to restrict ourselves to the part of the level set which is a subset of the set associated with node m. This subset is decomposed into connected components and these connected components are the child nodes of node m. The leaf nodes of a level set tree correspond to the local maxima of the function.

We use the term "level set tree" to denote a decorated tree, whose nodes are associated with levels and with subsets of $\mathbf{R}^d$ (connected components of level sets). Thus, when a function is represented with a level set tree, the only loss of information comes from the fact that only a finite number of levels is used. The recursive process of building a level set tree is illustrated later in Figure 1, where a contour plot is associated with a level set tree.

Level set trees help to detect and visualize such features of a function as the number, sizes, and locations of the local maxima. To study local minima of function f we can construct a level set tree for $-f$, or study the tree of sublevel sets $\{x \in \mathbf{R}^d : f(x) \leq \lambda\}$. It may also be useful to study the mapping $\lambda \mapsto C(f, \lambda) = \{x \in \mathbf{R}^d : f(x) = \lambda\}$, since this mapping gives information simultaneously about local minima and maxima.

Level set tree methods have been used by separate communities, in the study of statistics, machine learning, computational topology, topological data analysis, scientific visualization, and function optimization. Typically different communities are not aware of the work which has been done by another community. We try to provide some references which go beyond statistical literature, although we concentrate on those applications of level set trees where statistically interesting functions like probability density functions and regression functions are analyzed.

The figures of the article can be reproduced using the instructions and software in http://jussiklemela.com/art/wires.

Visualizing Functions

A level set tree is a tree whose nodes are associated with connected components of level sets. Visualizations are obtained by utilizing some information about the connected components, like levels, volumes, and barycenters.

Tree Plots

Level set trees visualize the complete tree structure of the level sets. In a tree plot the nodes correspond to the connected components of a level set, height of a node correspond to the level λ of the level set, and the parent-child relations are indicated by lines connecting the nodes.

Volume Functions

The nodes of a level set tree can be decorated with information about the properties of the set which is associated to the node. A volume function is obtained by decorating a node with

the volume of the associated set. A volume function is a one-dimensional function, whose level set tree has the same tree structure as the original d-dimensional function. In addition, the connected components of the level sets of a volume function have the lengths equal to the volumes of the corresponding connected components of the original d-dimensional function. Klemelä (2004*b*)

Adding some notation, let the original function $f : \mathbf{R}^d \to \mathbf{R}$ have level sets $\Lambda_1, \ldots, \Lambda_N \subset \mathbf{R}^d$. Then the nodes of the level set tree correspond to connected components $A_1, \ldots, A_M$ of these level sets, where $M \geq N$, because each level set Λ_i can be written as a union of some sets A_i. Now the volume function $\mathrm{volume}(f) : \mathbf{R} \to \mathbf{R}$ is such that the level set tree of $\mathrm{volume}(f)$ is otherwise the same as the level set tree of f, but the nodes are associated with intervals $B_1, \ldots, B_M \subset \mathbf{R}$. We have always

$$\mathrm{volume}(A_i) = \mathrm{volume}(B_i),$$

where the volume of a set is the Lebesgue measure of the set, and in particular, the volume of an interval is the length of the interval. Since the level set tree of $\mathrm{volume}(f)$ and f are the same, it holds that

$$A_i \subset A_j \text{ if and only if } B_i \subset B_j$$

and

$$A_i \cap A_j = \emptyset \text{ if and only if } B_i \cap B_j = \emptyset,$$

which cover the two possible relations between the sets associated to the nodes of a level set tree.

The volume function $\mathrm{volume}(f) : \mathbf{R} \to \mathbf{R}$ visualizes the largeness of the modes of the function $f : \mathbf{R}^d \to \mathbf{R}$, in addition to showing the mode structure. The largeness of the modes is defined using the concept of excess mass of Hartigan (1987) and Müller and Sawitzki (1991). Namely, it holds that

$$\int_A (f - \lambda) = \int_B (\mathrm{volume}(f) - \lambda),$$

where $A \subset \mathbf{R}^d$ is a connected component of level set $\Lambda(f, \lambda)$ and $B \subset \mathbf{R}$ is the set associated to a node of the level set tree of $\mathrm{volume}(f)$, B being such that the node of B corresponds to

the node of A. Since level set trees of f and volume(f) have the same structure, there is a bijective correspondence between the nodes of these trees.

Since $f(x) \geq \lambda$ for $x \in A \subset \Lambda(f, \lambda)$, the integral $\int_A (f - \lambda)$ is equal to the volume of the area

$$\big\{(x, y) \in \mathbf{R}^{d+1} : x \in A,\, y \leq f(x)\big\}.$$

This is the area that is delineated by the hyperplane $y = \lambda$ and the restriction of the graph of f to A. Note that the excess mass over λ can be defined as the volume of the area

$$\big\{(x, y) \in \mathbf{R}^{d+1} : f(x) \geq \lambda,\, y \leq f(x)\big\},$$

but we are more specifically interested in the excess mass associated with the connected components A of the level set $\Lambda(f, \lambda)$.

Barycenter Plots

A barycenter plot is otherwise similar to a simple tree plot, but now the x-coordinate of a node is equal to the ith coordinate of the barycenter of the connected component of a level set which corresponds to the node. There are d different barycenter plots, each plot corresponding to the choice of the coordinate $i = 1, \ldots, d$. Klemelä (2004*b*)

The barycenter of set $A \subset \mathbf{R}^d$ is the center of mass of A, defined as $\int_A x\, dx / \int_A dx \in \mathbf{R}^d$. The barycenter is the mean of the uniform distribution on A.

For the integral $\int_A x\, dx$ to be well defined, the boundedness of A is a sufficient condition. It is also a necessary condition, in the sense that the integral is not defined, if there does not exist a ball $B_r(\mu) = \{x : \|x - \mu\| \leq r\}$ such that $A \setminus B_r(\mu)$ has measure zero.

A 2D Example of Function Visualization

Figure 1 shows six different ways to visualize a 2D function. Panel (a) shows a perspective plot, panel (b) shows a contour plot, panel (c) shows a tree plot, panel (d) shows a plot of a volume function, panel (e) shows a barycenter plot of the first coordinate, and panel (f) shows a barycenter plot of the second coordinate. Each visualization has its advantages and its disadvantages. A perspective plot shows only one side of the function, and we need to look at the function from several directions. A contour plot does not give visual information

about the levels of the contours. (Which local extremes are local maxima and which are local minima?) A tree plot and a plot of the volume function do not provide information about the spatial location of the local maxima, but the volume function provides information about the sizes of the local maxima. The barycenter plots enhance the tree plot by providing information about the spatial location of the local maxima.

A perspective plot and a contour plot are defined only for two-dimensional functions $f : \mathbf{R}^2 \to \mathbf{R}$, whereas methods in panels (c)–(f) work in any dimension. However, a multivariate function can be visualized with perspective plots and contour plots of the two-dimensional slices and projections of the multivariate function. (Here we mean by slices of $f : \mathbf{R}^d \to \mathbf{R}$ functions of type $(x_1, x_2) \mapsto f(x_1, x_2, a_3, \ldots, a_d)$, where $a_3, \ldots, a_d$ are fixed, and by projections of f functions of type $(x_1, x_2) \mapsto \int_{\mathbf{R}^{d-2}} f(x_1, x_2, x_3, \ldots, x_d) dx_3 \cdots dx_d$. In the case f is a density function the projections are called marginal densities.)

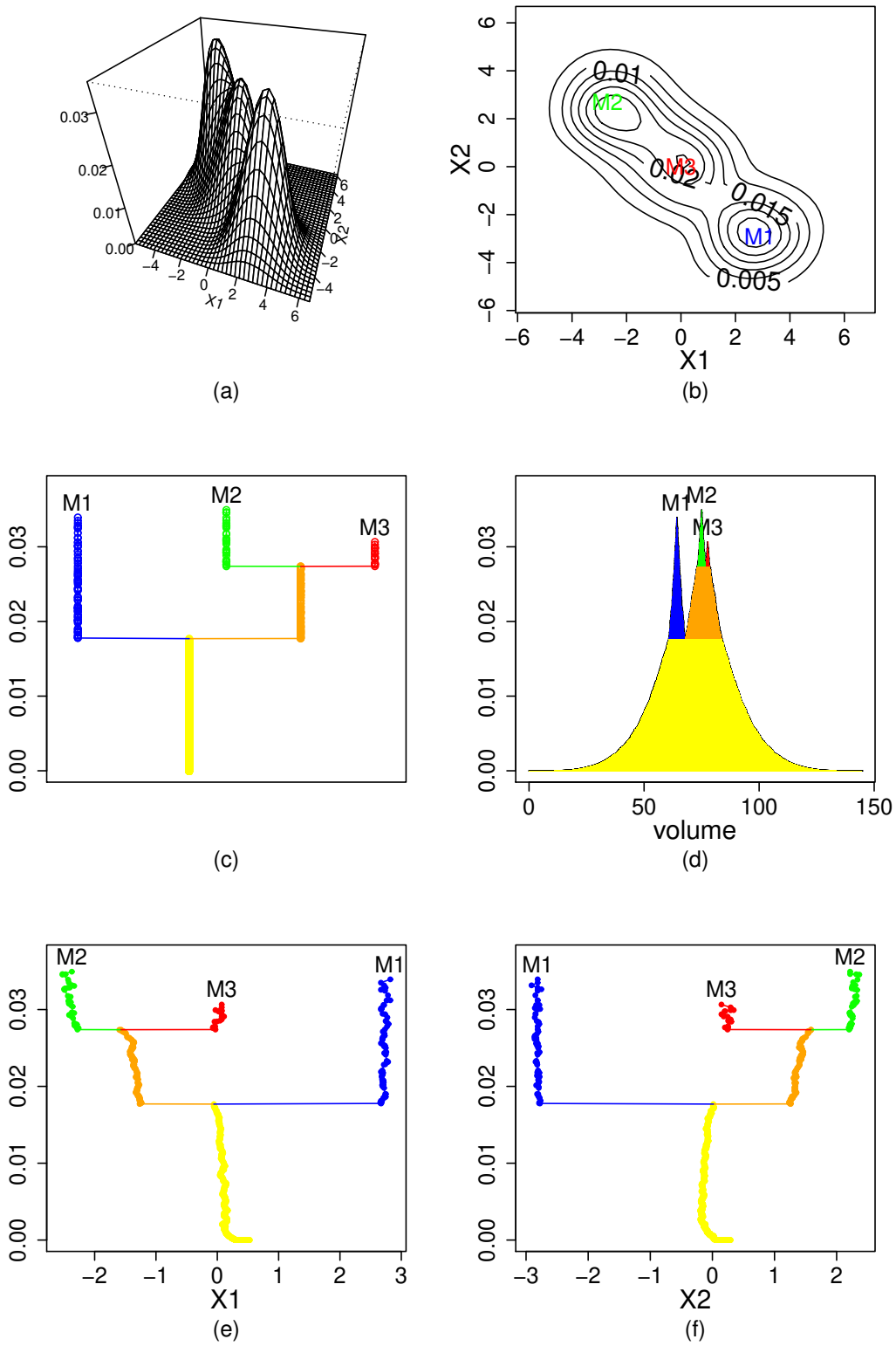

Figure 1: (a) A perspective plot; (b) a contour plot; (c) a tree plot; (d) a plot of a volume function; (e) a barycenter plot of the first coordinate; (f) a barycenter plot of the second coordinate.

A 3D Example of Function Visualization

Figure 2 and Figure 3 show different ways to visualize a 3D function.

Figure 2 visualizes the 3D function using 2D projections and slices. Panel (a) shows a contour plot of the projection $(x_1, x_2) \mapsto \int_{-\infty}^{\infty} f(x_1, x_2, x_3) dx_3$. Panel (b) shows a contour plot of the slice $(x_1, x_2) \mapsto f(x_1, x_2, 0)$. Panel (c) shows a contour plot of the projection $(x_1, x_3) \mapsto \int_{-\infty}^{\infty} f(x_1, x_2, x_3) dx_2$. Panel (d) shows a contour plot of the slice $(x_1, x_3) \mapsto f(x_1, 0, x_3)$. Panel (e) shows a contour plot of the projection $(x_2, x_3) \mapsto \int_{-\infty}^{\infty} f(x_1, x_2, x_3) dx_1$. Panel (f) shows a contour plot of the slice $(x_2, x_3) \mapsto f(0, x_2, x_3)$. We see that the slices do not reveal the local maxima, and more slices would be needed to do that. The projections reveal the local maxima, but some care is needed to identify the local maxima across the three panels.

Figure 3 applies level set trees. Panel (a) shows a tree plot, panel (b) shows a plot of a volume function, panel (c) shows the upper part of the volume function, panel (d) shows a barycenter plot of the first coordinate, panel (e) shows a barycenter plot of the second coordinate, and panel (f) shows a barycenter plot of the third coordinate. The local maxima can be easily identified, and information about the size of the maxima is obtained from the volume function.

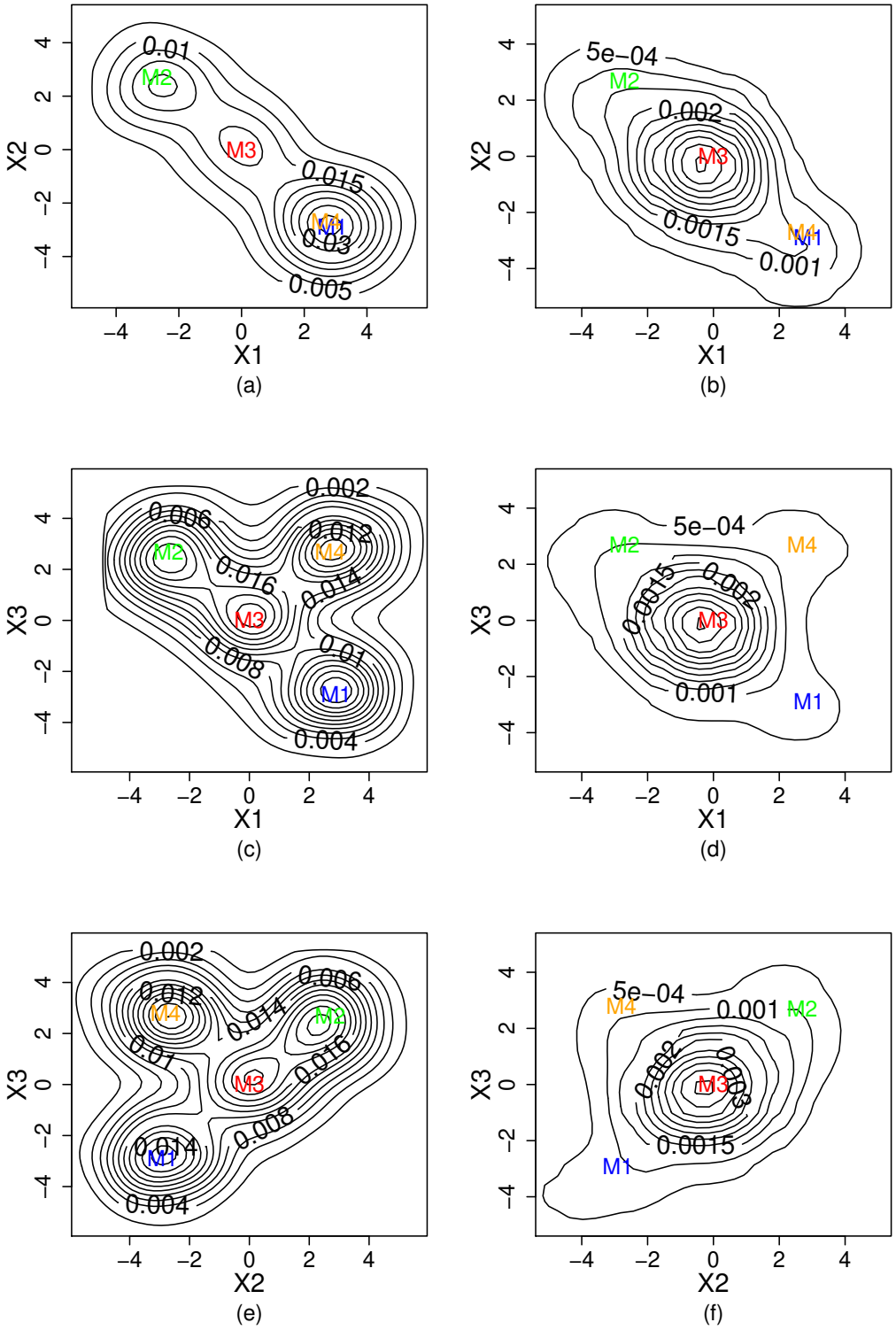

Figure 2: Contour plots of (a) projection $(x_1, x_2) \mapsto \int_{-\infty}^{\infty} f(x_1, x_2, x_3)dx_3$; (b) slice $(x_1, x_2) \mapsto f(x_1, x_2, 0)$; (c) projection $(x_1, x_3) \mapsto \int_{-\infty}^{\infty} f(x_1, x_2, x_3)dx_2$; (d) slice $(x_1, x_3) \mapsto f(x_1, 0, x_3)$; (e) projection $(x_2, x_3) \mapsto \int_{-\infty}^{\infty} f(x_1, x_2, x_3)dx_1$; (f) slice $(x_2, x_3) \mapsto f(0, x_2, x_3)$.

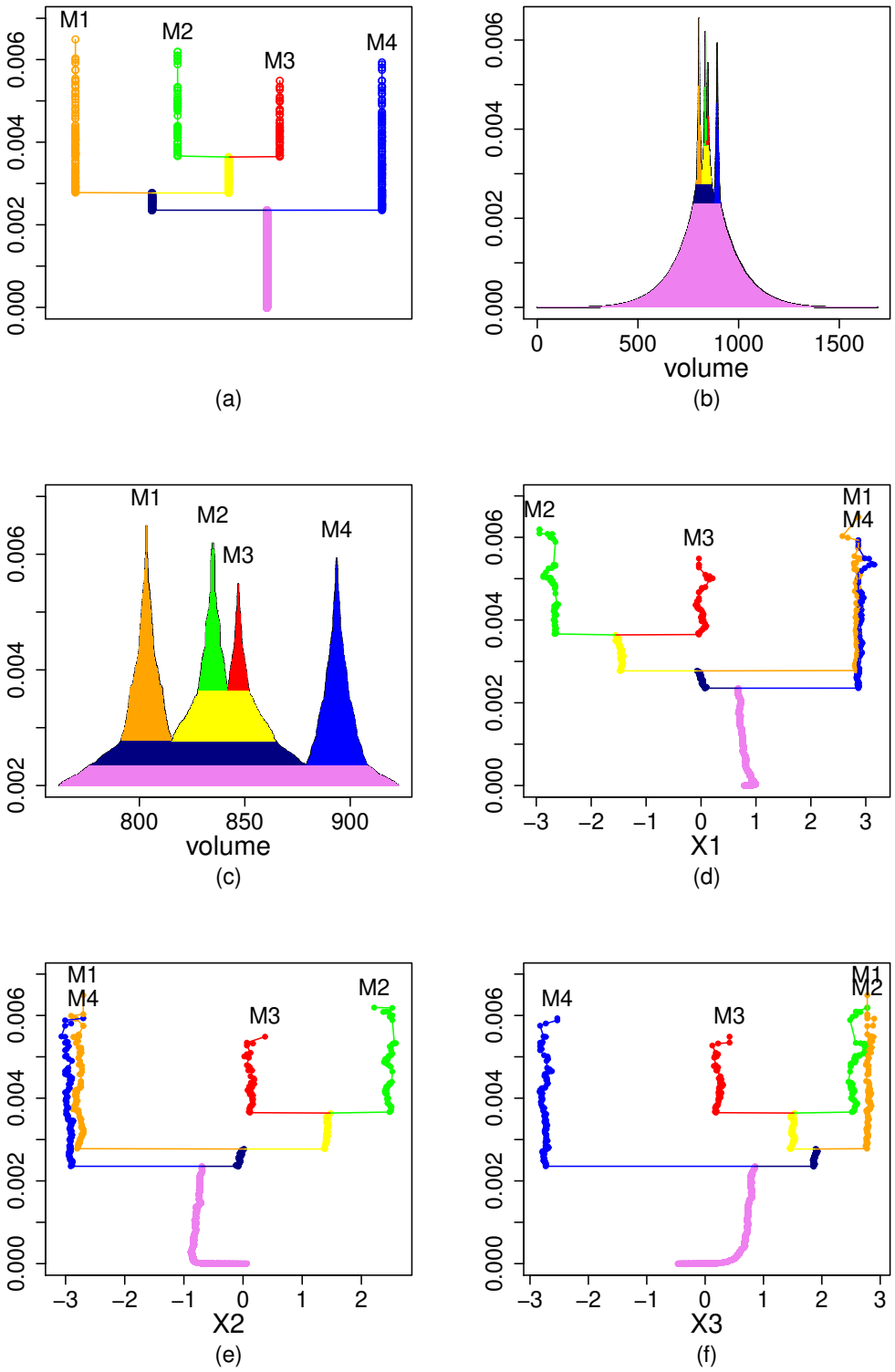

Figure 3: (a) A tree plot; (b) a plot of a volume function; (c) a plot of the upper part of the volume function; (d) a barycenter plot of the 1st coordinate; (e) a barycenter plot of the 2nd coordinate; (f) a barycenter plot of the 3rd coordinate.

Level Set Trees and Density Estimation

Some special issues arise when level set trees are constructed for a probability density function $f : \mathbf{R}^d \to \mathbf{R}$. The density function f of a probability distribution P is defined by the property that for each measurable $A \subset \mathbf{R}^d$ we have $P(A) = \int_A f$. Any function $f : \mathbf{R}^d \to \mathbf{R}$ satisfying $f \geq 0$ and $\int_{\mathbf{R}^d} f = 1$ is a density function of some probability distribution. Not every probability distribution has a density function.

Estimation of a Level Set Tree

The probability density function f is unknown, and the level sets of f have to be estimated using (approximately) independent and identically distributed observations $X_1, \ldots, X_n$, where each $X_i \in \mathbf{R}^d$ is distributed according to density f.

Level set estimators which are useful for the construction of a level set tree can be divided at least into four categories: (1) plug-in-estimators, (2) union of balls and related estimators, (3) union of polyhedrons and related estimators, and (4) estimators defined as optimizers of the empirical excess mass.

A plug-in estimator of a level set takes the estimate to be a level set of a density estimate. Convergence rates of plug-in-estimators, when the density estimator is the kernel estimator, has been studied in many articles; see Baíllo et al. (2000), Baíllo et al. (2001), Cuevas et al. (2000, 2001), Baillo (2003), Cadre (2006), Cuevas et al. (2006), Biau et al. (2007), Burman and Polonik (2009), Mason and Polonik (2009), Rigollet and Vert (2009), Singh et al. (2009), Rinaldo and Wasserman (2010), Mammen and Polonik (2013), Steinwart (2015). Rinaldo and Wasserman (2010) estimate the level sets using the level sets of a kernel estimator and their theory applies to probability distributions that have nonsmooth Lebesgue densities or do not even admit a density. Holmström et al. (2017) estimate and compute a level set tree and a volume function using an adaptive discretization of a kernel density estimator. First, an adaptive partition of the sample space is constructed with the help of a greedy splitting algorithm. The size of the adaptive partition is typically equal to the sample size n, but it can be chosen to have a smaller size. Each member of the partition is a d-dimensional rectangle. Second, a kernel density estimator is evaluated at the

centers of the members of the partition. This leads to a piecewise constant approximation of the kernel estimator. This method of grid construction can be considered as an attempt to find a partition into rectangles that best approximates the Voronoi partition. The Voronoi partition is the collection of the Voronoi cells. For a sample of points $X_1, \ldots, X_n$ the Voronoi cell of X_i is the set of those $x \in \mathbf{R}^d$ which are closer to X_i than to the other points X_j, $j \neq i$.

A union of balls estimator of a level set finds those observations where a density estimator takes a value which is larger of equal to the level λ of the level set. The estimator is the union of the balls whose centers are equal to these observations. Walther (1997) removes certain balls near the boundary to remove bias. The union of balls estimator is equal to the level set of a kernel estimator only when $\lambda = 0$ (so that the support is estimated) and the support of the kernel function is a ball. Devroye and Wise (1980) and Cuevas and Rodriguez-Casal (2004) study asymptotics of a union of balls estimator of the support. There are fast algorithms to compute the corresponding level set tree when the level sets are unions of balls, but the computation of the volume function is difficult, because the computation of the volume of a union of balls poses computational challenges.

A union of polyhedrons estimator has been applied in topological data analysis, when the support A of a uniform probability distribution is estimated using Delaunay partitions (triangulations); see Zomorodian (2012). A sample of independent and identically distributed observations from a uniform distribution whose support is set $A \subset \mathbf{R}^d$ is called point cloud data. The Delaunay triangulation is the collection of simplices with $d+1$ vertices when the vertices are the data points. Note that a Delaunay partition and a Voronoi partition are "dual" to each other. The number of simplices in the Delaunay partition grows exponentially with the dimension of the observations; see McMullen (1970). Thus, Delaunay partitions are computationally expensive, or unfeasible. Azzallini and Torelli (2007) apply Delaunay partitions in level set estimation. Aaron and Bodart (2016) reduce the number of sets in the partition by taking only those simplices which fit inside a small ball, or only those simplices which are such that the lengths of all edges are small.

A level set estimator can be defined as a minimizer of the empirical excess mass. The excess mass criterion was proposed by Hartigan (1987) and Müller and Sawitzki (1991), and studied by Nolan (1991) and Polonik (1995), who derive rates of convergence for support

estimation based on excess mass estimates. Mammen and Tsybakov (1995) study density support problem under a general setting of entropy conditions and their set up includes regions with boundaries that satisfy smoothness conditions (Dudley classes) and convex sets. Tsybakov (1997) extends the results to level set estimation. Klemelä (2004*a*) applies a recursive splitting algorithm to minimize an empirical excess mass in support estimation.

The level set estimators that could be applied to construct a level set tree should be such that they are able to estimate level sets with many connected components, which excludes the use of the convex hull estimator, and the piecewise polynomial estimator of the boundary function of a star shaped level set; see Korostelev and Tsybakov (1993).

Mode Detection

Level set trees can be combined with formal statistical mode testing procedures. In mode testing the purpose has been to test the existence of local maxima of a density function against to the null hypothesis of unimodality. Level set trees lead naturally to mode testing where the existence of several connected components of a level set is tested against the null hypothesis of a single component. The "branching map" is a related graphical tool, introduced by Klemelä (2009, Chapter 7.2).

Density Based Clustering

Density based clustering can be implemented with the help of level set trees. In density based clustering the clusters consist of observations that are inside a connected component of a level set. It is not enough to look at only one level set and its connected components, and that is why level set trees can be helpful, because a level set tree describes the complete tree structure of level sets; see Klemelä (2009).

Density based clustering was proposed by Wishart (1969) and Hartigan (1975), who defined clusters as regions of high density, separated from other such regions by regions of low density. Density based clustering has been reviewed in several articles; see Fraley and Raftery (2002) and Kriegel et al. (2011). Chacón (2015) studies the exact definition of the population goal of the density based clustering, and defines the goal as a collection of high density regions.

Stuetzle (2003) defines a cluster tree as a level set tree whose levels are such that the topology of the level set is changing at those levels. When the node of a level set tree is associated with set $A \subset \mathbf{R}^d$, then the node of a cluster tree is associated with observations satisfying $X_i \in A$. We use below the term "cluster tree of observations" to highlight that the nodes of a cluster tree are associated with subsets of observations. Level set trees of density functions generate high-density sample regions, whereas cluster trees of observations generate high-density clusters of data points.

Figure 4(a) shows a scatter plot of points in $\mathbf{R}^2$, which is generated from a density function $f : \mathbf{R}^2 \to \mathbf{R}$. In fact, the function visualized in Figure 1 is a kernel density estimate of f, constructed using the data in the scatter plot. Panel (b) shows a corresponding colored scatter plot. The observations in the same branch of the level set tree have the same color. The colors are the same as in Figure 1. The colored scatter plot suggests several possible clusterings of the observations.

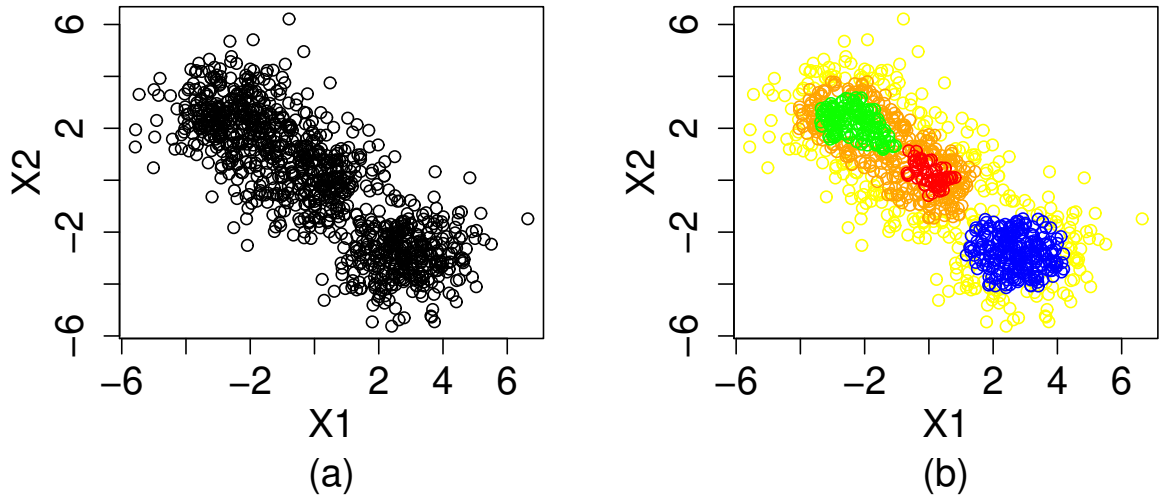

Figure 4: (a) A scatter plot of points, whose density is estimated in figure 1; (b) a colored scatter plot which groups those observations together, which are in the same branch of the level set tree.

Figure 5 shows a typical approach to density based clustering, which uses the connected

components of a level set with level λ to determine the clusters. Panel (a) shows the volume function together with a level λ which leads to two clusters, colored with blue and orange in panel (b). Panel (c) has a higher level λ which leads to three clusters, colored with blue, red, and green in panel (d).

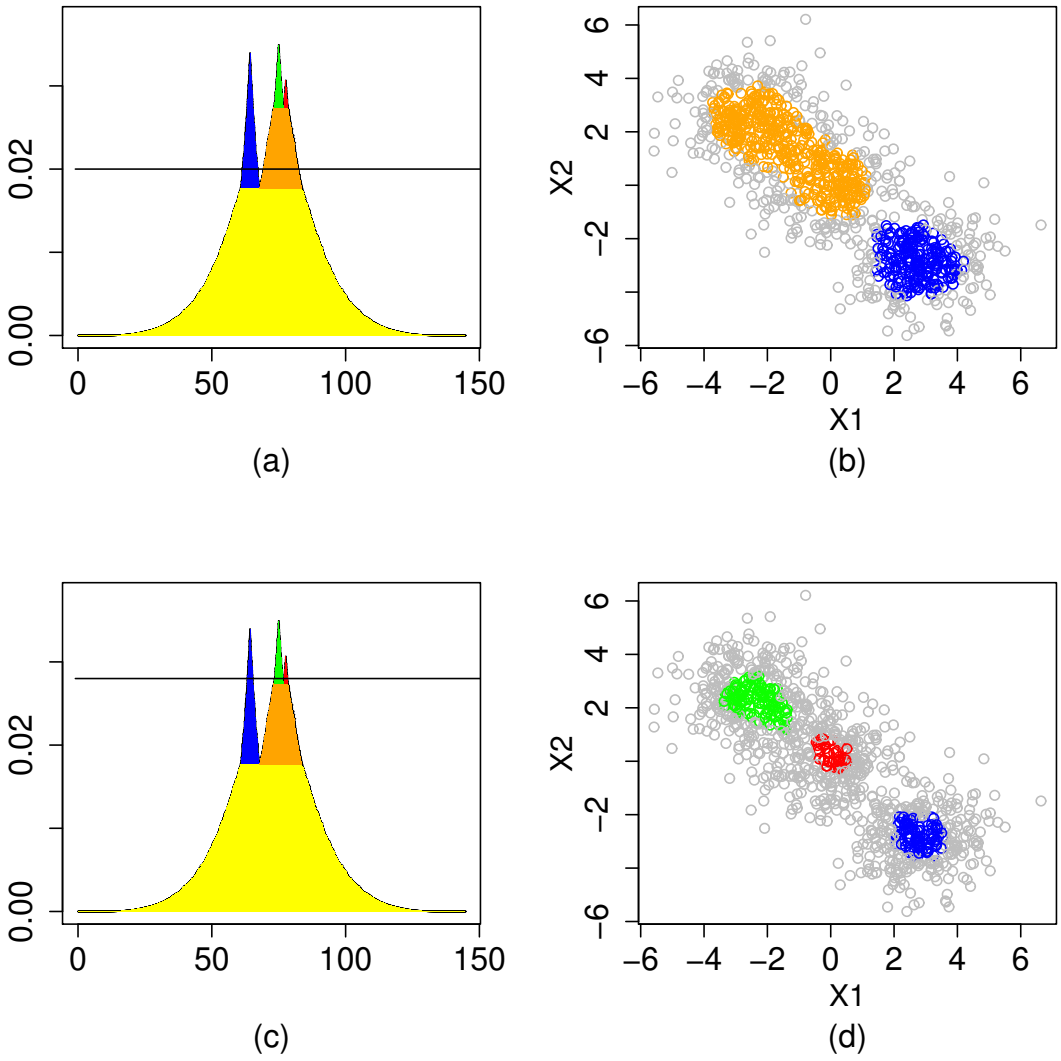

Figure 5: (a) A volume function with the level to determine clusters; (b) a scatter plot with the corresponding two clusters; (c) a volume function with a higher level to determine clusters; (d) a scatter plot with the corresponding three clusters.

Figure 6 shows a different approach to choose clusters than in Figure 5, where a fixed

level λ was used. There is no universal rule that level λ should be constant, but it can be different in different branches of the level set tree. Panel (a) shows two different levels, chosen according to the branching of the level set tree. Panel (b) shows the corresponding three clusters.

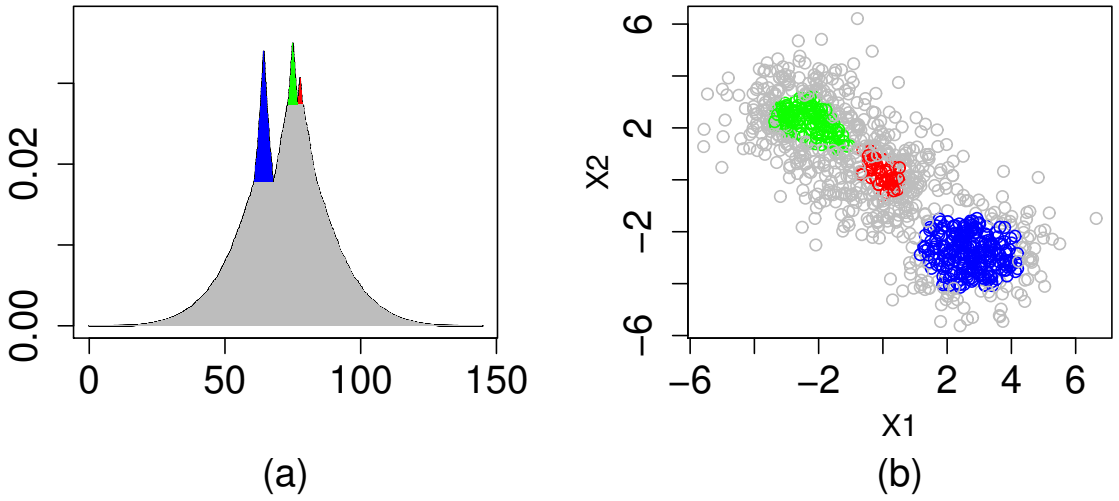

Figure 6: (a) A volume function with two levels to determine clusters; (b) a scatter plot with the corresponding three clusters.

Level Set Trees and Regression Function Estimation

We define regression function $f : \mathbf{R}^d \to \mathbf{R}$ as the conditional expectation $f(x) = E(Y \mid X = x)$, where $Y \in \mathbf{R}$ is the response variable and $X \in \mathbf{R}^d$ is the vector of explanatory variables. The regression function is unknown and has to be estimated using (approximately) independent and identically distributed observations $(X_1, Y_1), \ldots, (X_n, Y_n)$.

The local maxima of the regression function occur for those x-values for which the response variable takes typically large values. The local minima of the regression function occur for those x-values for which the response variable takes typically small values. Thus, we are interested both in the local maxima and in the local minima of a regression function. This is in contrast to the case of a density function, for which we are typically interested

only in the local maxima, which occur in the areas of the concentration of the probability mass.

In regression analysis we are also interested in the partial effects of the explanatory variables. The partial effect of the first explanatory variable x_1 is defined as the partial derivative $D_1 f(x) = \partial/\partial x_1 f(x)$. The partial effect is positive in those areas of x-values for which increasing x_1 typically increases the value of the response variable. The local maxima of the partial derivative $D_1 f$ occur for those x-values for which the partial effect of x_1 is large, assuming that there is a positive partial effect. Thus it is of interest to draw level set trees of $D_1 f$.

Algorithms for the Computation of a Level Set Tree

Computation of a level set tree of a function $f : \mathbf{R}^d \to \mathbf{R}$ requires the evaluation of the function at a finite number of points $x_1, \ldots, x_m \in \mathbf{R}^d$. Algorithms can be classified by their approach to choose the points $x_1, \ldots, x_m$. Often a regular equispaced grid (dense mesh) is chosen, but then the number of points grows exponentially when the dimension d increases.

An early algorithm for the computation of a level set tree used the disjoint-set data structure, which is also known as the union-find data structure; see Carr et al. (2003) and Tarjan (1976). The purpose was to compute a Reeb graph, which is a graph whose leaf vertices represent the local minima or maxima, and each interior vertex represent the joining or splitting of the contours of the function, when a contour of a function is defined as $\{x \in \mathbf{R}^d : f(x) = \lambda\}$, where $\lambda \in \mathbf{R}$; see Reeb (1946). The computation of a Reeb graph can be done by first computing a level set tree and a lower level set tree. A lower level set tree is otherwise similar to a level set tree, but its nodes correspond to the connected components of sublevel sets $\{x \in \mathbf{R}^d : f(x) \leq \lambda\}$. The level set tree and the lower level set tree are pruned so that all the other nodes are removed but the leaf nodes and the nodes with more than one child. (The pruned level set tree is called the split tree and the pruned lower level set tree is called the join tree.) Finally, the pruned trees are merged to obtain the Reeb graph.

Algorithms for Cluster Trees

Algorithms for the computation of a cluster tree of observations can sometimes be applied for the computation of a level set tree. This happens when the level sets are estimated as unions of balls, for example. It is also possible to create a level set tree from a cluster tree of observations by associating each cluster of observations with the union of the Voronoi cells of the observations in the cluster. Other constructions can also be applied. In fact, a cluster of observations can be taken as a point cloud approximating a connected component of a level set. However, it may be difficult to compute the volume function when the level set tree is created from Voronoi cells, due to the complexity of the Voronoi cells.

Typically algorithms for the computation of a cluster tree of observations first find those observations that are estimated to be inside of the level set with level λ. A graph is constructed whose vertices (nodes) are these observations, and edges connect the observations. Then connected components of the graph can be found by traveling the vertices of the graph, using the depth-first search as in Tarjan (1972). There are several methods to determine the edges. The simplest method makes an edge between the vertices for observations X_i and X_j if the distance between X_i and X_j is smaller than some threshold value, or if X_i and X_j are among each others k-nearest neighbors; see Maier et al. (2009), Kpotufe and von Luxburg (2011), Kent et al. (2013). A more complicated version places an edge between X_i and X_j if the amount of probability mass that would be needed to fill the valleys along a line segment between X_i and X_j is smaller than a user-specified threshold. Methods where the edges are weighted are called edge iteration methods and methods where the edges are unweighted are called point iteration methods, using the terminology of Kent et al. (2013).

Stuetzle and Nugent (2010) propose a three step approach for the computation of a cluster tree of observations. First, a density estimate is evaluated at the observations. Second, a graph G is created whose vertices are the observations, and edges connect the observations. The weight of an edge is the minimum value of the density estimate along the line joining the data points. Third, for each level λ a subgraph of G is created. The vertices of the subgraph are those observations where the density estimate takes a value larger or equal to λ. A connection exists between two data points when the weight of the edge is

larger or equal to λ. Finally, the connected components of all subgraphs are computed. Chaudhuri and Dasgupta (2010) apply the k-nearest neighbor density estimator, the graph contains an edge if the Euclidean distance between the two observations is small, and the weights of the edges are deduced from these Euclidean distances. Rinaldo and Wasserman (2010) compute first the ρ-nearest neighborhood graph from the observations that are inside an estimate of the level set. This graph is such that there is an edge between any two nodes if and only if they both belong to a ball of radius ρ. Menardi and Azzalini (2014) compute the cluster tree of observations by first looking at the all pairs of observations, making a graph whose vertices are the observations, and adding an edge between the observations when the kernel estimate does not have a valley along the segment joining the observations. At each level a subgraph is formed from the observations inside the kernel estimate of the level set. (They use the term cluster tree to denote what we call a level set tree.)

The algorithms for the computation of a cluster tree of observations reviewed in the previous paragraph can be applied for the computation of a level set tree when the observations are replaced by regions of the sample space. This kind of approach was followed by Ooi (2002), who uses a histogram estimate. First, a histogram is constructed using a recursive partitioning. The binary tree associated to the histogram is called a density tree. Second, an adjacency graph is constructed. Each vertex of the graph corresponds to a terminal region of the density tree and the vertices for adjacent regions are connected by edges. A weight of an edge can be taken as the arithmetic mean of the values of the empirical probabilities of the rectangles. A clustering tree is a binary tree obtained by merging the nodes of the density tree using the weighting function.

The algorithms described in the previous paragraphs take $O(dn^2)$ steps, when the distances between all pairs of n observations are computed. For example, computation of an Euclidean distance takes d steps. A k-d-algorithm can diminish the number of steps to $O(dn \log n)$, but then the size of memory is of order $n^{O(d)}$; see Indyk (2004). (The k-d-algorithm applies a k-d-tree to return the point in $\mathcal{X}$ closest to x, when the input is a finite set $\mathcal{X}$ of points in $\mathbf{R}^d$ and a query point x.) The algorithm of Stuetzle and Nugent (2010) seems to require $O(C_{n,d}n^2)$ steps, where $C_{n,d}$ can be much larger than d. Indeed, the minimum value of a density estimate on the all line segments joining the observations needs to

be found. For example, the evaluation of a kernel density estimate at one point takes $O(nd)$ steps and the evaluation at many points is needed. After the first proximity graph is computed, the traveling of the graphs and the decomposition of the graphs into the connected components takes $O(n)$ steps, but this has to be done for each level set separately. Thus, the previous algorithms seem to be designed for small sample sizes, although the dimension of the data can be high.

Further Algorithms

Algorithms for the computation of level set trees can be divided to those which compute the level set tree starting from the roots (roots first algorithms) and to those which compute the level set tree starting from the leafs (leafs first algorithms).

Klemelä (2006) introduced the Leafsfirst algorithm to compute a level set tree. Leafsfirst algorithm can be used when the level sets of $f : \mathbf{R}^d \to \mathbf{R}$ for levels $\lambda_1 < \cdots < \lambda_L$ can be written as

$$\Lambda(f, \lambda_l) = \bigcup_{j=l}^{L} A_j, \tag{1}$$

for $l = 1, \ldots, L$, where $A_1, \ldots, A_L \subset \mathbf{R}^d$ is a collection of sets-. Condition (1) says that there exists a collection $A_1, \ldots, A_L$ of "elementary sets" or "atoms" which are such that all the level sets are unions of these atoms. The lowest level set is a union of all atoms and higher level sets are unions of subcollections of the atoms. For example, condition (1) holds when f is piecewise constant as

$$f(x) = \sum_{i=1}^{L} \lambda_i I_{A_i}(x).$$

Unlike the previously described algorithms, Leafsfirst algorithm does not compute a proximity graph. Instead, the algorithm starts at atom $A \in \{A_1, \ldots, A_L\}$ with the highest function value. Atom A is the first leaf node. Atom A is merged with the atoms A_i with the next highest function values if those atoms are connected to the previous atoms. Otherwise, new leaf nodes are created. The worst case complexity is $O(dL^2)$. With bounding boxes it is possible to avoid making all pairwise comparisons between the atoms (to find which atoms touch each other), because the comparison is needed only when the next atom is connected to

a bounding box of connected atoms which was already found. Thus, running the algorithm takes in typical cases a smaller number of steps than $O(dL^2)$.

Other algorithms are available when function f satisfies further conditions. Let us assume that f is piecewise constant, and the pieces are constructed using a recursive splitting of the support. Then f can be represented with a binary tree. A level set of f can be divided into connected components with a dynamic programming algorithm, which joins the connected sets by going through the binary tree from the leaves to the root; see Klemelä (2005).

Further Uses

Visualizing Sets

A set $A \subset \mathbf{R}^d$ can be visualized with the help of a boundary function as in Klemelä (2006).

Let us assume that A is star shaped. Set A is said to be star-shaped when there exist a center $\mu \in A$ from which all the remaining points of A are visible, which means that the segment joining μ and x is included in A for all $x \in A$. The boundary function of a bounded star-shaped set A from the center μ, is a real function $b : \mathbf{S}_{d-1} \to [0, \infty)$, where $\mathbf{S}_{d-1}$ denotes the unit sphere on $\mathbf{R}^d$ and $b(\eta)$ is defined as the distance from μ to the closest boundary point of A.

More formally, set A is star-shaped, when there is a center point $\mu \in \mathbf{R}^d$ and a boundary function

$$b : \mathbf{S}_{d-1} \to [0, \infty) \tag{2}$$

so that the A can be written as

$$A = \big\{\mu + r\eta \in \mathbf{R}^d : \eta \in \mathbf{S}_{d-1}, 0 \le r \le b(\eta)\big\},$$

where $\mathbf{S}_{d-1} = \{x \in \mathbf{R}^d : \|x\| = 1\}$ is the unit sphere.

Now we can study and visualize the boundary function b using level set trees. For example, level set trees are not very helpful for directly analyzing a unimodal density function $f : \mathbf{R}^d \to \mathbf{R}$. However, it can be useful the study the shapes of the level sets of f, or the shapes of the level sets of the density of the copula of f. The copula means the distribution function that has been obtained by normalizing the marginals to have a specified fixed

distribution function, which is often chosen to be the uniform distribution. Thus, the copula density can be used to conveniently study the dependence between the components of a random vector, because the effect of the marginals has been eliminated.

Term level set methods refers sometimes to numerical analysis of surfaces and shapes, where a closed curve Γ is analyzed by finding a function f such that $\Gamma = \{x : f(x) = 0\}$; see Sethian (1996) and Osher and Fedkiw (2002).

Optimization of Functions

Let us consider maximization of function $f : A \to \mathbf{R}$, where $A \subset \mathbf{R}^d$. Creating a level set tree of f is one way to solve the optimization problem of finding all local maxima of function f.

A large body of optimization literature considers the case of convex optimization. In convex optimization function f is convex (for minimization) or concave (for maximization), and the domain is a convex set. Particular cases of convex optimization are the optimization of linear or quadratic functions under linear or quadratic constraints. Since convex (concave) functions do not have many local minima (maxima), the optimization of these functions using level set tree methods has a limited interest.

A local maximum of a continuous function can be found by iterative algorithms. Many iterative algorithms are modifications of Newton's algorithm. In Newton's algorithm we choose a starting point $x_0 \in A$ and define a sequence of vectors by

$$x_{n+1} = x_n - [Hf(x_n)]^{-1} \nabla f(x_n),$$

where $n \geq 0$, $Hf(x_n)$ is the Hessian matrix ($d \times d$ matrix of the second partial derivatives) and $\nabla f(x_n)$ is the gradient (d vector of the first partial derivatives). Sequence $\{x_n\}$ converges to a point x_* such that $\nabla f(x_*) = 0$, so that x_* can be a local maximum, local minimum, or saddle point. We can find all the local maxima by choosing a large number of different starting points. Iterative algorithms can be used to find out the collection of local extremes and the saddle points but they do not give any other information about the shape of the function; see Nocedal and Wright (1999).

Gorban (2013) has used level set tree methods to study dynamical systems with a strictly

convex Lyapunov function f defined on a positively invariant convex polyhedron. Level set trees can help to find the admissible paths, along which f decreases monotonically, and find the states that are attainable from the given initial state along the admissible paths.

Functions Whose Domain is a Metric Space

Level set trees can be defined for functions $f : \mathbf{M} \to \mathbf{R}$, where $\mathbf{M}$ is a metric space. The most applications are for the Euclidean space $\mathbf{M} = \mathbf{R}^d$ with the Euclidean distance. An other interesting case is the unit sphere $\mathbf{M} = \mathbf{S}_{d-1}$, where $\mathbf{S}_{d-1} = \{x \in \mathbf{R}^d : \|x\| = 1\}$, accompanied with the geodesic distance (Riemannian distance). Note that we defined in (2) a boundary function of a star shaped set as a function with the domain $\mathbf{S}_{d-1}$. Already in the case $d = 3$ it is useful to draw a level set tree, because a perspective plot of a function $f : \mathbf{S}_2 \to \mathbf{R}$ is not easy to draw and interpret.

Further Applications

We discuss below the applications of level set trees to flow cytometry data and to Bayesian data analysis. Further applications of level sets have been found in engineering (anomaly detection) Desforges et al. (1998), in medical imaging Willett and Nowak (2005), in astronomy Jang (2006), and in econometrics (partially identified models when the estimated set is a subset of the parameter space) Chernozhukov et al. (2007); Bugni (2010).

Flow Cytometry Data

A flow cytometer is a laser instrument, which measures chemical and biophysical characteristics of cellular particles. The number of analyzed cells may be much larger than 10^5 in one sample. Up to 20 measurements are made from a single cell. Cancer diagnostics is one important use of flow cytometry.

Level set tree methods combined with nonparametric density estimation provide promising tools for analyzing and clustering flow cytometry data Karttunen et al. (2014); Holmström et al. (2017). In particular, clustering of cells can indicate the stage of a disease. Clustering has traditionally been manual, and it has been based on one- and two-dimensional marginal plots of

the high-dimensional observations. Sometimes mixture models and k-means clustering have been applied. Mixture models and k-means clustering lead to convex clusters, whereas clustering with nonparametric density estimation can find nonconvex clusters. Nonparametric density estimation has been applied to cluster flow cytometry data in Walther et al. (2009), Naumann et al. (2010), Duong et al. (2008), Duong et al. (2009).

Bayesian Data Analysis

Zhou and Wong (2008) apply level set tree techniques to make Bayesian inference about a posterior distribution related to DNA sequence segmentation. They generate Monte Carlo samples from the posterior distribution and use a clustering algorithm to find clusters which are considered as estimates of level sets. The level sets are not the level sets of the posterior density f but of $h(x) = -\log f(x)$, whose local minima are of interest. They use a bottom-up partition algorithm, which belongs in our terminology to the leafs first type.

CONCLUSIONS

Level set tree methods have so far found most applications in cluster analysis, but we believe that they have a large number of potential applications in function visualization, optimization, and regression function estimation, for example. Level set tree methods provide tools for analyzing interactions and dependencies, which involve simultaneously more than one or two variables of a multivariate function.

FURTHER READING

Fomenko and Kunii (1997) make a link "between the theoretical aspects of modern geometry and topology, on the one hand, and experimental computer geometry, on the other". Fomenko and Kunii (1997, Definition 8.1.2) define a Reeb graph of a function $f : \mathbf{M} \to \mathbf{R}$, where $\mathbf{M}$ is a compact manifold, as the space of connected components of the contours (isosurfaces) of the function, and they refer to Reeb (1946) as the origin of the concept. Applications of Reeb graphs in computer geometry were described by Kunii and Shinagawa (1992).

Milnor (1963), Guillemin and Pollack (1974), and Matsumoto (2000) provide a mathematical discussion of differential topology.

Related research has been done in topological data analysis, where the purpose has often been to reconstruct topological properties of an object using point cloud data, which means that the topological properties of the support of a probability distribution are estimated when a sample of i.i.d. observations is available. Edelsbrunner et al. (2000) introduce a persistence diagram (barcode) to represent topological information about functions, and Singh et al. (2007) visualize data by simplicial complexes, where volume information is included as the largeness of the nodes of the complexes. Carlsson (2009) gives a review of concepts of topological data analysis.

In the context of scientific visualization Weber et al. (2007) use volume information to make a 2D configuration of contour trees.

Klemelä (2009) discusses level set trees in Chapter 4, visualization of sets using level set trees in Chapter 5, visualization of data clouds using level set trees in Chapter 6, visualizing scales of density estimated using level set trees in Chapter 7, using level set trees in cluster analysis in Chapter 8.3, and algorithms for computing level set trees in Chapter 13.

References

Aaron, C. and Bodart, O. (2016), 'Local convex hull support and boundary estimation', *J. Multivariate Anal.* **147**, 82–101.

Azzallini, A. and Torelli, N. (2007), 'Clustering via nonparametric density estimation', *Statistics and Computing* **17**, 71–80.

Baillo, A. (2003), 'Total error in a plug-in estimator of level sets', *Stat. Probab. Lett.* **65**, 411–417.

Baíllo, A., Cuesta-Albertos, J. A. and Cuevas, A. (2001), 'Convergence rates in nonparametric estimation of level sets', *Statist. Probab. Lett.* **53**, 27–35.

Baíllo, A., Cuevas, A. and Justel, A. (2000), 'Set estimation and nonparametric detection', *Canadian J. Statist.* **28**, 765–782.

Biau, G., Cadre, B. and Pelletier, B. (2007), 'A graph-based estimator of the number of clusters', *ESAIM Probab. Stat.* **11**, 272–280.

Bugni, F. (2010), 'Bootstrap inference in partially identified models defined by moment inequalities: Coverage of the identified set', *Econometrica* **76**, 735–753.

Burman, P. and Polonik, W. (2009), 'Multivariate mode hunting: Data analytic tools with measures of significance.', *J. Multivariate Anal.* **100**, 1198–1218.

Cadre, B. (2006), 'Kernel estimation of density level sets', *J. Multivariate Anal.* **97**(4), 999–1023.

Carlsson, G. (2009), 'Topology and data', *Bulletin Am. Math. Soc.* **46**(2), 255–308.

Carr, H., Snoeyink, J. and Axen, U. (2003), 'Computing contour trees in any dimension', *Comput. Geometry: Theory Appl.* **24**(2), 75–94.

Chacón, J. E. (2015), 'A population background for nonparametric density-based clustering', *Statist. Sci.* **30**(4), 518–532.

Chaudhuri, K. and Dasgupta, S. (2010), Rates of convergence for the cluster tree, *in* J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel and A. Culotta, eds, 'Advances in Neural Information Processing Systems 23', Curran Associates, Vancouver, BC, pp. 343–351.

Chernozhukov, V., Hong, H. and Tamer, E. (2007), 'Estimation and confidence regions for parameter sets in econometric models', *Econometrica* **75**, 1243–1284.

Cuevas, A., Febreiro, M. and Fraiman, R. (2000), 'Estimating the number of clusters', *Canad. J. Statist.* **28**, 367–382.

Cuevas, A., Febreiro, M. and Fraiman, R. (2001), 'Cluster analysis: A further approach based on density estimation', *Comput. Stat. Data Anal.* **36**, 441–459.

Cuevas, A., González-Manteiga, W. and Rodriguez-Casal, A. (2006), 'Plug-in estimation of general level sets', *Aust. N. Z. J. Stat.* **48**, 7–19.

Cuevas, A. and Rodriguez-Casal, A. (2004), 'On boundary estimation', *Adv. Appl. Probab.* **36**(2), 340–354.

Desforges, M. J., Jacob, P. J. and Cooper, J. E. (1998), 'Application of probability density estimation to the detection of abnormal conditions in engineering', *Proc. Institute Mechanical Engineering* **212**, 687–703.

Devroye, L. and Wise, G. L. (1980), 'Detection of abnormal behavior via nonparametric estimation of the support', *SIAM J. Appl. Math.* **38**, 480–488.

Duong, T., Cowling, A., Koch, I. and Wand, M. P. (2008), 'Feature significance for multivariate kernel density estimation', *Comput. Statist. Data Anal.* **52**(9), 4225–4242.

Duong, T., Koch, I. and Wand, M. P. (2009), 'Highest density difference region estimation with application to flow cytometry data', *Biometrical Journal* **51**, 504–521.

Edelsbrunner, H., Letscher, D. and Zomorodian, A. (2000), Topological persistence and simplification, *in* 'Proc. 41st Ann. IEEE Sympos. Found Comput. Sci.', IEEE Computer Society, Washington, DC, pp. 454–463.

Fomenko, A. T. and Kunii, T. L., eds (1997), *Topological Modeling for Visualization*, Springer, Berlin.

Fraley, C. and Raftery, A. E. (2002), 'Model-based clustering, discriminant analysis and density estimation', *J. Am. Statist. Assoc.* **97**, 611–631.

Gorban, A. N. (2013), 'Thermodynamic tree: The space of admissible paths', *Siam J. Applied Dynamical Systems* **12**(1), 246–278.

Guillemin, V. and Pollack, A. (1974), *Differential Topology*, Prentice-Hall, Englewood Cliffs, NJ.

Hartigan, J. A. (1975), *Clustering Algorithms*, Wiley, New York.

Hartigan, J. A. (1987), 'Estimation of a convex density cluster in two dimensions', *J. Am. Statist. Assoc.* **82**, 267–270.

Holmström, L., Karttunen, K. and Klemelä, J. (2017), 'Estimation of level set trees using adaptive partitions', *Comput. Statist.* **32**, 1139–1163.

Indyk, P. (2004), Nearest neighbors in high-dimensional spaces, *in* J. E. Goodman and J. O'Rourke, eds, 'Handbook of Discrete and Computational Geometry', Chapman & Hall/CRC, Boca Raton, FL, pp. 877–892.

Jang, W. (2006), 'Nonparametric density estimation and clustering in astronomical sky survey', *Comp. Statist. Data Anal.* **50**, 760–774.

Karttunen, K., Holmström, L. and Klemelä, J. (2014), Level set trees with enhanced marginal density visualization: Application to flow cytometry, *in* 'Proceedings 5th International Conference on Information Visualization Theory and Applications', pp. 210–217.

Kent, B. P., Rinaldo, A. and Timothy, V. (2013), DeBaCl: A Python package for interactive DEnsity-BAsed CLustering, Technical report. arXiv:1307.8136.

Klemelä, J. (2004*a*), 'Complexity penalized support estimation', *J. Multivariate Anal.* **88**, 274–297.

Klemelä, J. (2004*b*), 'Visualization of multivariate density estimates with level set trees', *J. Comput. Graph. Statist.* **13**(3), 599–620.

Klemelä, J. (2005), 'Algorithms for the manipulation of level sets of nonparametric density estimates', *Comput. Statist.* **20**, 349–368.

Klemelä, J. (2006), 'Visualization of multivariate density estimates with shape trees', *J. Comput. Graph. Statist.* **15**(2), 372–397.

Klemelä, J. (2009), *Smoothing of Multivariate Data: Density Estimation and Visualization*, Wiley, New York.

Korostelev, A. P. and Tsybakov, A. B. (1993), *Minimax Theory of Image Reconstruction*, Vol. 82 of *Lecture Notes in Statistics*, Springer, Berlin.

Kpotufe, S. and von Luxburg, U. (2011), Pruning nearest neighbor cluster trees, *in* 'Proceedings of the 28th International Conference on Machine Learning', Vol. 105, pp. 225–232.

Kriegel, H.-P., Kröger, P., Sander, J. and Zimek, A. (2011), 'Density-based clustering', *Wires Data Mining Knowledge Discovery* **1**, 231–240.

Kunii, T. L. and Shinagawa, Y., eds (1992), *Modern Geometric Computing for Visualization*, Springer, Berlin.

Maier, M., Hein, M. and von Luxburg, U. (2009), 'Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters', *Theoretical Computer Science* **410**(19), 1749–1764.

Mammen, E. and Polonik, W. (2013), 'Confidence regions for level sets', *J. Multivariate Anal.* **122**, 202–214.

Mammen, E. and Tsybakov, A. B. (1995), 'Asymptotical minimax recovery of sets with smooth boundaries', *Ann. Statist.* **23**, 502–524.

Mason, D. and Polonik, W. (2009), 'Asymptotic normality of plug-in level set estimates', *Ann. Appl. Probab.* **19**, 1108–1142.

Matsumoto, Y. (2000), *An Introduction to Morse Theory*, Vol. 208 of *Translations of Mathematical Monographs*, AMS, Providence, RI. Originally published 1997 in Japanese.

McMullen, P. (1970), 'The maximum numbers of faces of a convex polytope', *Mathematika* **17**, 179–184.

Menardi, G. and Azzalini, A. (2014), 'An advacement in clustering via nonparametric density estimation', *Stat. Comput.* **24**, 753–767.

Milnor, J. (1963), *Morse Theory*, Princeton University Press, Princeton, NJ.

Müller, D. W. and Sawitzki, G. (1991), 'Excess mass estimates and tests of multimodality', *J. Am. Statist. Assoc.* **86**, 738–746.

Naumann, U., Luta, G. and Wand, M. P. (2010), 'The curvHDR method for gating flow cytometry samples', *BMC Bioinformatics* **11**(44).

Nocedal, J. and Wright, S. J. (1999), *Numerical Optimization*, Springer, Berlin.

Nolan, D. (1991), 'The excess-mass ellipsoid', *J. Multivariate Anal.* **39**, 348–371.

Ooi, H. (2002), 'Density visualization and mode hunting using trees', *J. Comput. Graph. Statist.* **11**, 328–347.

Osher, S. J. and Fedkiw, R. P. (2002), *Level Set Methods and Dynamic Implicit Surfaces*, Springer, Berlin.

Polonik, W. (1995), 'Measuring mass concentration and estimating density contour clusters - an excess mass approach', *Ann. Statist.* **23**, 855–881.

Reeb, G. (1946), 'Sur les points singuliers d'une forme de pfaff completement integrable ou d'une fonction numerique', *Comptes Rend. Acad. Sci. Paris* **222**, 847–849.

Rigollet, P. and Vert, R. (2009), 'Optimal rates for plug-in estimators of density level sets', *Bernoulli* **15**, 1154–1178.

Rinaldo, A. and Wasserman, L. (2010), 'Generalized density clustering', *Ann. Statist.* **38**, 2678–2722.

Sethian, J. A. (1996), *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University Press, Cambridge, UK.

Singh, A., Scott, C. and Nowak, R. (2009), 'Adaptive Hausdorff estimation of density level sets', *Ann. Statist.* **37**, 2760–2782.

Singh, G., Memoli, F. and Carlsson, G. (2007), Topological methods for the analysis of high dimensional data sets and 3D object recognition, *in* M. Botsch, R. Pajarola, B. Chen and M. Zwicker, eds, 'Symp. Point Based Graphics', Eurographics Association, Prague, pp. 91–100.

Steinwart, I. (2015), 'Fully adaptive density-based clustering', *Ann. Statist.* **43**, 2132–2167.

Stuetzle, W. (2003), 'Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample', *J. Classification* **20**(5), 25–47.

Stuetzle, W. and Nugent, R. (2010), 'A generalized single linkage method for estimating the cluster tree of a density', *J. Comput. Graph. Statist.* **19**, 397–418.

Tarjan, R. E. (1972), 'Depth-first search and linear graph algorithms', *SIAM J. Computing* **1**(2), 146–160.

Tarjan, R. E. (1976), 'Efficiency of a good but not linear set union algorithm', *J. ACM* **22**, 215–225.

Tsybakov, A. B. (1997), 'On nonparametric estimation of density level sets', *Ann. Statist.* **25**, 948–969.

Walther, G. (1997), 'Granulometric smoothing', *Ann. Statist.* **25**, 2273–2299.

Walther, G., Zimmerman, N., Moore, W., Parks, D., Meehan, S., Belitskaya, I., Pan, J. and Herzenberg, L. (2009), 'Automatic clustering of flow cytometry data with density-based merging', *Advances in Bioinformatics* **2009**, 686–759.

Weber, G., Bremer, P.-T. and Pascucci, V. (2007), 'Topological landscapes: A terrain metaphor for scientific data', *IEEE Trans. Visualization Computer Graphics* **13**(6), 1416–1423.

Willett, R. M. and Nowak, R. D. (2005), 'Level set estimation in medical imaging', *Proc. IEEE Statistical Signal Processing* **5**, 1089–1092.

Wishart, D. (1969), Mode analysis: A generalization of nearest neighbor which reduces chaining effects, *in* A. J. Cole, ed., 'Numerical Taxonomy', Academic Press, New York, pp. 282–311.

Zhou, Q. and Wong, W. H. (2008), 'Bayesian inference of DNA sequence segmentation', *Ann. Appl. Statist.* **2**(4), 1307–1331.

Zomorodian, A. (2012), Topological data analysis, *in* A. Zomorodian, ed., 'Advances in Applied and Computational Topology', Vol. 70, American Mathematical Society, Providence, pp. 1–40.