Carnegie Mellon University

Level Set Trees for Applied Statistics

A Dissertation Submitted to the Graduate School in Partial Fulfillment of the Requirements

for the degree

Doctor of Philosophy

in

Statistics

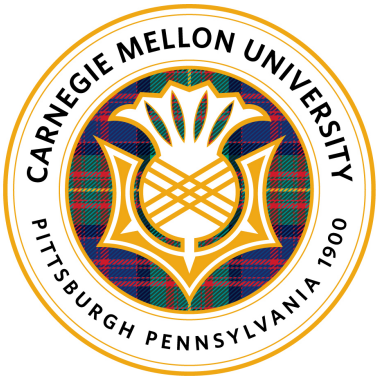
by

Brian Patrick Kent

Department of Statistics

Carnegie Mellon University

Pittsburgh, PA 15213



December, 2013

Abstract

The level set tree concisely describes the hierarchy of modes in a probability density function and is a statistically principled tool for clustering and topographical data analysis. To make level set trees more useful for practical data analysis, we generalize the method to non-Euclidean data and develop a suite of tools for interactive tree visualization, data exploration, and clustering. We also propose several descriptive and quantitative techniques for comparing level set trees, enabling inference about differences between underlying populations and evaluation of uncertainty in tree estimation.

We tested the proposed methods in a range of simulations and real applications. Level set tree clustering performed as well as several popular off-the-shelf methods in simulation experiments, and successfully captured multi-scale structure in population genetic data with over 11,500 covariates and infinite-dimensional neuroimaging data with 100,000 observations.

All of the methods discussed in this thesis are implemented in the publicly available Python package *DeBaCl*. The software emphasizes speed, efficiency, and easy customization; in combination with our methodological extensions, the toolbox transforms the level set tree from an elegant theoretical construct into a powerful and widely applicable data analysis technique.

Acknowledgments

The success of this thesis is due in large part to the vision and guidance of my advisers Alessandro Rinaldo and Timothy Verstynen. Applying sophisticated statistical methods to complex neuroimaging problems requires mastery of each individual domain as well as a willingness to find common ground on terminology, methods, and goals. It is a noble but challenging pursuit, and the wisdom and patience of Ale and Tim helped me to navigate the course with the least possible difficulty. I would also like the thank the rest of my committee—Chris Genovese, Rob Kass, and Aarti Singh—and Larry Wasserman and Fabrizio Lecci for helpful discussions on the statistics side, and Kevin Jarbo for analysis of neuroimaging results.

I am very grateful to my colleagues in the DAWG—Stefa Etchegaray, Zachary Kurtz, and Sonia Todorova—who were an invaluable resource for feedback on various chapters and papers, and who kept me focused over the years on the big picture. I am also thankful for the opportunities given to me by Bernie Devlin, Bill Eddy, and Kathryn Roeder, which helped me to find my home in applied and computational statistics.

Finally, my entire graduate school experiment would never have happened if my parents Ray and Mary had not shown me the statistics path and kept me on it when the going got tough. Last but not least, I'd like to thank my brother Eric for reminding me not to destroy the human race with sentient robots.

Related publications

Much of the text in this thesis and many of the figures and tables appear in the following publications:

- Brian P. Kent, Alessandro Rinaldo, Fang-Cheng Yeh, and Timothy Verstynen. Mapping Topographic Structure in White Matter Pathways with Level Set Trees. *arXiv:1311.5312 (submitted to PLoS ONE)*, 2013.
- Brian P. Kent, Alessandro Rinaldo, and Timothy Verstynen. DeBaCl: A Python Package for Interactive DEnsity-BAsed CLustering. *arXiv:1307.8136 (submitted to the Journal of Statistical Software)*, 2013.
- Brian P. Kent, Alessandro Rinaldo, Timothy Verstynen, and Kevin Jarbo. Branching Out with Level Set Trees: Extensions for Applied Statistics. *in preparation, for submission to The Annals of Applied Statistics*, 2013.

Contents

List of Tables

List of Figures

Notation and acronyms

Symbol	Item
$\mathbb{R}^d$	d-dimensional Euclidean space
x	point in Euclidean space
P	probability distribution
pdf	probability density function
f	a specific probability density function
$\widehat{f}$	density or pseudo-density estimator
f_h	mollified density function of bandwidth h
$\mathbb{E}$	expected value
$\mathbb{X}_n$	sample of size n
x_i	Euclidean sample point
$x^{(i)}$	modal data point in cluster C_i
$\|.\|$	Euclidean norm
kNN	k-nearest neighbor (density estimator or similarity graph)
v_d	volume of d-dimensional ball
$r_k(x)$	radius of the ball centered at x containing $k+1$ data points
K	number of clusters
$\mathcal{O}()$	order of a function (probabilistic by context)
ρ	degree of ease for clustering simulations
X, Y, Z	sampled fiber streamlines
m	number of points on a fiber streamline
D_{mam}	max-average-min distance
d_{cham}	one-directional chamfer distance
D_{mdf}	minimum average direct-flip distance
ARI	adjusted Rand index
AMI	adjusted mutual information index
SNP	single nucleotide polymorphism
HGDP	Human Genome Diversity Project
DWI	diffusion weighted imaging
ROI	region of interest

Symbol	Item
λ	density level
α	mass level
κ	branch mass index
$L_f(\lambda)$	λ-upper level set of f
$\mathcal{T}$	level set tree
$\widehat{\mathcal{T}}$	level set tree estimator
G	similarity graph
e	graph edge
q	number of edges in a graph
G_λ	similarity subgraph induced by $L_f(\lambda)$
γ	pruning parameter
C	high-density cluster
$\mathcal{E}$	set of all level set tree clusters with the same death level λ''.
β	Chaudhuri-Dasgupta robustness parameter
$\mathcal{R}$	cluster tree
`kid`	node child
`sib`	node sibling
`parent`	node parent
M	cluster mass
$\widehat{M}_i$	fraction of data in cluster C_i
$\kappa'' - \kappa'$	littoral mass of cluster tree node i
κ', α', λ'	κ, α, λ birth levels
κ'', α'', λ''	κ, α, λ death levels
background	points left unlabeled by a level set tree clustering method
D_{emd}	earth mover distance
D_{pmd}	paint mover distance
D_α	α-earth mover distance
J	number of values of α used to compute D_α
$\mathcal{S}$	dendrogram signature
z, y	horizontal and vertical plot canvas coordinates
$[a, b]$	silo of whitespace around a dendrogram branch
Υ	orchard, a collection of level set trees
υ	number of resampled data sets

Chapter 1

Introduction

The level set tree of a probability density function (pdf) concisely represents the hierarchy of the function's modes. This simple idea was proposed at least as early as 1975 as a statistically principled way to define the slippery concept of a cluster (Hartigan, 1975), and offers numerous theoretical and methodological advantages as a clustering solution. Despite the elegance of the approach, however, it remains underutilized as a tool for practical data analysis. This thesis and the accompanying software are intended to promote the use of level set trees in applied statistics by showing the effectiveness of level set tree clustering and by demonstrating that the level set tree provides a global, multi-scale, and easily visualized summary of data topography that is more informative than any single clustering result.

Clustering is one of the most fundamental tasks in statistics and machine learning, and numerous algorithms are available to practitioners. Some of the most popular methods, such as K-means (MacQueen, 1967; Lloyd, 1982) and spectral clustering (Shi and Malik, 2000), rely on the key operational assumption that there is one optimal partition of the data into K well-separated groups, where K is assumed to be known *a priori*. While effective in some cases, this flat notion of clustering is inadequate when the data are very noisy or corrupted, or exhibit complex multimodal

behavior and spatial heterogeneity, or simply when the value of K is unknown. In these cases hierarchical clustering provides a more fitting framework because the multi-scale clustering features can be captured by a hierarchy of nested subsets of the data. The expression of these subsets and their order of inclusions—typically depicted as a dendrogram—provides information well beyond typical clustering output. In particular, it serves as a useful global summary of the entire dataset and allows the practitioner to identify and focus on interesting sub-clusters at different levels of spatial resolution.

There are, of course, myriad algorithms for hierarchical clustering. In most cases, however, their use is advocated on the basis of heuristic arguments or computational ease, rather than well-founded theoretical guarantees. Single linkage clustering, for example, is known to be inconsistent in dimensions greater than one (Hartigan, 1981) and suffers from the problem of chaining (Kuiper and Fisher, 1975). Complete linkage and average linkage, on the other hand, capture spherical clusters but are not suitable for complex data with irregular groups (Sneath, 1969). Furthermore, The dendrograms that result from agglomerative hierarchical clustering do not indicate the optimal number of clusters; to obtain a class assignment for each point, the practitioner must specify the desired number of clusters or a threshold at which to cut the dendrogram. Finally, the dendrograms that result from these methods are rarely used as statistical descriptors in their own right.

The high-density hierarchical clustering paradigm proposed by Hartigan (1975) is different. For intuition, suppose a sample $\mathbb{X}_n$ of independent draws from an unknown probability distribution P on $\mathbb{R}^d$ with probability density function f (with respect to Lebesgue measure). The probability of observing a data point inside a subset $A \subset \mathbb{R}^d$ can be computed as

$$P(A) = \int_{x \in A} f(x) dx, \tag{1.1}$$

where the integral is the Lebesgue integral in d-dimensions.[1] It is clear that a set A where f takes on large values has a high probability of containing many of the sample points. Consequently, points in the $\mathbb{X}_n$ are likely clustered inside such a set, so it is natural to define clusters as regions of high density separated by regions of low density (Hartigan, 1975).

This intuition can be formalized by defining the λ-upper level set of f to be

$$L_f(\lambda) = \{x \in \mathbb{R}^d : f(x) \geq \lambda\}, \tag{1.2}$$

for any threshold value $\lambda \geq 0$. A high-density cluster is a maximal connected component of $L_f(\lambda)$ for any λ and the level set tree $\mathcal{T}$ is simply the set of all such clusters. $\mathcal{T}$ has a hierarchical structure in that for any two different high-density clusters C_i and C_j, either $C_i \subset C_j$, $C_j \subset C_i$, or $C_i \cap C_j = \emptyset$ (Hartigan, 1975), and we implicitly include these *parent-child* relationships when discussing a level set tree. Figure 1.1 illustrates how the tree is constructed by identifying upper level sets and high-density clusters for successively higher values of λ. The tree property can be seen in the fact that each high-density cluster is a subset of some cluster portrayed immediately below it but is disjoint from all other clusters at the same level.

This formalization has numerous advantages, particularly with respect to other forms of hierarchical clustering: (1) it provides a probabilistic notion of clustering that conforms to the intuition that clusters are the regions with largest probability to volume ratio; (2) it establishes a direct link between the clustering task and the fundamental problem of nonparametric density estimation; and (3) it allows for a clear definition of clustering performance and consistency (Hartigan, 1981) that is suitable for rigorous theoretical analysis. As a result, level set trees provide a

[1] Technically, A must be a *measurable* subset of $\mathbb{R}^d$ in order for the integral to be well defined; see Billingsley (2012). Throughout we will implicitly assume that measurability holds, although from a practical standpoint, this is inconsequential.

means to represent and visualize data arising from complex and high-dimensional distributions that is statistically accurate in the sense of being a faithful encoding of the level sets of a *bona fide* density function. This property extends to any sub-tree of a level set tree, so it is possible to extract subsets of data at multiple resolutions while retaining the same probabilistic faithfulness, effectively allowing for dynamic and multi-scale clustering.

Despite the methodological advantages and improving theoretical understanding of level set trees, the approach has not caught on as a tool for applied statistical analysis. One reason for this is that estimating level set trees is a computational challenge. The first instinct is to use the level set tree of a suitable density estimate $\widehat{f}$ as a tree estimate $\widehat{\mathcal{T}}$, because for a well-behaved f and a large sample size, $\widehat{f}$ is close to f with high probability (Chaudhuri and Dasgupta, 2010). With a kernel density estimator, for example, the plug-in estimate of $L_f(\lambda)$ at any λ accurately captures the true level set under certain conditions as $n \to \infty$. In fact, the stability of level set and tree estimates can provide guidance for choosing the bandwidth of the kernel density estimator (Rinaldo et al., 2012).

The plug-in approach is not feasible in practice even for low-dimensional data, however, because it requires both evaluation of $\widehat{f}$ on a dense mesh in $\mathbb{R}^d$ and a combinatorial search for connected components over all possible paths between each proposed component.

Many methods have been proposed to overcome these computational obstacles. One family of techniques remains faithful to the idea that clusters are regions of the sample space. Members of this family include histogram-based partitions (Klemelä, 2004; Steinwart, 2011), binary tree partitions (Klemelä, 2005), and Voronoi tessellations (Azzalini and Torelli, 2007). These techniques tend to work well for low-dimensional data, but suffer from the curse of dimensionality because partitioning the sample space requires an exponentially increasing number of cells (Azzalini and

Torelli, 2007). The Steinwart algorithm is notable in that it is proven to find (asymptotically) the smallest value of λ where there are two clusters in the level set tree, and correctly recovers the clusters at this level (as $n \to \infty$). This procedure can be extended recursively to estimate the whole tree, although Steinwart notes that the algorithm is not likely to work well in practice.

A second family of level set tree estimators produces high-density clusters of data points rather than sample space regions. Conceptually, these methods estimate the tree by intersecting the level sets of f with the sample points $\mathbb{X}_n$ and then evaluating the connectivity of each set by graph theoretic means. This typically consists of three high-level steps: estimation of the probability density $\widehat{f}$ from the data; construction of a graph G that describes the similarity between each pair of data points; and a search for connected components in a series of subgraphs of G induced by removing nodes and/or edges of insufficient weight, relative to various density levels.

The variations between the graph-based methods of the latter category are found in the definition of G, the set of density levels over which to iterate, and the way in which G is filtered through the density levels. *Edge iteration* methods assign a weight to the edges of G based on the proximity of the incident vertices in feature space (Chaudhuri and Dasgupta, 2010) or the value of $\widehat{f}$ at the incident vertices (Wong and Lane, 1983) or on a line segment connecting them (Stuetzle and Nugent, 2010). For these procedures, the relevant density levels are the edge weights of G. Frequently, iteration over these levels is done by initializing G with an empty edge set and adding successively more heavily weighted edges, in the manner of traditional single linkage clustering. In fact, single linkage itself is a consistent estimator for $d = 1$, but only weakly consistent for higher dimensions (Hartigan, 1975, 1981; Penrose, 1995).[2]

The Chaudhuri and Dasgupta algorithm (which is a generalization of Wishart

[2]Hartigan also points out that average and complete linkage are "hopelessly inconsistent" for high-density clusters (Hartigan, 1981).

(1969)) is a particularly interesting member of this family because it was the first general level set tree estimator with provable consistency and finite-sample rates of convergence (Chaudhuri and Dasgupta, 2010). Balakrishnan et al. show that for data on or near a low-dimensional manifold embedded in $\mathbb{R}^d$, the Chaudhuri-Dasgupta tree estimator is consistent and converges quickly to the true level set tree at a rate that depends on the manifold dimension but not d (Balakrishnan et al., 2013). Unfortunately, the Chaudhuri-Dasgupta procedure is too computationally demanding for practical use, because it requires estimation of connected components in n^2 filtered similarity graphs (see Chapter 6 for details).

A different class of graph-based level set tree estimator, *point iteration* methods construct G with unweighted edges to indicate similarity, and assign weights to the vertices according to the value of $\widehat{f}$ at the corresponding data point. Subgraphs are created by removing vertices from G in increasing order of these weights. A somewhat complicated version places an edge e_{ij} in G 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 (Menardi and Azzalini, 2013). Another proposed method uses the Gabriel graph for G, with vertices representing data points and edges e_{ij} when there is no other vertex in the smalled closed ball which has x_i and x_j on its border (Oesterling et al., 2011). Oesterling et al. augment this graph by adding a vertex at the midpoint of each edge if the density estimate at the augmented vertex is lower than at the adjacent vertices. Graph filtration can be done very quickly by estimating the join tree with a union-find algorithm (Carr et al., 2003).

A third version of the point iteration estimator class uses a k-nearest neighbor (kNN) similarity graph for G, with vertices for data points and edges e_{ij} if x_i or x_j are among the other point's k-closest neighbors. (Kpotufe and Luxburg, 2011). This method is straightforward and reasonably computationally efficient (although

not generally as fast as construction of the join tree with union-find). Combined with a particular tree pruning scheme, it is also supported by theoretical results. Under the assumption that f is bounded and Hölder continuous: (1) the connected components of any level set can be recovered by subgraphs of G for sufficiently large n; (2) in a finite sample all clusters in $\widehat{\mathcal{T}}$ correspond to a cluster at *some* level of $\mathcal{T}$; yet (3) salient clusters will be recovered; and (4) the pruned tree estimate is a consistent estimator for $\mathcal{T}$. This is the method that we adopt as our standard estimator, although we typically use a more intuitive pruning method for applications.

Because the level set tree is such a fundamental probabilistic object, it is closely related to many other density-based statistical methods. The most obvious relationship is with procedures that estimated individual density level sets and high-density clusters at a fixed value of λ. There are numerous results in this field (see, for example, Polonik (1995); Cuevas et al. (2000); Maier et al. (2009); Rinaldo and Wasserman (2010) and Cadre et al. (2013)), but it is not generally clear how to extend these results to hold over all levels of the level set tree.

One very popular method in this category is Density-Based Spatial Clustering of Applications with Noise (*DBSCAN*) (Ester et al., 1996). DBSCAN finds clusters that consist of core points—defined as having k neighbors within a distance ϵ—and border points—defined as being within ϵ of a core observation whose ϵ-neighborhoods intersect. Although this formalization does not explicitly utilize the pdf, it is clear that core observations are the same data points that have high estimated density while fringe observations have intermediate estimated densities so that DBSCAN clusters are indeed clusters with high density. A recently proposed modification finds the optimal values of k and ϵ in a data-driven fashion, where optimal is defined as in Steinwart (2011) as the lowest density level where two clusters occur (Sriperumbudur and Steinwart, 2012). The clusters returned by DBSCAN at these parameter values are consistent.

The *mean shift* algorithm uses iterative estimates of the density gradient to simultaneously find modes and define clusters as the basins of attraction around those modes (Fukunaga and Hostetler, 1975; Cheng, 1995; Comaniciu and Meer, 2002). This type of method is the subject of a great deal of recent and ongoing research, particularly with the goal of improving computational performance (Carreira-Perpinan, 2006; Yuan et al., 2012). Also of note is an algorithm that find K high-density modes if K is specified *a priori* (Carreira-Perpinan and Wang, 2013).

Mean shift and DBSCAN estimate clusters based on the density function, but do not describe the cluster hierarchy or overall structure of the data. The *OPTICS* algorithm is an exception (Ankerst et al., 1999). Like level set trees, it orders data points by estimated density and allows for the retrieval of clusters at any density level or dendrogram visualization of overall cluster structure. The ordering is designed so that observations that are spatially proximal are also proximal in the OPTICS output; the clusters at a given density level are essentially obtained from DBSCAN.

Finally, the level set tree is similar—especially in name—to the *mode tree* (Minnotte and Scott, 1993), which illustrates the locations of the modes in a one-dimensional pdf as the bandwidth of a density estimator varies. As the bandwidth decreases, more modes appear, and the tree structure is imposed artificially by designating these as either "new" or as children of modes at larger bandwidth values. Contrast this with the level set tree, which represents the hierarchy of high-density clusters over levels in the true density, or for a fixed bandwidth in the case of tree estimation.

This thesis extends level set trees in two primary directions. First, we improve the use of level set trees for clustering, the original motivation behind the method. In Chapter 3 we show how level set trees can be used to quickly retrieve many different clustering results for a single sample, and we show in a range of simulations that clustering with level set trees is at least as accurate as several standard methods in

low-dimensional but challenging problems. For reasons of both numerical instability and poor statistical performance, density-based methods are often not used with high-dimensional data, but we show that high-dimensional level set trees can be effective nevertheless by successfully clustering individuals in the Human Genome Diversity Project, whose genetic information is measured at over 11,500 single nucleotide polymorphisms. An even more fundamental limitation of the level set tree method is the requirement of a pdf, which would seem to preclude the method's application to infinite-dimensional functional data. In Chapter 4 we overcome this hurdle by applying level set trees to pseudo-densities (Ferraty and Vieu, 2006) for functional data rather than *bona fide* density functions. The method is used successfully to cluster North Atlantic hurricane tracks and deterministic fiber streamlines from diffusion weighted neuroimaging (DWI).

The second and more fundamental way in which we extend the level set tree method is to show that the trees solve another fundamental statistical challenge: the estimation and description of structure in a density function. The pdf is a highly intuitive way to describe the distribution of a random variable, but direct visualization of density functions is impossible in more than three dimensions. The power of level set trees is in their simple, low-dimensional representation of the key features of density functions. Toward this end, in Chapter 2 we first describe the α scale that indexes high-density clusters by the mass of the upper level set to which they belong (Rinaldo et al., 2012) and we propose a new κ index to further improve the interpretability and generality of level set trees. Then we describe a procedure for construction of the level set tree dendrogram that emphasizes the clear communication of key tree features, and in Chapter 3 we show that the dendrogram is a highly useful scaffold for interactive data exploration and clustering, particularly for complex or high-dimensional data where direct visualization is difficult. Finally, Chapter 5 presents several ideas for the use of the level set trees as statistics, from

which we can infer and predict things about the underlying distributions. The potential power of these ideas is illustrated in a simple exercise discriminating different Gaussian mixture distributions as well as a comparison of level set trees for DWI fiber streamlines from multiple individuals.

All of the methods described in this thesis are implemented in the Python `DeBaCl` toolbox (for *DEnsity-BAsed CLustering*). This toolbox—and the code used to generate the results in this document—are available online and comments are welcomed by the author. In addition to the standard level set tree algorithm that we prefer to work with, `DeBaCl` also contains the first implementation (of which we are aware) of the Chaudhuri-Dasgupta level set tree algorithm. This algorithm and the challenges of its practical implementation are discussed in Chapter 6.

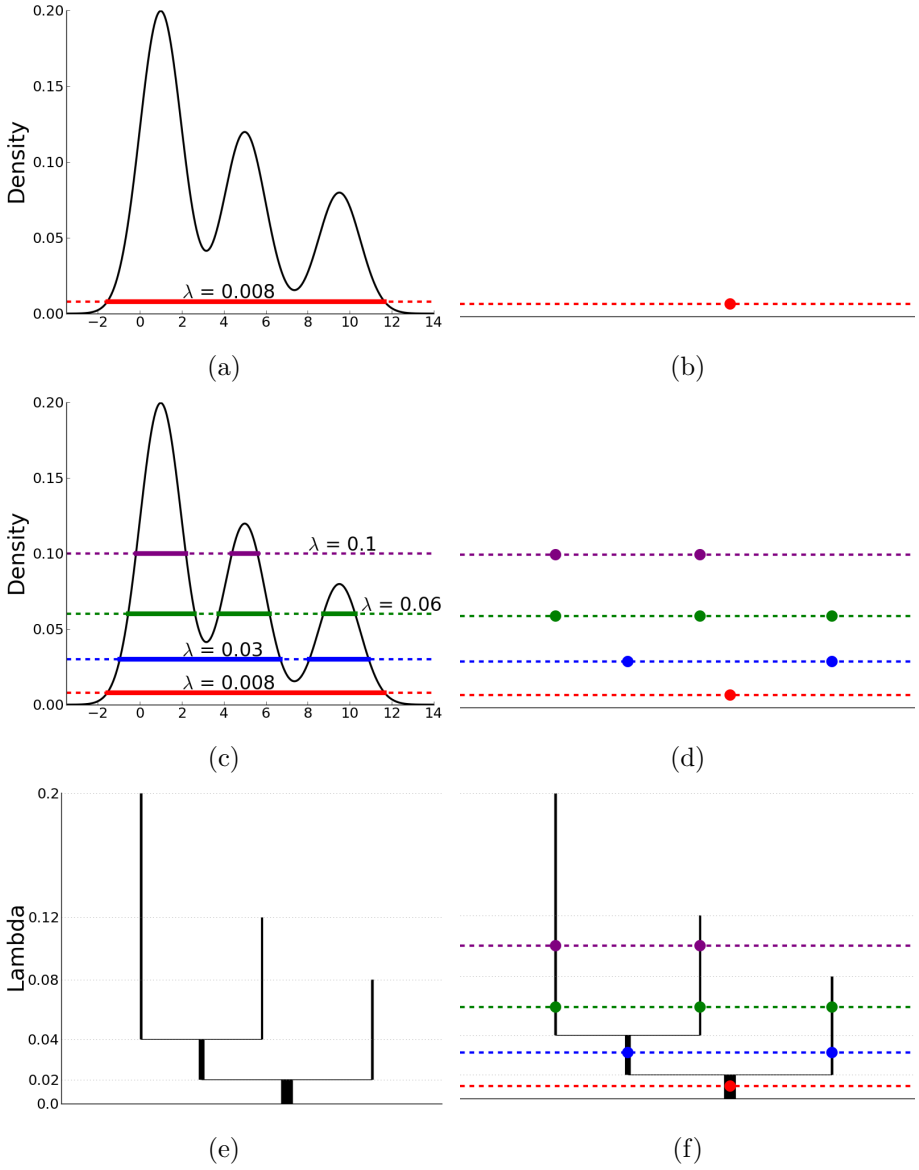


Figure 1.1: Level set tree construction. (a) The density function for a mixture of three Gaussian distributions in $\mathbb{R}^1$ (black curve), with a highlighted density level at $\lambda = 0.008$ (dashed red line), and the upper level set at $\lambda = 0.008$ (solid red line). Note that the upper level set is an interval in $\mathbb{R}^1$, but is shown above the abscissa for illustration. (b) The single connected component of panel (a) is represented on the dendrogram by a single dot at a height of $\lambda = 0.008$. (c) Three more λ-upper level sets and their high-density clusters are added to the pdf. (d) Each high-density cluster of panel (c) is represented by a disc on the dendrogram. (e) The final dendrogram. (f) The dendrogram with selected λ levels and clusters highlighted.

Chapter 2

Tree indexing and dendrograms

2.1 Alpha tree index

Interpretation of λ-indexed level set trees is difficult because the tree depends on the height of f, which is specific to individual distributions. It is not clear without further context, for example, whether we should consider $\lambda = 1$ as a high or low density threshold. To remove this scale dependence, level set trees can also be indexed to the probability content of upper level sets. Let λ_α be the density level such that the λ_α-upper level set of f has probability content no smaller than α (Rinaldo et al., 2012):

$$\lambda_\alpha = \sup \left\{ \lambda : \int_{x \in L_f(\lambda)} f(x)dx \geq \alpha \right\}. \tag{2.1}$$

The map $\alpha \mapsto \lambda_\alpha$ gives a monotonically decreasing one-to-one correspondence between values of α in $[0, 1]$ and values of λ in $[0, \max_x f(x)]$. In particular, $\lambda_1 = 0$ and $\lambda_0 = \max_x f(x)$. Because this map is monotonic, we can express the height of the tree in terms of the probability content α instead of λ without changing the topology (i.e. number and ordering of the branches) of the tree. The α-indexing is generally not, however, a linear re-indexing of λ, so the re-indexed tree will be a deformation of the original tree in which some branches are dilated and others are

compressed.

Indexing a level set tree to α values is useful for several reasons. First, the high-density clusters at a particular α-upper level set are the $1-\alpha$ fraction of "most clusterable" points. In addition, because $\alpha \in [0,1]$ it is easy to compare α-index level set trees across different distributions, even in spaces of different dimension. Finally, the α-index is more effective than the λ index at representing regions of high probability content but low density.

2.2 Cluster trees

For many applications and analyses of level set trees it is useful to switch to the cluster tree representation, where all of the tree nodes in a density mode are collapsed into a single node that contains a birth level λ', death level λ'', and a reference to the node's parent in the cluster tree (Stuetzle and Nugent, 2010).

Formally, for any cluster C_i in the level set tree, let λ'' be the lowest level such that $L_f(\lambda) \cap C_i$ has more than one connected component (i.e. a mode splits) or $L_f(\lambda) \cap C_i = \emptyset$ (i.e. a mode vanishes). Let $\mathcal{E}$ be the collection of all high-density clusters with the same λ'' such that $C_i, C_j \in \mathcal{E} \implies C_i \subset C_j$ or $C_i \supset C_j$. $\mathcal{E}$ corresponds to a single node l of the cluster tree $\mathcal{R}$, associated with a scalar λ''_l and a maximal high-density cluster C_l such that $C_l = C_i$ for some $C_i \in \mathcal{E}$ and $C_l \supset C_j$ for all other $C_j \in \mathcal{E}$. Let $\texttt{parent}_l$ be the smallest node of $\mathcal{R}$ such that $C_{\texttt{parent}_l} \supset C_l$. For notational simplicity, let $\lambda'_l = \lambda''_{\texttt{parent}_l}$ be the birth density level of cluster node l. Note that each cluster tree node also has birth and death mass levels α' and α'', derived directly from Equation 2.1.

2.3 Kappa trees

Although α-indexed level set trees are better than λ-indexed trees at representing probability content in an intuitive way, they are often misinterpreted as showing the mass of cluster tree nodes rather than the mass of each α-upper level set. The κ-index cluster tree facilitates this "size-based" intuition by precisely encoding the probability content of each tree branch rather than density level.

Suppose cluster tree node i represents cluster C. In addition to a parent (defined in Section 2.2), node i has *mass* $M_i = \int_{\mathcal{C}_i} f(x)dx$; *children* $\texttt{kid}_i$, the set of nodes whose clusters are maximal disjoint subsets of C_i in the cluster tree; and *siblings* $\texttt{sib}_i$, the set of nodes $\{j\}$ such that $\texttt{kid}_{\texttt{parent}_i} = (i, \{j\})$.

Assuming the cluster tree is rooted, the κ tree is defined recursively by associating with each node of the cluster tree two scalar values κ' and κ'', as follows:

$$\kappa'_0 = 0 \tag{2.2}$$

$$\kappa'_i = \kappa''_{\texttt{parent}_i} \tag{2.3}$$

$$\kappa''_i = \kappa'_i + M_i - \sum_{j \in \texttt{kid}_i} M_j \tag{2.4}$$

We designate the quantity $\kappa''_i - \kappa'_i$ as the *littoral mass* of cluster tree node i. Figure 2.1 illustrates the areas that constitute cluster littoral masses for an example density function in $\mathbb{R}^1$. The κ tree is ideal for describing and conveying the true size of high-density clusters, and—like the α tree—it allows for comparisons between trees estimated for distributions in very different spaces. A major weakness of the κ tree, however, is instability around flat regions of the density function. Small perturbations in a flat density can dramatically change the littoral mass of nearby clusters, and the more mass contained in a flat region the worse the instability can be.

2.4 Estimation

2.4.1 Lambda trees

Except for the final pruning step, our standard algorithm for estimating λ-indexed level set trees closely follows the algorithm of Kpotufe and Luxburg (2011). It accepts as input the data $\mathbb{X}_n = \{x_1, \ldots, x_n\}$, a smoothing parameter k, a pruning parameter γ, and a distance function. The standard algorithm assumes Euclidean distance $\|.\|$. The algorithm outputs an estimated level set tree $\widehat{\mathcal{T}}$, which is a collection of subsets of $\mathbb{X}_n$, ordered by inclusion relationships. The algorithm comprises the following steps.

1. Compute a similarity graph G, whose vertices correspond to $\mathbb{X}_n$. Typically we use a k-nearest neighbor graph, where edge e_{ij} is included in G if x_i is one of the k closest points to x_j or vice versa. More formally, let $r_k(x_i)$ be the radius of the smallest ball that is centered at x_i and includes $k+1$ data points. Include e_{ij} in G if $\|x_i - x_j\| \leq \max\{r_k(x_i), r_k(x_j)\}$ and $i \neq j$.

2. Estimate the density $\widehat{f}$ and evaluate it on $\mathbb{X}_n$. Our standard method uses a kNN density estimator:
$$\widehat{f}(x) = \frac{k}{n \cdot v_d \cdot r_k^d(x)} \tag{2.5}$$
where v_d is the volume of a d-dimensional ball (Devroye and Wagner, 1977; Maier et al., 2009). Let $\lambda_j = \widehat{f}(x_j)$.

3. Put the density estimates in increasing order and use them to filter the similarity graph. Specifically, for each value λ_j:

   (a) The data upper level set is $L_{\widehat{f}}(\lambda_j) = \{x_i : \widehat{f}(x_i) \geq \lambda_j\}$.

   (b) The filtered graph G_{λ_j} is the subgraph of G induced by $\widehat{L}_{\lambda_j}$.

   (c) Find the connected components of G_{λ_j}.

4. Let $\widehat{\mathcal{T}}$ be the set of all connected components, indexed by the density level (λ) to which they belong and ordered by inclusions.

5. Prune $\widehat{\mathcal{T}}$ by merging components with fewer than γn elements into larger siblings.

Figure 2.2a shows the estimated density and level set tree for a sample drawn from the distribution in Figure 1.1. The observations highlighted on the abscissa and the density function in Figure 2.2a correspond to the high-density mode in Figure 2.2b with the same color.

In this algorithm, the parameter k controls both the smoothness of the density estimate $\widehat{f}$ and the degree of connectivity in the similarity graph G. Larger values of k produce smoother and flatter density estimates with small variances but large biases. As a result, choosing a large k reduces the chance of finding spurious clusters but makes it harder to detect and separate true clusters that are very close to each other. Choosing a small k yields nearly unbiased density estimates with large variances. Based on our experiments we tend to favor relatively large values of k.

The final step is to prune small components of the tree that occur due to sampling variability or insufficient statistical power. Pruning merges components that contain fewer than γn data points into other nearby components. Larger values of γ correspond to more aggressive pruning, where only connected components of large relative size are deemed as separate clusters. On the other hand, setting γ to be very small enhances the resolution of the clustering procedure but increases the chance of seeing spurious clusters.

The algorithm allows for a great deal of flexibility in these choices of density estimator and similarity graph. Most notably, the method works with kernel density estimators and ϵ-neighborhood or Gaussian similarity graphs, and we experiment with generalizations of the density estimator in Chapter 4. We prefer the kNN density estimator for our standard tree estimation because it is computationally

efficient, in that r_k is already known from construction of G.

2.4.2 Computational complexity

Our estimation procedure is efficient, although determining a precise order of complexity is challenging in part because of the high degree of flexibility in how each phase is implemented. The first phase is computation of the similarity graph G, which naïvely requires $\mathcal{O}(n^2)$ computations to find all pairwise distances between the n observations. A major advantage of the kNN similarity graph, however, is that in many cases we need not compute all pairwise distances. For low-dimensional Euclidean data the *k-d tree* provides an exact kNN query that is extremely fast and memory efficient by representing a recursive axis-aligned partition of $\mathbb{R}^d$ in a binary tree (Bentley, 1975). Although complexity analysis is somewhat challenging, construction of the k-d tree is generally of order $\mathcal{O}(n \log n)$ and the kNN query for a single point is $\mathcal{O}(\log n)$ (if we suppose each distance computation takes constant time) (Friedman et al., 1977). To build the similarity graph then requires total complexity of $\mathcal{O}(n \log n)$, which is substantially faster than the $\mathcal{O}(n^2)$ computations required by the naïve computation of all pairwise distances.

For high-dimensional data the k-d tree fails because of the curse of dimensionality; each query requires traversal of a very high number of tree nodes. The k-d tree also fails in the case of functional data stored in lists or a ragged array because the k-d tree requires access to data elements at specific indices (Moore, 2000). In these cases the *ball tree* can provide a fast, exact kNN query if the ground distance satisfies the triangle inequality (Omohundro, 1989; Liu et al., 2006). There are many different flavors, but construction generally requires no more than $\mathcal{O}(n \log^2 n)$ computations (again assuming for simplicity that pairwise distance computation is $\mathcal{O}(1)$ and the query for a single point is of the order $\mathcal{O}(\log n)$ (Omohundro, 1989), yielding a total complexity of $\mathcal{O}(n \log^2 n)$. For matrix-aligned data there are very fast

implementations of both k-d trees and ball trees, and this thesis contributes to the canon of statistical algorithms by providing a Python implementation of the ball tree for functional data that cannot be reshaped to fit in a matrix.

Assuming construction of the similarity graph yields $r_k(x)$ as auxiliary information—which is the case for the kNN graph—computation of the kNN density estimate for each data point requires only constant time. Sorting the vertices in preparation for graph filtration can be done with quicksort, which has $\mathcal{O}(n \log n)$ run time. (Cormen et al., 2009)

The run time of the connected component search over all density levels depends heavily on the particular implementation. The most straightforward method is to simply use a breadth- or depth-first search to identify connected components in G_{λ_j}, for each value of λ_j. For a given level λ_j, the run time for both algorithms is $\mathcal{O}(n_{\lambda_j}+q_{\lambda_j})$ (Cormen et al., 2009), where n_{λ_j} is the number of vertices in G_{λ_j} and q_{λ_j} is the number of edges. Because we use a kNN graph, $q_{\lambda_j} \leq k n_{\lambda_j}$ for any λ_j, and the run time for a given density level can be simplified to $\mathcal{O}(n_{\lambda_j}+k n_{\lambda_j}) = \mathcal{O}((k+1)n_{\lambda_j})$. Assuming k is small and fixed, this further simplifies to $\mathcal{O}(n)$. To find the connected components over all density levels, the run time is $\mathcal{O}(n) + \mathcal{O}(n-1) + \ldots + \mathcal{O}(1) = \mathcal{O}(n^2)$.

By using a more complicated implementation, however, the process of finding connected components can be accelerated dramatically. For applications in image processing, Najman and Couprie propose a method based on the union-find algorithm with run time $\mathcal{O}(n\alpha(n))$, where $\alpha(n)$ is the diagonal inverse of the Ackermann function. $\alpha(n)$ grows extremely slowly; for any practical value of n, $\alpha(n)$ is less than 4. (Najman and Couprie, 2006)

Finally, the worst-case complexity of pruning occurs when the un-pruned tree has n clusters each containing a single point as the leaves. If γ is set to be large, the pruned tree combines all of these leaves into a single root, which requires $2n$ merge

operations, each of which involves only constant-time book-keeping. In sum, if the connected components are found by straightforward breadth- or depth-first search, the total procedure has $\mathcal{O}(n^2)$ run time. If a union-find-type algorithm is used for the component search, the limiting phase is construction of the similarity graph. If all pairwise distances must be computed the total procedure is also $\mathcal{O}(n^2)$, but if a k-d tree or ball tree can be used (for example), then the total run time drops as low as $\mathcal{O}(n \log n)$.

2.4.3 Estimating alpha, kappa, and cluster trees

To estimate the α-indexed level set tree, we follow the standard algorithm from the previous section with one key exception. Instead of filtering the similarity graph over estimated density values λ_j in step 3, we iterate over α-quantiles of the sample densities. For each $\alpha \in \left\{0, \frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}, 1\right\}$, set λ_α to be the α-quantile of the n estimated sample densities. The upper level set and filtered similarity graph are constructed as before, using λ_α. The α level set tree again contains all connected components ordered by inclusions, but each component is now indexed by a value of α.

For any λ level set tree $\widehat{\mathcal{T}}$, estimation of the cluster tree $\widehat{\mathcal{R}}$ follows immediately from the definition of the cluster tree. First, create a cluster tree node l for each unique death level λ'' in $\widehat{\mathcal{T}}$. Associate with l the death level λ'', the (unique) birth level λ' that corresponds to λ'', and a maximal high-density cluster C_l that is the superset of all clusters in $\widehat{\mathcal{T}}$ with death level λ''.

Because the map $\alpha \mapsto \lambda_\alpha$ is one-to-one and monotone, the sets and inclusion relationships are identical in the λ and α level set trees, and the α level set tree can be reduced to the same cluster tree estimate as $\widehat{\mathcal{T}}$. This cluster tree also contains all of the information necessary to estimate the κ tree. Let $\widehat{M_i}$ be the fraction of data contained in cluster tree node i. As with the population κ tree, we estimate

the sample κ tree recursively:

$$\widehat{\kappa}'_0 = 0, \tag{2.6}$$

$$\widehat{\kappa}'_i = \widehat{\kappa}''_{\texttt{parent}_i}, \tag{2.7}$$

$$\widehat{\kappa}''_i = \widehat{\kappa}'_i + \widehat{M}_i - \sum_{j \in \texttt{kid}_i} \widehat{M}_j. \tag{2.8}$$

When estimating the κ tree it is often instructive to identify flat regions of the density function, as these regions are potential sources of tree instability. Loosely speaking, if there is a flat region at density level λ, the set $L_f(\lambda)$ will have substantially larger volume than the set $L_f(\lambda + \epsilon)$. One way to check this in practice is to examine the sequence of sorted density estimates for consecutive subsequences where the density estimate increases by less than a small amount. This method requires explicit specification of the subsequence length and density change thresholds that trigger a warning, and finding such a subsequence does not necessarily imply a flat density region. Nevertheless, it can be useful for flagging areas that need further investigation.

A more direct way to look for instability in the κ tree is to estimate the tree for numerous data subsamples or bootstrap samples, and to look for areas of large variability in the dendrograms. See Chapter 5 for details on analysis of sets of dendrograms.

The fact that the cluster tree estimate is the same for all three types of level set trees, aside from the relevant birth and death values, makes cluster trees the easier object to work with computationally. Table 2.1 shows the information in a cluster tree for a simple Gaussian mixture simulation in $\mathbb{R}^2$. Each row of the table corresponds to a node of the tree, which is associated with birth and death values for α and λ indexing, pointers to the parent and children, and the indices of sample points in the node's cluster. This last set is represented in Table 2.1 as the "size" of

the cluster; from this value it is trivial to compute the birth and death κ values if needed.

node	λ'	λ''	α'	α''	size	parent	children
0	0.000	0.001	0.000	0.007	1000	None	[1, 2]
1	0.001	0.013	0.007	0.971	329	0	[]
2	0.001	0.003	0.007	0.209	664	0	[3, 4]
3	0.003	0.014	0.209	0.984	255	2	[]
4	0.003	0.017	0.209	1.000	280	2	[]

Table 2.1: The summary table for an example cluster tree. Each row of the table represents a node of the cluster tree, and shows the birth and death λ and α levels, as well as the number of observations in the node's cluster, and the parent-child relationships of the node. Note that a node's littoral mass can be obtained by subtracting the size of its children from its own size. This tree was constructed from a sample of 1,000 observations in $\mathbb{R}^2$ from a known Gaussian mixture distribution with three components.

2.5 Dendrograms

The tree property of level set trees (and by extension cluster trees) implies that they can be represented by dendrogram plots. In fact, for distributions with $d > 2$—where density functions cannot be directly visualized—the ability to understand and communicate the structure of a distribution in an intuitive visual way is one of the most powerful features of level set trees (Oesterling et al., 2011). Both Klemelä (2004) and Oesterling et al. (2013) have proposed interesting visualization techniques for density-based trees, but for aesthetic and statistical (see Chapter 5) reasons we describe our own dendrogram construction in this section.

The dendrograms for a level set tree and its corresponding cluster tree are identical, and it is easiest to construct the dendrogram from the cluster tree. For a λ-cluster tree, node i is represented in the dendrogram by a vertical line segment (called a dendrogram *branch* if i is an internal node of the cluster tree or *leaf* if i is a

leaf node) whose endpoints are positioned vertically at λ_i' and λ_i''. A horizontal line segment connects the bottom of the branch for node i to the top of the branch for $\texttt{parent}_i$. For α and κ level set or cluster trees, the vertical positions of the branches are α' and α'' or κ' or κ'', respectively.

Determining the horizontal position of the dendrogram branches is slightly less straightforward than the vertical position. At a high-level, we first recursively construct a set of *silos* that bound the horizontal positions of the branches, then set the branch position within the silos. There are multiple options for each of these phases. For silo construction there are two universal rules: the silo for the root node 0 is the interval $[0, 1]$, and the dendrogram branches in a family are ordered from left-to-right by decreasing mass. Silos are constructed recursively from the root to leaf nodes. Suppose the silo for node i is set at $[a_i, b_i]$. For *uniform* silo construction, equal-width silos are created for each node in $\texttt{kid}_i$. For example, suppose node i has two children: $\texttt{kid}_i = (j, j')$. Then these children have silos $[a_j, b_j] = \left[a_i, \frac{a_i+b_i}{2}\right]$ and $[a_{j'}, b_{j'}] = \left[\frac{a_i+b_i}{2}, b_i\right]$.

For *mass* silos, the width of child silos is proportional to the mass of the child clusters, M_j and $M_{j'}$, with $M_j > M_{j'}$ (without loss of generality). In this case the child silos are

$$[a_j, b_j] = \left[a_i, a_i + \frac{M_j}{M_j + M_{j'}}\right] \tag{2.9}$$

$$[a_{j'}, b_{j'}] = \left[a_i + \frac{M_j}{M_j + M_{j'}}, b_i\right] \tag{2.10}$$

Both of these procedures are extended easily to nodes with more than two children. For both theoretical and aesthetic reasons, we prefer mass silos for all the but the simplest demonstration level set trees (see Section 5 for the theoretical considerations).

There are also (at least) two options for setting the horizontal position of a

dendrogram branch within its silo. Suppose the horizontal branch position for cluster tree node i is z_i. The most aesthetically appealing way to set z_i is the *mean* method:

$$z_i = \begin{cases} \frac{1}{|\texttt{kid}_i|} \sum_{j \in \texttt{kid}_i} z_j & \text{if } |\texttt{kid}_i| \neq 0 \\ \frac{a_i + b_i}{2} & \text{else.} \end{cases} \tag{2.11}$$

To define tree distances in Chapter 5 we instead use the *boundary* method for horizontal positioning. For uniform silos, $z_i = \left[\frac{a_i+b_i}{2}\right]$ for all i. For mass silos:

$$z_i = \begin{cases} \frac{a_i+b_i}{2} & \text{if } |\texttt{kid}_i| = 0 \\ a_i + \frac{M_j}{M_j + M_{j'}} & \text{if } |\texttt{kid}_i| = 2 \\ \frac{1}{|\texttt{kid}_i|} \sum_{j \in \texttt{kid}_i} z_j & \text{if } |\texttt{kid}_i| > 2. \end{cases} \tag{2.12}$$

While this dendrogram construction is useful for defining tree distances, it tends to be aesthetically jarring. Unless otherwise noted, the dendrograms in this thesis are constructed with mass silos and mean-positioned branches. Even when uniform silos are used, the mass of a cluster tree node is represented by the relative thickness of its dendrogram branch.

Figure 1.1 illustrates dendrogram construction for a simple mixture of three Gaussian distributions in $\mathbb{R}^1$. For the level set in 1.1b there is one connected component, which maps to a single disc in the dendrogram canvas (Figure 1.1b). In Figure 1.1c three more upper level sets are shown at higher values of λ; each connected component in these intervals corresponds to a disc of the same color on the dendrogram canvas (Figure 1.1d). When we sweep over all values of λ, the clusters form vertical line segments in the dendrogram (Figure 1.1f). Each high-density cluster in the level set tree has a specific location in the dendrogram, but for simplicity we omit the discs and use only the vertical line segments to represent clusters in the same mode (Figure 1.1e). Each vertical segment in the dendrogram corresponds to

a node of the cluster tree, while branching points in the dendrogram correspond exactly to levels at which two or more modes of the pdf, i.e. new clusters, emerge. Segments that do not split are considered to be high-density modes.

As described in Section 2.1, the existence of a one-to-one map $\alpha \mapsto \lambda_\alpha$ means that switching between λ and α indices does not change the overall shape of the tree. Changing from a λ or α index to a κ index does change the topology, however, which is frequently reflected in very different dendrograms. In particular, the tallest leaf of the κ tree corresponds to the high-density mode with largest empirical mass. In both the λ and α trees, on the other hand, leaves correspond to clusters composed of points with high density values. The difference can be substantial, as demonstrated in the "crater" example of Figure 2.3.

This example consists of a central Gaussian with high density and low mass surrounded by a ring with high mass but low uniform density, shown in Figure 2.3a. The red points belong to the central cluster and constitute 28% of the data, the blue points are higher-density points of the outer ring and make up 37% of the data, and the remaining 35% of the data is low-density, unclustered, gray points. The λ tree (Figure 2.3b) correctly indicates the heights of the modes of the estimated density function $\widehat{f}$ (not shown) but the λ dendrogram tends to produce the incorrect intuition that the outer ring (blue node and blue points) is small. The α tree and dendrogram (Figure 2.3c) ameliorates this problem by indexing node heights to the quantiles of $\widehat{f}$. The blue node appears at $\alpha = 0.35$, when 65% of the data remains in the upper level set, and vanishes at $\alpha = 0.74$, when only 26% of the data remains in the upper level set. It is tempting to say—incorrectly—this means the blue node contains $0.74 - 0.35 = 0.39$ of the mass. This interpretation is precisely the design of the κ tree, however, where we indeed see that the blue node contains $0.72 - 0.35 = 0.37$ of the data, the red node contains $0.63 - 0.35 = 0.28$ of the data, and the unclustered points form the remaining $0.35 - 0.0 = 0.35$ of the data.

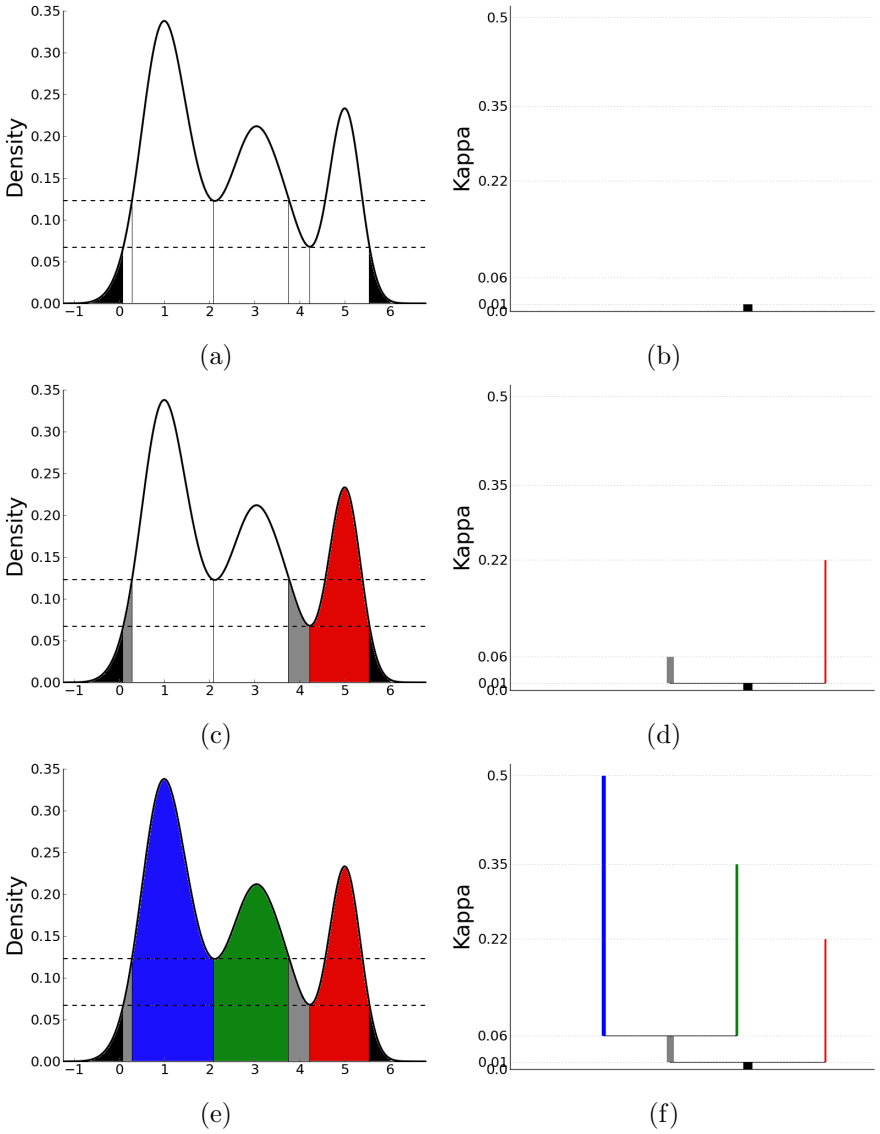
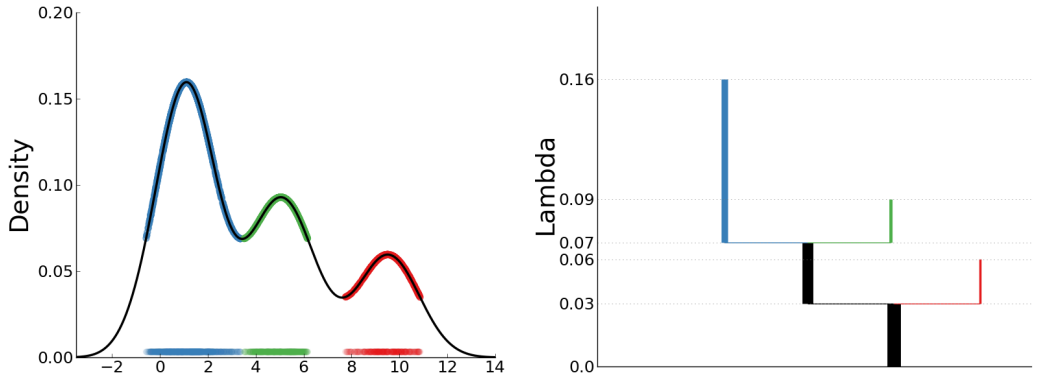


Figure 2.1: Construction of the κ tree. Panels in the left column show the pdf for a mixture of three Gaussian distributions in $\mathbb{R}^1$ (black curve); panels in the right column show the κ dendrogram under construction. (a), (b) As λ increases from 0 to 0.067 (lower dashed line) there is a lone connected component whose littoral mass is the shaded black area under the pdf. This is equal to the length of the black line segment in the dendrogram. (c), (d) At $\lambda = 0.067$ two new clusters are born; the first is a leaf of the tree, so its littoral mass is equal to its cluster mass (red area and dendrogram branch). The second cluster has two children of its own, which appear when λ is further increased to 0.123 (upper dashed line). The littoral mass of this cluster is shown on both the pdf and the dendrogram in gray. (e), (f) The final clusters are leaves, shown in blue and green on the pdf and the dendrogram.



(a) (b)

Figure 2.2: Estimated level set tree. (a) Kernel density estimate for a sample of 2,000 points drawn from the distribution in Figure 1.1. (b) Estimated level set tree for the sample. Color indicates the correspondence between high-density modes of the tree and the observations in these modes.

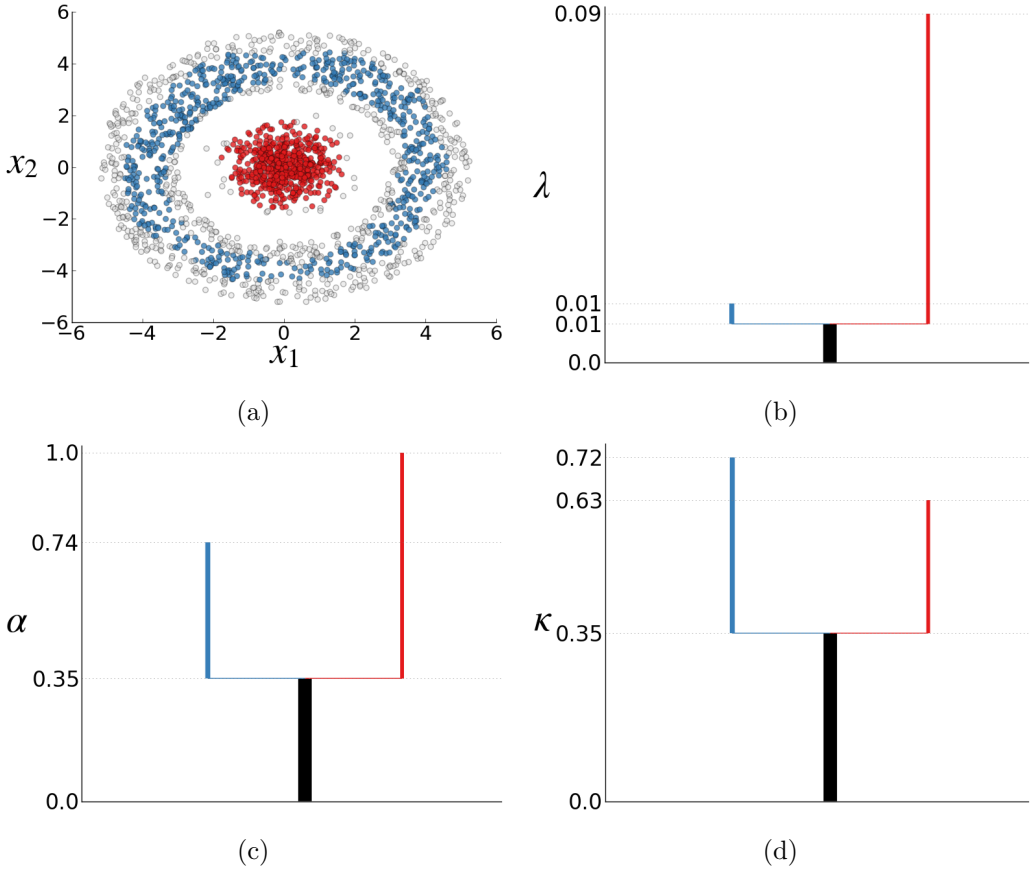


Figure 2.3: Comparison of tree indices. The "crater" distribution has a Gaussian distribution at its core with high-density mean and 30% of the mass, while the outer ring has 70% of the mass but uniformly low density. (a) A sample of 2,000 points drawn from the crater distribution, colored according to membership in branches of the sample level set tree. 28% of the data belong to the red cluster, 37% belong to the blue cluster, while the remainder is background. Note the abuse of notation: x_1 and x_2 refer here to dimensions of a single sample point, not separate observations. This is true for all figures in this thesis. (b) The λ-indexed dendrogram correctly represents the critical levels in the estimated density but gives the incorrect impression that the blue cluster is small. (c) The α-indexed dendrogram. (d) The κ-indexed dendrogram, whose branch heights precisely indicate the littoral mass of each cluster.

Chapter 3

Data exploration and clustering

3.1 Structured exploratory data analysis

The mode hierarchy in a level set tree is a very natural platform for statistically principled interactive exploration of data. For data that are difficult to visualize directly, either because of a high degree of complexity or $d > 3$, it is extremely useful to be able to "walk through" interesting data subsets in a structured way. For low-dimensional data these subsets can be isolated and visualized in feature space but for high-dimensional data they must be projected into other meaningful spaces or summarized numerically.

To facilitate the use of level set tree dendrograms for interactive data exploration, we created two graphical user interface (GUI) software tools that allow a user to select high-density clusters by clicking on dendrogram branches or density/mass levels. We illustrate these tools here with an application to low-dimensional but highly complex *fiber endpoint* neuroimaging data. Usage of the branch selection tool is further illustrated in Chapter 4 for functional data.

The fiber endpoint data is derived from *in vivo* diffusion weighted brain imaging (DWI) collected at the Scientific Imaging and Brain Research Center at Carnegie Mellon University in 2012 for 30 neurologically healthy controls (the *CMU-30* group).

From the DWI data, deterministic fiber tractography was used to simulate smooth 1-dimensional manifolds (with boundaries) called *fiber streamlines* that represent tracks of strong water diffusion in the brain (Hagmann et al., 2006). Streamlines are represented as variable-length sequences of points in $\mathbb{R}^3$. Further detail about participants, image acquisition, and fiber tractography can be found in Kent et al. (2013) and at http://www.psy.cmu.edu/~coaxlab/?page_id=423.

For the analysis in this section, 10,000 fiber streamlines were mapped from the cortex into the striatum for a single subject. To keep the demonstration simple, we use only the last point of each streamline, located (by construction) in the striatum. The raw data are shown in Figure 3.1. Although two views of the data are shown, it is clear that a thorough understanding of the complex data structure is impossible from a traditional scatterplot, due to overplotting effects.

The dendrogram provides a scaffold for isolating and visualizing coherent clusters of data, ameliorating the overplotting problem to a great extent. Figure 3.2 shows how the dendrogram and the branch selection GUI can be used to walk through the complex fiber endpoint data. The points associated with one of the large branches that begin near $\alpha = 0$ are part of a spatially distinct cluster in the dorsal portion of the striatum, specifically the dorsal caudate nucleus. By zooming in on different dendrogram branches, we see that these correspond to coherent sub-clusters at various "resolutions". These are known, anatomically distinct sub-regions of the projections (Haber and Knutson, 2010).

The second GUI tool allows a user to select a density or mass level in order to visualize or retrieve high-density clusters. Figure 3.3 illustrates the high-density clusters associated with two mass levels near interesting splits in the level set tree. This tool allows us to quickly and interactively visualize how a level set tree encodes high-density clusters at successively higher density or mass levels. For this dataset, it reveals both the obvious split between points in the caudate and putamen and

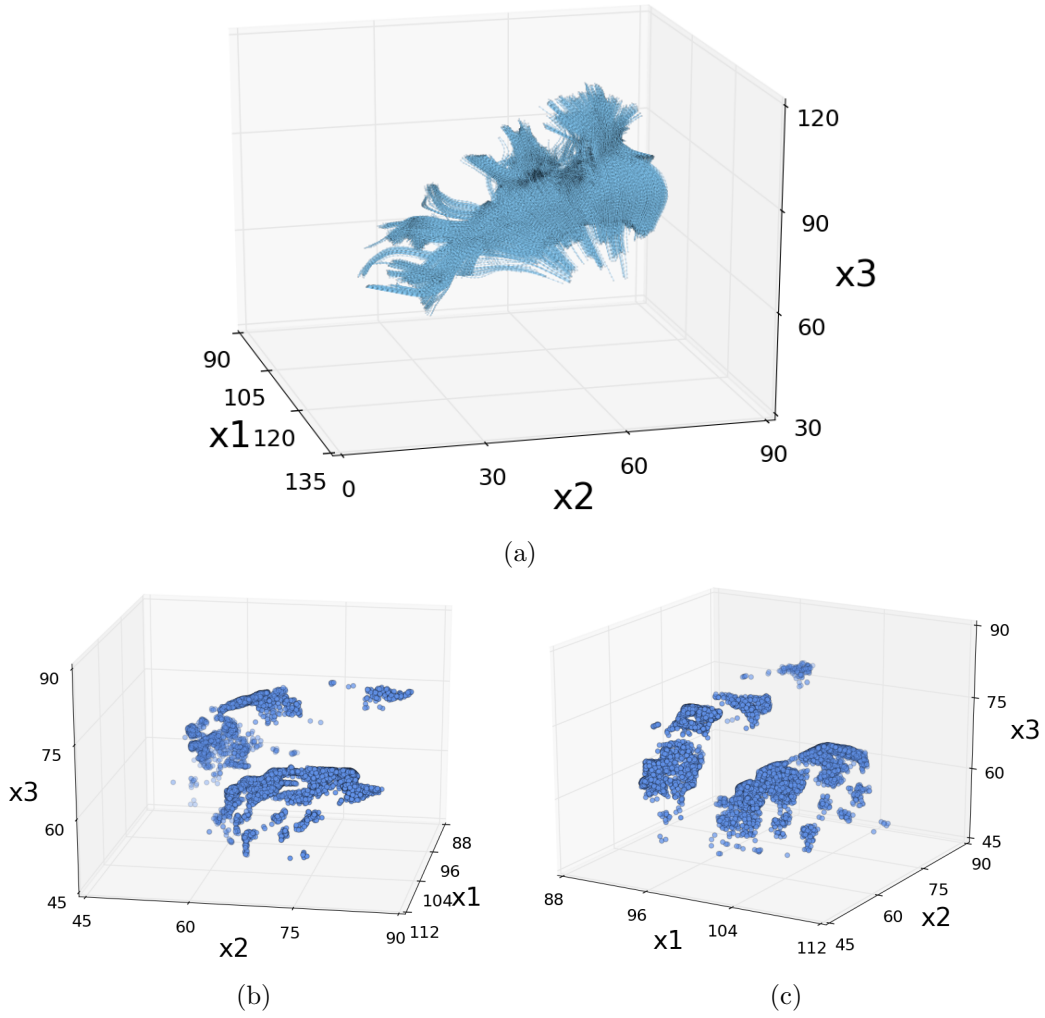


Figure 3.1: Fiber streamlines and streamline endpoints for one subject in the CMU-30 group. (a) 10,000 streamlines mapped from the middle frontal gyrus to the striatum, with a deterministic fiber tractography algorithm, based on DWI data. (b), (c) Striatal endpoints of the 10,000 streamlines, from two camera angles.

not-so-obvious high-density modes within each region.

3.2 Cluster retrieval options

Although the level set tree provides a great deal more information about data topography than the typical clustering method, sometimes the goal is to obtain an

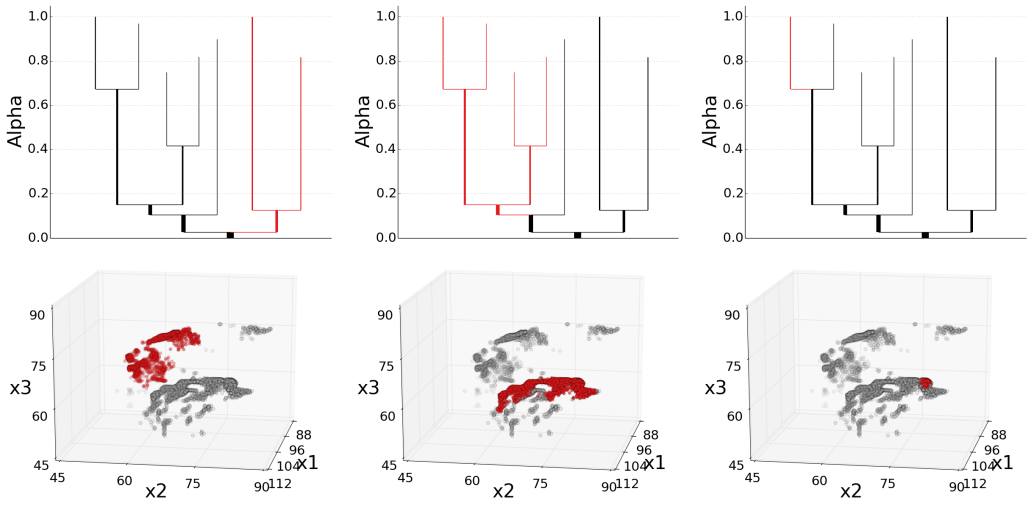


Figure 3.2: Interactive exploration of data subsets with the dendrogram. The top row shows the dendrogram for the striatal endpoints in Figure 3.1, with different branches highlighted. The data subsets corresponding to the selected branches are illustrated in the bottom row.

unsupervised partition of the data. When this is the case, level set trees provide several ways to define clusters. The level selection GUI tool described in Section 3.1 is based on the most obvious method; because the level set tree is a compilation of high-density clusters over many density or mass values, it is natural to select a value of λ or α of particular interest and retrieve the corresponding high-density clusters. In fact, the level λ (or α) provides a clustering resolution of sorts, with lower values of λ corresponding to larger and coarser clusters and higher values to smaller, more sharply defined clusters.

In addition to its definitional nature, this *level-set* method conveys the most intuitive sense for where the highest density data subsets are located. It also allows the investigator to control the number of points in the clusters; choosing a low mass level produces clusters that contain most of the data, while high mass thresholds produce clusters with only the peaks of the data modes. Finally, this method avoids the need to specify *a priori* the number of clusters, which must be chosen heuristically in

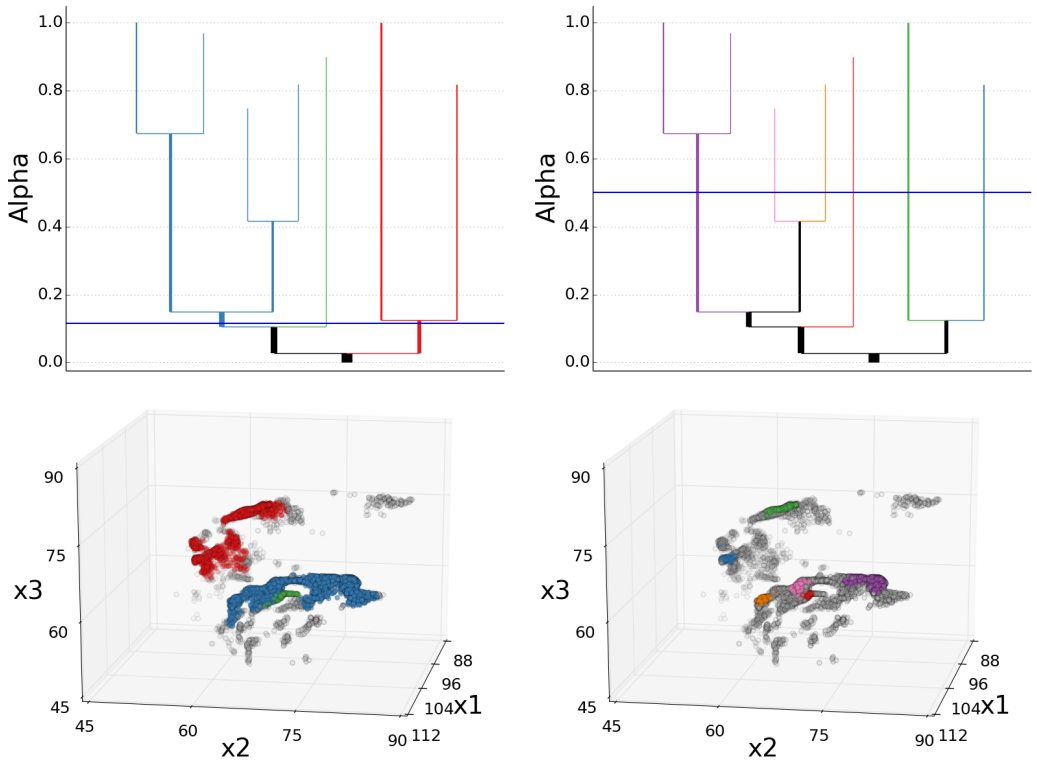
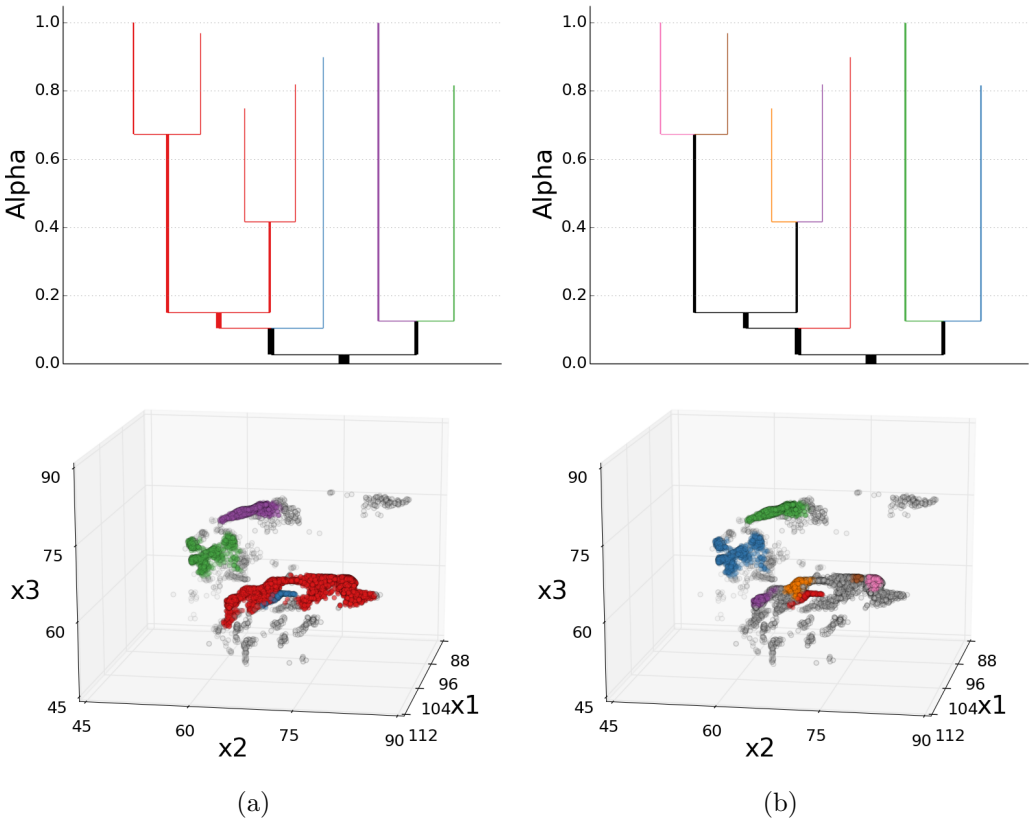


Figure 3.3: Interactive exploration of high-density clusters with the dendrogram. The top row shows the dendrogram for the striatal endpoints in Figure 3.1, with two different α levels selected (horizontal blue line). The bottom row shows the high-density clusters for the respective α levels.

many popular clustering methods (k-means, for example).

On the other hand, if the clustering task demands a pre-set number of clusters, K, this can be done with a level set tree by identifying the first K disjoint components to appear in the tree as the level increases from $\lambda = 0$. Unlike k-means (and related methods), however, there is no guarantee that there will be K disjoint nodes in a level set tree. This *first-K* option is illustrated for the fiber endpoint data in Figure 3.4a.

The drawback of level-set and first-K clustering is that they require an arbitrary choice of either λ, α, or K. *All-mode* clustering, which uses each leaf of a level set tree as a cluster, automatically chooses the number of clusters and avoids the



(a) (b)

Figure 3.4: Additional options for assigning cluster labels from the level set tree. The top row shows the dendrogram for the striatal endpoints in Figure 3.1 and the bottom row shows the corresponding high-density clusters (color indicates correspondence between dendrogram and data). (a) The *first-K* method, which returns the clusters produced by the first $K-1$ splits in the tree as α increases from 0. (b) The *all-mode* method, which designates each leaf of the cluster tree as a cluster.

arbitrary choice of a density or mass level at which to cut the tree (Azzalini and Torelli, 2007) (see Figure 3.4b for an example). This method does remain sensitive to the choice of smoothing and pruning parameters, however. In particular, for a given degree of pruning, this method tends to produce more and smaller clusters than level set clustering.

Each of these three methods assigns cluster labels to only a fraction of the sample, and leaves the remaining *background* points unlabeled (see Figures 3.3 and 3.4 for examples). The fraction varies greatly depending on the choices of cluster

labeling method and parameter values. Having unlabeled background points can be an advantage in that it intelligently removes outliers, but it is a disadvantage if a full data partition is needed. In the latter case, any classification method can be used to label the unclustered points, although more sophisticated methods take advantage of the already clustered data (see for example Azzalini and Torelli (2007)).

Taken together, the advantages of a level set tree approach—avoiding the need to specify the cluster number, multiple cluster retrieval options, visualization of many cluster permutations, interactive clustering exploration, and automatic outlier removal—allow the practitioner to gain greater insight into the clustering behavior of a data set, using fewer assumptions than would be necessary for standard methods. Furthermore, all of these features can be explored from the level set tree structure, avoiding the need to run the clustering algorithm repeatedly.

3.3 Performance Comparison Experiments

To analyze the effectiveness of level set tree clustering we tested it against several standard clustering methods in a range of simulations. The experiment suggests that level set clustering is at least as accurate in practical clustering tasks, particularly with challenging non-convex clusters.

3.3.1 Methods

We compared the performance of level set trees against several popular methods: k-means++ (Arthur and Vassilvitskii, 2007), Gaussian mixtures (Hastie et al., 2009), hierarchical agglomeration with the Ward criterion (Hartigan, 1975), hierarchical agglomeration with the single linkage criterion (Hastie et al., 2009), spectral clustering (von Luxburg, 2006), diffusion maps (Coifman and Lafon, 2006), and DBSCAN (Ester et al., 1996). Where possible, the methods were given the true number of clusters K in order to isolate the effectiveness of the algorithms from heuristics

for choosing K.

The methods were tested in three scenarios in $\mathbb{R}^3$ with an increasing degree of similarity to the hard problem of clustering fiber endpoints. The easiest scenario was a mixture of six Gaussian distributions, the moderate scenario was a mixture of three Gaussian distributions and three noisy arcs, and the difficult scenario was resampled from a set of 10,000 fiber streamline endpoints from a real striatal fiber tractography dataset, with Gaussian noise added to each resampled observation. For the hard scenario the "ground truth" clusters were determined by a careful, manually tuned application of level set tree clustering. The group means were contracted toward the grand mean by a *degree of ease* coefficient ρ, which took 20 values on a grid ranging from 0.1 to 1.2 (i.e. larger values mean an easier clustering task). Finally, for each scenario and value of ρ, 20 datasets were simulated with 5,000 points apiece. Figures 3.5a, 3.5c, and 3.5e show examples of the simulation scenarios at large values of ρ.

For level set tree clustering, the smoothing and pruning parameters were set to $k = 150$ and $\gamma = 50$, values which were intentionally not fine-tuned to optimize the results. We retrieved cluster labels from each tree by finding the lowest density level λ such that the upper level set $L_{\hat{f}}(\lambda)$ had K high-density clusters. This is very similar, but not identical to the first-K cluster retrieval method. Note also that this ignores the ability of level set trees to identify the correct number of clusters with the all-mode retrieval method. Background points were assigned to a cluster with a kNN classifier ($k = 11$), which is a straightforward but sub-optimal approach.

Both types of agglomerative hierarchical clustering were implemented with the `R hclust` function (R Core Team, 2012). K-means++, Gaussian mixture modeling (GMM), and DBSCAN were implemented with the Python module `scikit-learn` (Pedregosa et al., 2011). For DBSCAN we set the neighborhood parameter ϵ to be the second percentile of all pairwise distances and the level set parameter (i.e. the number of neighbors required for a point to be a core point) to be the first percentile

of pairwise distances. Note that DBSCAN does not allow K to be specified, making it difficult to compare to other methods.

We used our own implementations for spectral clustering and diffusion maps. For spectral clustering we constructed a symmetric kNN graph on the data, with k set to one percent of the sample size. The points in the first percentile of degree in this graph were removed as outliers. For diffusion maps we used a complete similarity graph with Gaussian edge weights:

$$e_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right) \tag{3.1}$$

with σ set to twice the squared median of all pairwise distances (Richards et al., 2009a). For both spectral and diffusion map clustering we used the random walk form of normalized graph Laplacian:

$$L = D^{-1}(D - W) \tag{3.2}$$

where W is the similarity graph adjacency matrix, and D is the diagonal degree matrix for W (von Luxburg, 2006). For diffusion maps the i'th eigenvector ψ_i is scaled by a function of its corresponding eigenvalue ξ_i:

$$\psi_i' = \left(\frac{1 - \xi_i}{\xi_i}\right)\psi_i \tag{3.3}$$

which creates a multi-scale diffusion map (Richards et al., 2009b). For spectral clustering and diffusion maps we used k-means++ to cluster the data after it was projected into the eigenspace, and for spectral clustering we used a kNN classifier to assign outliers to clusters.

3.3.2 Results

Not surprisingly, for the pure Gaussian mixture scenario all methods achieved perfect identification of the true clusters when the groups were well separated (Figure 3.5b). Single linkage hierarchical clustering had a very high error rate even at medium degrees of separation between clusters, due to the well-studied problem of chaining, discussed in Chapter 1. The density-based methods DBSCAN and level set trees also required more separation between clusters before achieving the same error rate as parametric methods, possibly due to the challenge of assigning low-density points to a cluster.

The results are more difficult to interpret for the scenario with three arcs and three spherical Gaussians (Figure 3.5d). Single linkage hierarchical clustering again required the most separation between clusters to achieve highly accurate classification. Spectral clustering was perfect when the clusters were well-separated and was as good as any other method when the clusters were very close, but performed poorly at mid-range degrees of separation. The closely related technique of diffusion maps actually became less accurate at large degrees of separation. Level set tree clustering performed poorly for tightly packed clusters, but was comparable to the parametric methods (k-means++, Ward linkage, and Gaussian mixtures) for medium and well separated clusters.

While the second simulation type is more challenging than the Gaussian-only mixture, it is still much easier to cluster than the highly non-convex, multi-resolution data in fiber tractography datasets. In the most complex and realistic setting with points resampled from real fiber tractography data, the parametric methods performed poorly, achieving only about 70% accuracy, even when the clusters are well separated (Figure 3.5f). Each of the nonparametric methods (level set clustering, DBSCAN, diffusion maps, and spectral clustering) performed best at some degree of separation, making it difficult to identify clearly superior or inferior methods.

DBSCAN and level set trees had accuracies somewhat less than 100% even for well separated clusters, probably due to the problem of assigning low-density points to clusters. A more nuanced classifier for this step in level set tree clustering would likely improve the results for level set trees in particular.

3.4 High-dimensional experiments

For reasons of numerical instability and poor statistical performance, density-based methods are not often used with high-dimensional data. However, a handful of theoretical results suggests that level set trees may still be useful in this domain, despite their reliance on density estimation. First, note that estimating the level set tree and retrieving "good" high-density clusters does not necessarily require accurate density estimation; these are two distinct tasks. In fact, inconsistency in a density estimator does not necessarily imply inconsistency in the resulting level set tree (Hartigan, 1981). As with pseudo-density estimation for functional data in the following chapter, the level set tree requires only the order relationships between points based on their estimated densities (Hartigan, 1981).

More concretely, suppose an i.i.d. sample and a kernel density estimator $\widehat{f}(x)$ with a fixed nonzero bandwidth h. The *mollified density* is

$$f_h(x) = \mathbb{E}(\widehat{f}(x)), \tag{3.4}$$

which is equivalent to convolving the distribution P with a mollifier kernel. By fixing h we introduce bias, which allows the density estimate to converge to the mollified density at a rate that does not depend on d: $\sup_x |\widehat{f}(x) - f_h(x)| = \mathcal{O}(\sqrt{\log n/n})$, a.e. P (Rinaldo and Wasserman, 2010). It seems possible to accurately estimate the level set tree of $f_h(x)$ regardless of the dimension, and if $f_h(x)$ contains the same clustering structure as P, then the level set tree will be accurate for P as well. The

result of Balakrishnan et al. mentioned in the Introduction appears to confirm this, showing that that level set tree estimation is possible in a high-dimensional setting if the data lie near a manifold with lower dimension (Balakrishnan et al., 2013).

To illustrate the potential of level set trees for high-dimensional data analysis, we applied the method to a human population genetics dataset. This analysis also illustrates how the lack of a dominant heuristic for choosing k is often not a problem for practical data analysis. We initially chose k to be larger than we thought optimal, in order to oversmooth the density and connectivity estimates. Then we reduced k until the clusters fragmented in non-useful ways.

3.4.1 Methods

The Human Genome Diversity Project (HGDP) dataset includes 1,064 individuals sampled from 53 geographically isolated populations, chosen to represent a large range of worldwide human genetic diversity (Li et al., 2008). The full dataset contains genotypes at about 660,000 biallelic single nucleotide polymorphisms (SNPs) for each individual; it is publicly available at `http://www.hagsc.org/hgdp/files.html`.

Our data preprocessing steps followed the recipe in Crossett et al. (2010). First, a small subset individuals were removed due to an excess of missing data or likely relation to another subject, based on the results of both Crossett et al. and Rosenberg (2006). Second, SNPs with excessive missingness, low rare allele frequency, or Hardy-Weinberg disequilibrium were removed from the data. Finally, a relatively small number of SNPs were chosen by traditional hierarchical clustering (with Ward linkage) to have low correlation with each other. The processed data for this experiment had 931 individuals in 53 populations, genotyped at 11,667 SNPs. Further detail about the data cleaning process can be found in Crossett et al. (2010).

The goal of level set tree estimation in this analysis is to identify the hierarchy

of high-density clusters of individuals in the sample, ideally capturing the correct membership in populations. Because the individuals lie in such a high dimension ($d = 11,667$), we used a kNN pseudo-density estimate with the dimension set to $d = 1$. Although the pseudo-density is not appropriately normalized, its order statistic is the same as that of the properly normalized density and it is easier to compute. The order statistic—which ranks the individuals by genetic proximity to their neighbors—is the key input to the level set tree estimator (see Chapter 4 for more detail on pseudo-densities). For all trees the pruning parameter was set so remove clusters with fewer than 3 people.

3.4.2 Results

The level set tree in Figure 3.6a was constructed with $k = 40$ individuals, which intentionally oversmooths the density and connectivity estimates (note 40/931 is a much larger fraction than was used in the fiber endpoint analyses). Not surprisingly, the all-mode clusters from this level set tree capture only coarse continent groups, not individual populations (Figure 3.6b). The map does show that there are no errors in the clusters; all of the individuals in each cluster belong to only one continent group. The tree also also indicates that the Europe/Middle East/Central Asian and East Asian groups are more similar to each other than either is to the African group or the American group (note these are indigenous American populations).

Reducing the smoothing parameter to $k = 6$ yields a more informative hierarchy that successfully captures both continent groups at low values of α and individual populations at the leaves of the tree. With the all-mode clustering from this tree, there are very few individuals incorrectly clustered, and many of the individual populations are identified. Several of the European populations are notably absent, however, from the list of clusters. The ability of the tree to encode clusters at multiple "resolutions" can be seen in the maps in Figure 3.8, which were produced by cutting

the tree at α levels illustrated in Figure 3.7. When $\alpha = 0.13$, there are large clusters for Europe/Middle East, East Asia, Africa, and a handful of individual populations in American and Oceania. When the tree is cut at $\alpha = 0.47$, on the other hand, many more populations are identified by individual clusters. The absence of multi-colored pie charts on the map in Figure 3.8a indicates few classification errors. Furthermore, nodes of the tree that are closer together tend to correspond to populations that are nearby physically as well, a sign that the method works well.

Because of its multi-scale nature, the level set tree provides a much greater deal of interpretability for this type of problem than popular methods like spectral clustering. There is an important caveat, however: the intentionally extreme sampling bias in these data erodes some of the most useful aspects of the level set tree. In particular, the heights of the branches are meaningless, and the order in which the branches occur is heavily biased. Nevertheless, this density-based method is able to recover populations with a high degree of accuracy and power in a space with over 11,500 dimensions.

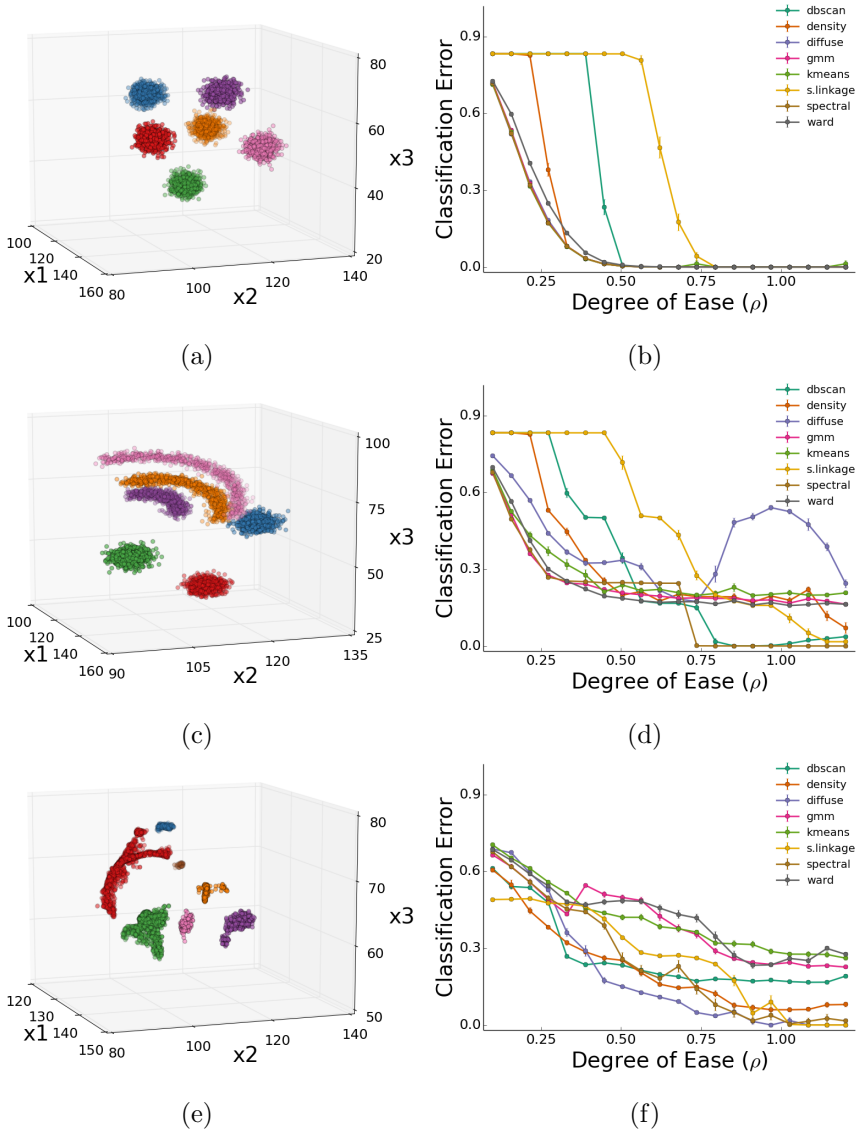
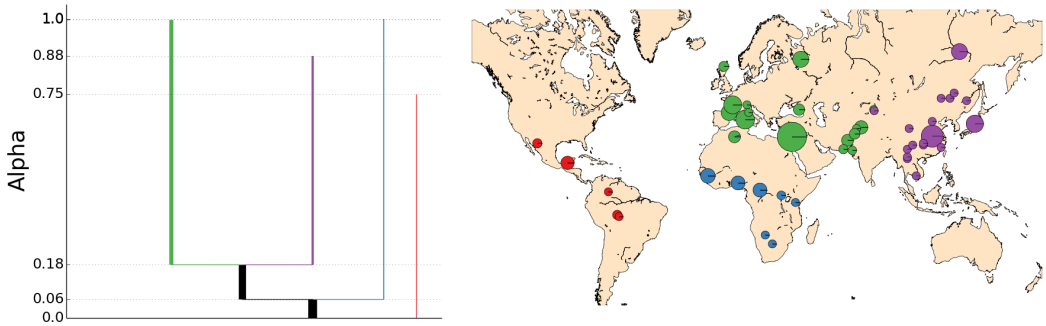
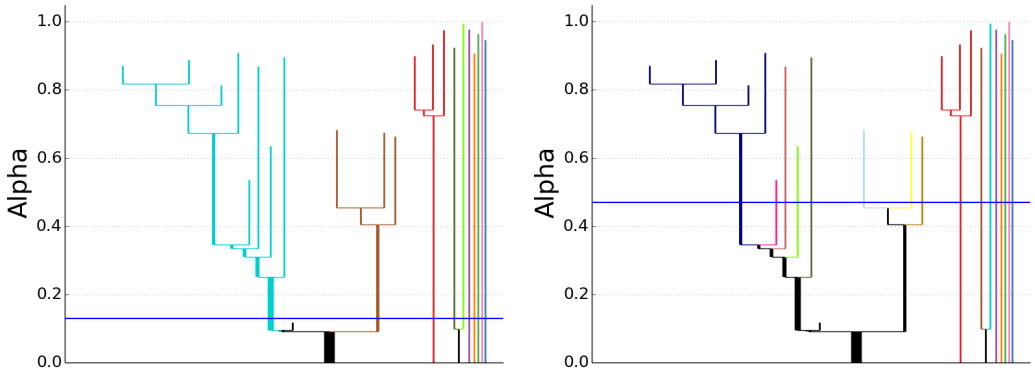


Figure 3.5: Clustering accuracy of several methods in simulated data. (a), (c), (e) Example draws from each of the three simulation scenarios (Gaussians, Gaussians and quarter circles, and resampled striatal fiber streamline endpoints), with observations colored according to true group membership. (b), (d), (f) The error rate for each type of simulation. For each simulation scenario the difficulty was increased by shrinking the group means toward the grand mean by a factor ρ. The mean and standard error of misclassification error are reported for 20 samples of 5,000 points for each the eight clustering methods: DBSCAN (dbscan), level set tree clustering (density), diffusion maps (diffuse), Gaussian mixture models (gmm), k-means++ (kmeans), hierarchical clustering with single linkage (s.linkage), spectral clustering (spectral), and hierarchical clustering with linkage by the Ward criterion (ward).



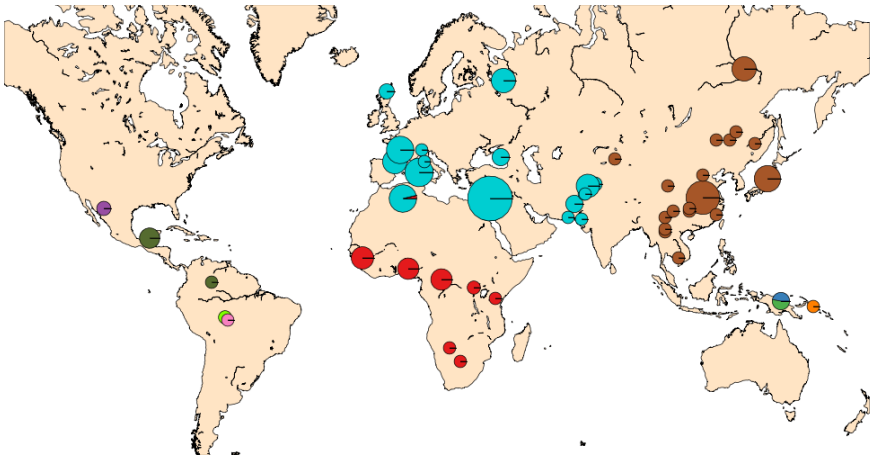
(a) (b)

Figure 3.6: Oversmoothed level set tree result for population genetics. (a) When the smoothing parameter k is set to 40, the estimated level set tree for the HGDP population genetic data has only four all-mode clusters. (b) The map shows these clusters describe continent groups well, but do not capture more detailed population affiliations. Each pie chart on the map represents a true population, and the slices of each pie represent the contribution from each cluster (matched by color to the dendrogram). The clusters in this result are very well matched to populations.

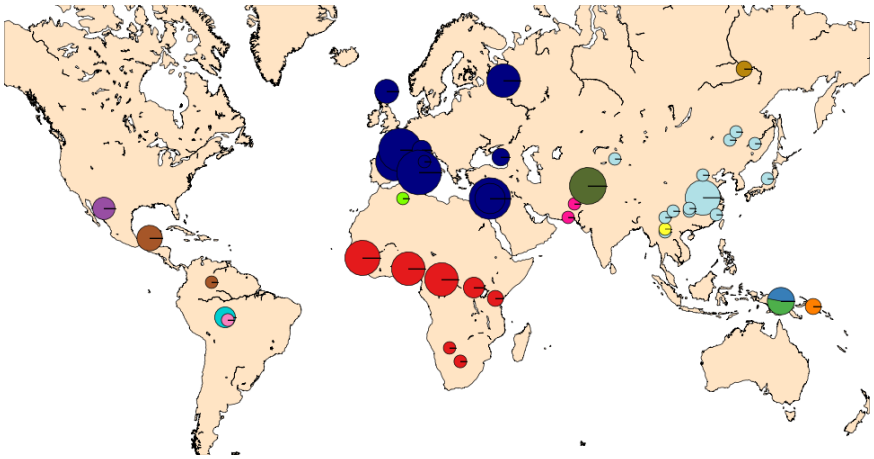


(a) (b)

Figure 3.7: Exploring high-density population genetics clusters. The level set tree is constructed with $k = 6$, yielding a more detailed and multi-scale set of high-density clusters. (a) Cutting the tree at a low α level yields continent groups and highly dissimilar populations (Figure 3.8a). (b) Cutting at a higher α level produces high-density clusters that better capture individual populations (Figure 3.8b).



(a)



(b)

Figure 3.8: Maps indicating clusters for the dendrograms in Figure 3.7. Each pie chart on the map represents a true population, and the slices of each pie represent the contribution from each cluster (matched by color to the respective dendrogram in Figure 3.7). (a) High-density clusters for a low α cut of the tree (Figure 3.7a). Populations in Papua New Guinea and the Americas are identified, but only continent groups are recovered in Africa, Eurasia, and East Asia. (b) High-density clusters for the high α cut (Figure 3.7b) correspond better to individual populations.

Chapter 4

Pseudo-densities

4.1 Pseudo-densities

The fiber endpoint dataset used to illustrate level set tree EDA and clustering is a very impoverished distillation of the original set of complete fiber streamlines. Recall from Section 3.1 that we think of a fiber streamline as a set of points X sampled irregularly along a random curve in $\mathbb{R}^3$ (Hagmann et al., 2006). Describing the topography of fiber streamline datasets would yield great insight into the anatomical connections in human white matter, but the high degree of complexity of fiber streamlines makes this a difficult challenge (Wedeen et al., 2008, 2012; Hagmann et al., 2008; Yeh et al., 2013). The overwhelming majority of efforts to date have simplified the problem to a clustering task, but most approaches are statistically unmotivated, too complex to replicate reliably, or fail to scale to datasets densely sampled across whole brains (Garyfallidis et al., 2012). For a review, see O'Donnell et al. (O'Donnell et al., 2013). Some of the most recently proposed methods do overcome some of these challenges. *QuickBundles*, for example, is a memory-efficient, linear time algorithm proposed by Garyfallidis et al. for preliminary data reduction of extremely large fiber streamline sets (Garyfallidis et al., 2012). The authors report clustering 100,000 fiber streamlines in under 20 seconds, but the method lacks a

statistical foundation and is not invariant to permutations of the data. Guevara et al. also tackle the challenge of scale, segmenting up to 1.6 million fibers in a single analysis (Guevara et al., 2011). Although their protocol contains elements of density-based and hierarchical clustering, it is a hodgepodge of various techniques that seems unlikely to generalize well to all streamline datasets.

We take a different tack and adapt the level set tree to work with functional data like fiber streamlines, leveraging the ability of level set trees to emphasize data structure over clustering at a single density level and to scale efficiently to large data sets. The primary obstacle is that while probability distributions for these random functions are well-defined, they cannot be represented with pdfs (Billingsley, 2012). To get around this problem, we use a *pseudo-density function* instead of a pdf to measure the similarity between non-Euclidean data points and the overall connectivity of the space (Ferraty and Vieu, 2006). Specifically, we use the kNN density estimate as in Equation 2.5 but expunge the term v^d and set d arbitrarily to 1:

$$\widehat{f}_{\text{pseudo}}(X) = \frac{k}{n \cdot r_k(X)}, \tag{4.1}$$

where $r_k()$ is based on a distance relevant to the functional space. In general this does not yield a *bona fide* density function, but it is sufficient to induce an ordering on the data points based on each point's proximity to its neighbors. The similarity graph G is constructed from the chosen distance function, and $\widehat{f}_{\text{pseudo}}$ is then used to filter the graph to find connected components at successively higher pseudo-density levels, as described in Section 2.4.

4.2 Phonemes

Because fiber streamlines are difficult to visualize, we first illustrate pseudo-densities and functional level set trees with two simpler datasets. The *phoneme* dataset,

from Ferraty and Vieu (2006) (originally from the TIMIT Acoustic-Phonetic Continuous Speech Corpus, NTIS, U.S. Department of Commerce via Hastie et al. (1995)), contains log-periodograms of 2000 instances of digitized human speech, divided evenly between five phonemes: "sh", "dcl" (as in "dark"), "iy" (as in the vowel of "she"), "aa", and "ao". Each recording is treated as a single functional observation, which we smoothed using a cubic spline. The smoothed data are shown in Figure 4.1.

Each phoneme observation is recorded at the same 150 frequencies, so we use Euclidean distance to measure the distance between each pair of functional observations. We could use $d = 150$ to compute a valid density estimate, but for illustration purposes we compute $\widehat{f}_{\text{pseudo}}$ instead. Figure 4.2 shows each observation colored according to its pseudo-density estimate; those that are very similar to their neighbors ($k = 20$) clearly have high pseudo-density (shown by more intense shades of purple).

The level set tree and the all-mode clusters are shown in Figure 4.3, along with the mean function for each high-density cluster. The clustering results are highly accurate: although the second cluster captures both the "aa" and "ao" phonemes, these are seen in Figure 4.1 to be extremely similar groups. There are no other errors, although a substantial fraction of the data is left unlabeled (Table 4.1). More importantly, the hierarchy of the level set tree provides additional information about these data, namely that the "sh" and "iy" phonemes are very similar to each other while the "dcl" phoneme is dissimilar from the other four (with respect to Euclidean distance).

4.3 Hurricane tracks

The functional observations in the phoneme data are sampled at the same identical 150 frequencies, allowing us to use simple Euclidean distance and making it some-

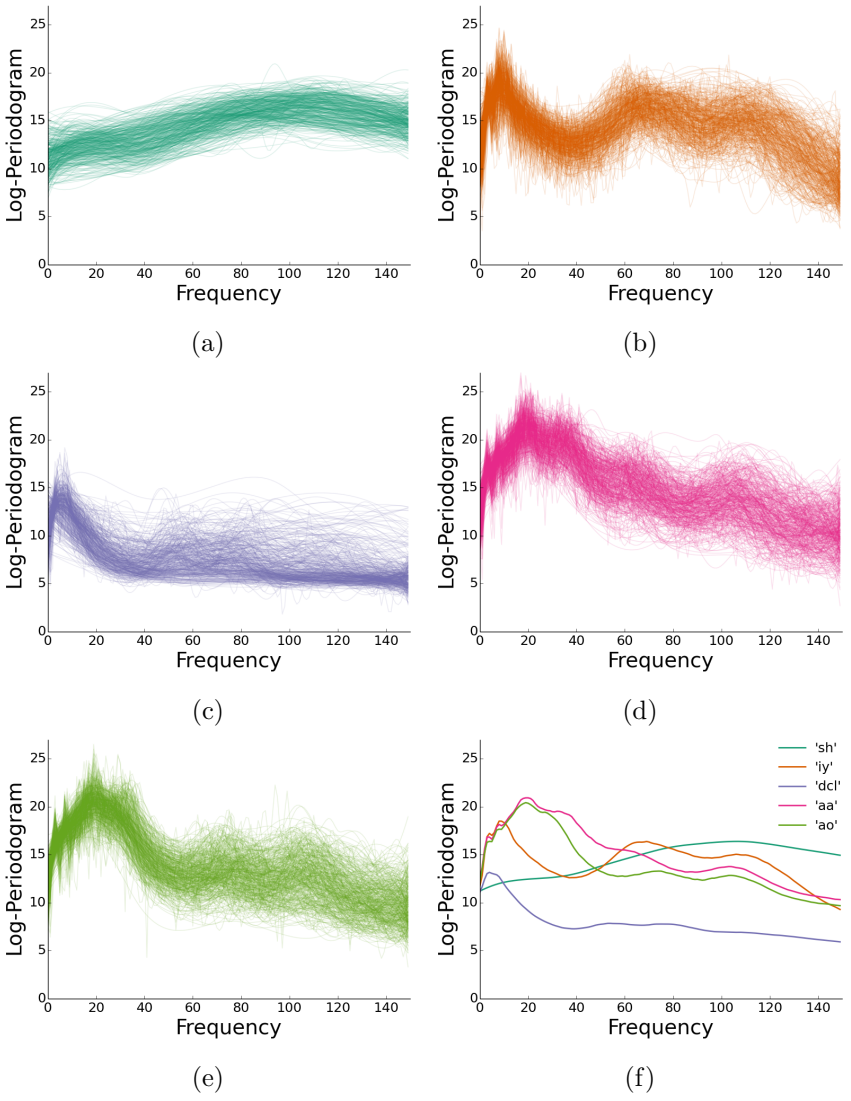


Figure 4.1: Spoken phoneme data, separated into the 5 true classes. Each phoneme is represented by the log-periodogram at 150 fixed frequencies (see Hastie et al. (1995) for details), which we smoothed with a cubic spline. The phonemes are (a) 'sh', (b) 'iy', (c) 'dcl', (d) 'aa', and (e) 'ao'. (f) The mean function for each phoneme; color indicates correspondence between phoneme class and mean function.

what contrived to use to a pseudo-density estimate instead of a *bona fide* density. Furthermore, the phoneme dataset has been curated carefully by Hastie et al. (1995) and Ferraty and Vieu (2006) for the purpose of illustrating statistical methods.

Hurricane trajectories are also easily visualized functional data, but they are

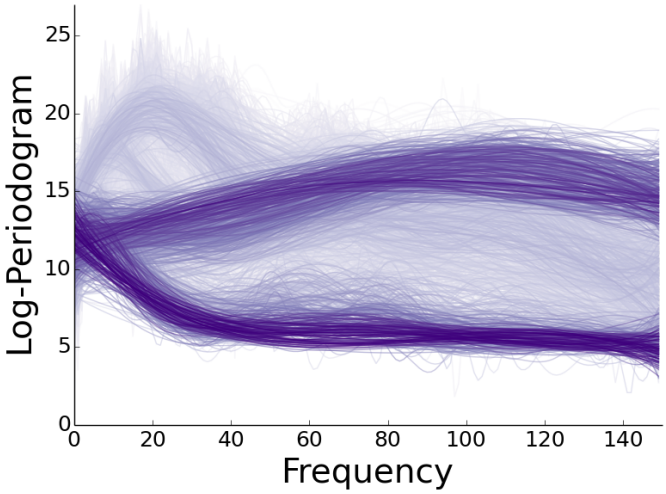
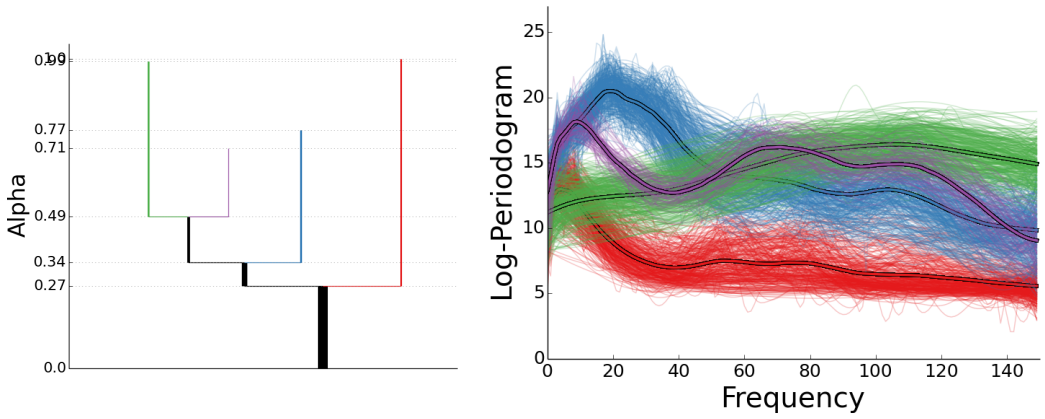


Figure 4.2: Pseudo-density values for the phoneme data. More intense shades indicate higher pseudo-density values, implying greater proximity to neighbors.



(a) (b)

Figure 4.3: (a) Phoneme level set tree. (b) All-mode clusters. The modal observation in each all-mode cluster is shown with a black outline.

sampled irregularly (in space), have variable lengths, and lie in an ambient dimension larger than $d = 1$, making them much more like white matter fiber streamlines. We show that level set trees can be used effectively to describe the topography of a hurricane track dataset and to gather tracks into coherent clusters, opening a new

Table 4.1: Confusion matrix for level set tree all-mode clustering of phonemes. Phoneme data, by true group. Observations in the "background" column do not belong to any of the high-density clusters.

	Cluster				
phoneme	1	2	3	4	background
'sh'	0	0	391	0	9
'iy'	0	0	0	85	315
'dcl'	343	0	0	0	57
'aa'	0	183	0	0	217
'ao'	0	230	0	0	170

avenue for improvements in track prediction.

The U.S. National Hurricane Center's HURDAT2 dataset contains positional and atmospheric measurements of North Atlantic tropical cyclones from 1851 to 2012 (Landsea et al., 2013). The coordinates (in degrees latitude and longitude) for each storm are recorded at least every six hours. The quality of tropical cyclone has improved over time (Landsea et al., 2013), so we selected storms that occurred relatively recently (between 1950 and 2012) to maintain a high degree of data integrity. We removed storms with fewer than 10 observations and those that never achieved hurricane status of sustained 65 knot winds. The processed dataset contained 398 hurricane tracks, chosen independently of the difficulty of clustering them.

Pairwise distances between hurricane trajectories were measured with the *max-average-min distance* D_{mam}, which is a fiber streamline distance described in detail in Section 4.4. It is not a metric, but it does not require tracks to have the same number of samples and it has produced good results in fiber streamline datasets. A complication with hurricane trajectories is that the distance between a pair of points on two tracks was measured by a distance that takes into account the oblate spheroid shape of the Earth. Because true storm tracks have infinite dimension, and are recorded on varying numbers of observations, we used the k-nearest neighbor

pseudo-density estimate, with $k = 6$ and $\gamma = 2$. These parameters were chosen by trial-and-error with a qualitative evaluation of the information content of the result; higher values of k result in overly simple level set trees while lower values lead to fragmented results with many very small clusters and little hierarchical structure. The data are shown in Figure 4.4, colored by pseudo-density (darker hues correspond to high pseudo-density values). Just by looking at this map, we expected at least one major mode centered near the Yucatán peninsula and one just offshore of the U.S east coast.

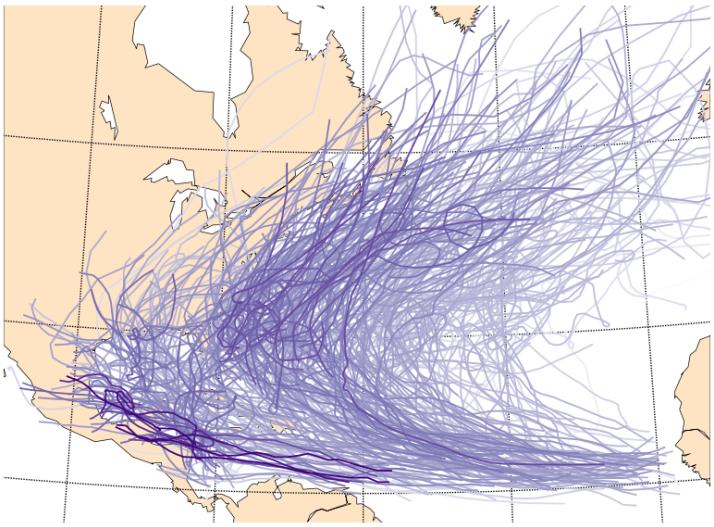
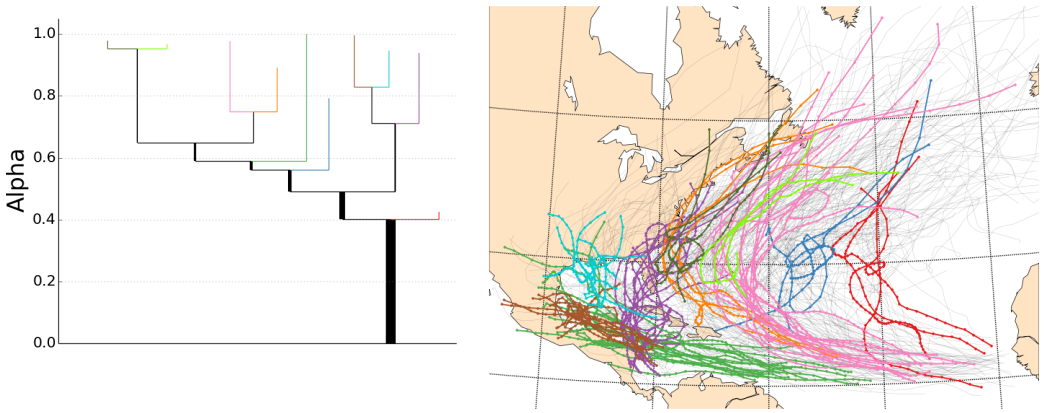


Figure 4.4: Hurricane track data, shaded by pseudo-density value. Curves with more intense color have higher pseudo-density, indicating greater proximity to neighbors.

Figure 4.5 shows the estimated level set tree and corresponding all-mode clusters. By visual inspection, the tree is very successful at separating clusters with either different shape trajectories or spatial separation. The former case, the cyan, brown, and purple clusters are all very near each other in the Gulf of Mexico, but are distinct clusters because they follow different tracks. On the other hand, the pink and orange pseudo-density clusters follow similar trajectories but are distinct because they are separated in space. The level set tree also correctly captures the hierarchy

of similarity; that is, the Gulf of Mexico storms are more similar to each other than to any other cluster. The only cluster that seems possibly out of place is the plain green, which is closer to the mid-Atlantic pink and orange than to the Gulf of Mexico storms, despite substantial overlap with the latter.



(a) (b)

Figure 4.5: (a) Hurricane track level set tree. (b) All-mode clusters. Color indicates correspondence between dendrogram branches and clusters on the map; low-density, unclustered background tracks are shown faintly in gray.

4.4 Fiber streamlines

4.4.1 Fiber streamline distances

Returning now to the original challenge of building level set trees for fiber streamlines, a major decision in analyzing this type of data (or any functional data) is the choice of a distance function between two observations. One of the most popular choices in fiber tractography analysis is D_{mam} (also known as *chamfer*) distance (Moberts et al., 2005; O'Donnell et al., 2013), which first matches each point on a streamline to the closest point on the opposite streamline, then takes the average of those matched pair distances. Let X and Y be fiber tracks with m_X and

m_Y points respectively. The MAM distance is

$$D_{mam}(X,Y) = \max\left\{ \frac{1}{m_X}\sum_{i=1}^{m_X} \min_j \|X_i - Y_j\|, \quad \frac{1}{m_Y}\sum_{j=1}^{m_Y} \min_i \|X_i - Y_j\| \right\} \tag{4.2}$$

where $\|X_i - Y_j\|$ is the Euclidean distance between two points in $\mathbb{R}^3$. Although it works well in practice, D_{mam} does not satisfy the triangle inequality, as shown with a counterexample in Figure 4.6. This complicates its use in pseudo-density level set trees in two ways: from a theoretical perspective, it is problematic to use a "similarity radius" for the computation of the pseudo-density estimate, rather than a true distance; and on the computational side, the violation of the triangle inequality prevents the use of ball trees to quickly find the k-nearest neighbors of each streamline.

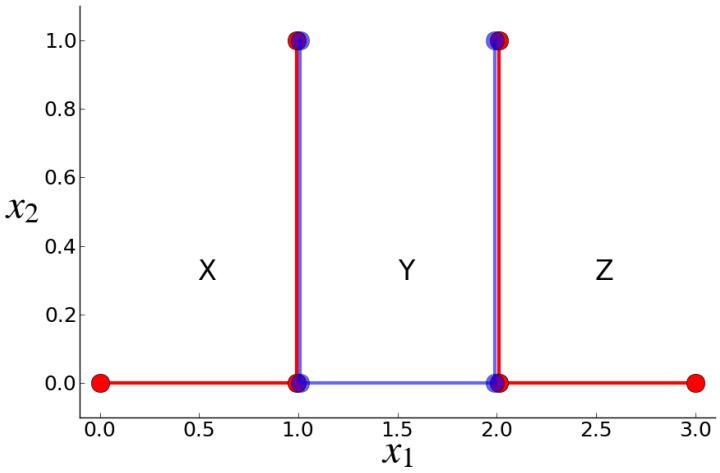


Figure 4.6: A counterexample to the triangle inequality for the max-average-min fiber distance. Assume the points of fibers X and Z at $x_1 = 1$ and $x_1 = 2$ are co-located with the points of Y, although for illustration they are shown with a gap. The non-symmetric distance from X to Y is $d(X,Y) = \frac{1}{m}\sum_{i=1}^{m} \min_j \|X_i - Y_j\|$. In this case $d(X,Y) = (1/3)(0+0+1) = 1/3$ and $d(Y,X) = (1/4)(0+0+1+1) = 1/2$. The MAM distance is the larger of these two, or $D(X,Y) = 1/2$. By symmetry $D(Y,Z) = 1/2$ as well. The distance between the outside fibers is $D(X,Z) = (1/3)(1+1+2) = 4/3$. In sum, $D(X,Y) + D(Y,Z) = 1/2 + 1/2 = 1 < 4/3 = D(X,Z)$, so the triangle inequality is violated.

The *minimum average direct-flip distance* D_{mdf} is also intuitive, fast to compute, and satisfies the triangle inequality (Garyfallidis et al., 2012). Assuming fibers X and Y are both sampled at m points, the MDF distance is

$$D_{mdf}(X,Y) = \min \left\{ \frac{1}{m} \sum_{i=1}^{m} \|X_i - Y_i\|, \quad \frac{1}{m} \sum_{i=1}^{m} \|X_i - Y_{m-i}\| \right\}. \tag{4.3}$$

The Python DiPy package contains code for this function as well as the necessary preprocessing step of equalizing the number of points in each fiber through linear interpolation. For robustness, we typically set m to be the 95th percentile of the points in all original streamlines (Garyfallidis et al., 2011).

4.4.2 Single ROI streamline results

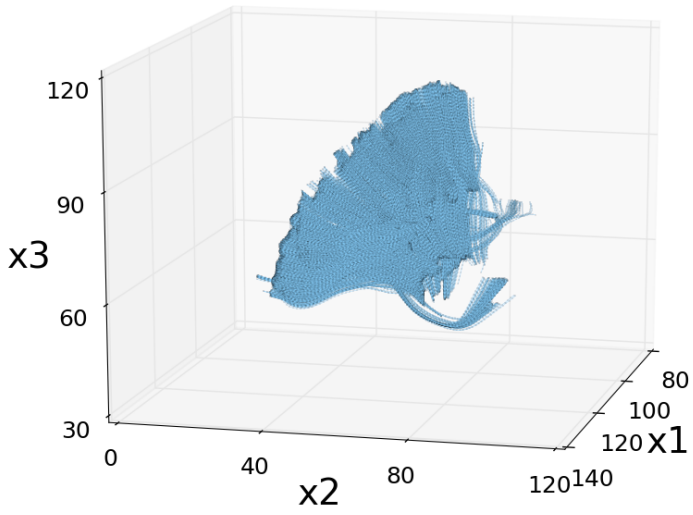
Our first analysis focuses on relatively small sets of fiber streamlines that project from two specific regions-of-interest (ROIs) into the striatum. The *lateral frontal cortex* set consists of 51,126 total streamlines (Figure 4.7a), while the *orbitofrontal cortex* set includes 3,038 streamlines (Figure 4.8a). Unlike the fiber endpoint dataset, which were simulated from DWI data of a single subject, the streamlines in this section are simulated from *CMU-30 template* DWI data, which averages the precursor data across scans of 30 participants. Further detail about the DWI parameters and construction of the CMU-30 template can be found in Kent et al.(2013) and at `http://www.psy.cmu.edu/~coaxlab/?page_id=305`. We focus on these datasets first because the corticostriatal fibers that terminate on the striatal nuclei are known to be topographically organized based on cortical origin of the fibers (Verstynen et al., 2012a; Draganski et al., 2008), with regions of high-density projections (also known as "focal" projection fields (Haber et al., 2006)), making them ideal for exploring local density structure in white matter pathways.

To estimate level set tree for each ROI-based streamline set, we computed D_{mam} between all pairs of observed curves. As discussed in Section 5.3, the max-average-

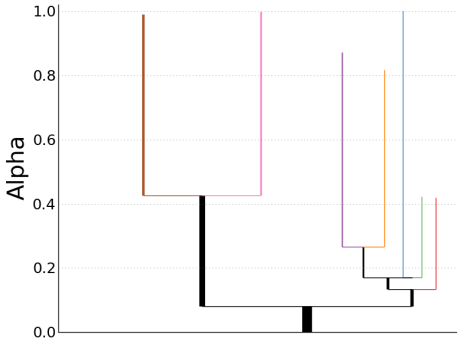
min distance is popular in the fiber tractography community and was shown to work well in a comparison of fiber streamline segmentation methods (Moberts et al., 2005; O'Donnell et al., 2013). D_{mam} is not a metric, but for samples with fewer than roughly 50,000 observations it was not a computational burden to compute all pairs of distances and the ball tree shortcut (which relies on the triangle inequality) was not needed.

In the lateral frontal cortex we detected seven clusters of streamlines (containing 34,982 foreground fibers) that were organized in a consistent, evenly spaced rostral-caudal direction along the middle frontal gyrus (Figure 4.7c), an organization that is consistent with previous reports in both the animal and human literatures (Dragan-ski et al., 2008; Haber and Knutson, 2010; Verstynen et al., 2012a). Each identified cluster reflects regions of high pseudo-density along the middle frontal gyrus. It is important to note that this whole-fiber clustering was able to capture divergent pat-terns in the white matter pathways. The blue and orange streamlines start in the same region of the middle frontal gyrus, but diverge to different sub-cortical targets (namely, the caudate and putamen). This split is easy to identify in the level set tree by the emergence of an early branching in the tree into two major divisions that reflect caudate versus putamen fibers (Figure 4.7b). This provides a clean anatomi-cal segmentation of the fibers despite the fact that these two fiber sets start in the same region of the middle frontal gyrus.

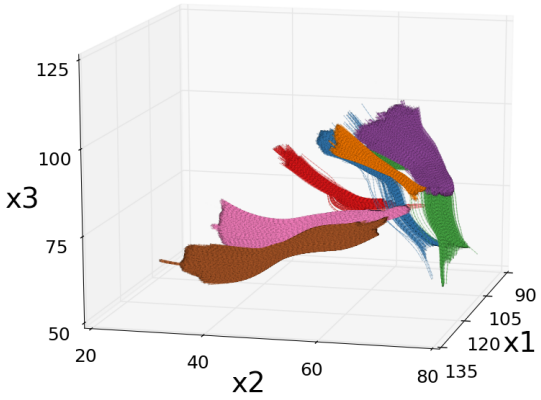
In the projections from the orbitofrontal cortex we identified five mode clus-ters (Figure 4.8c). Close inspection of the striatal endpoints of these streamlines reveals that each cluster forms a striated-like pattern in the caudate that is simi-lar to patterns previously reported in corticostriatal projections (Verstynen et al., 2012a). These striated formations are thought to reflect the modularized biochem-ical makeup of the striatum (Graybiel and Ragsdale, 1978; Ragsdale and Graybiel, 1990). This complex arrangement is difficult to capture with clustering methods



(a)



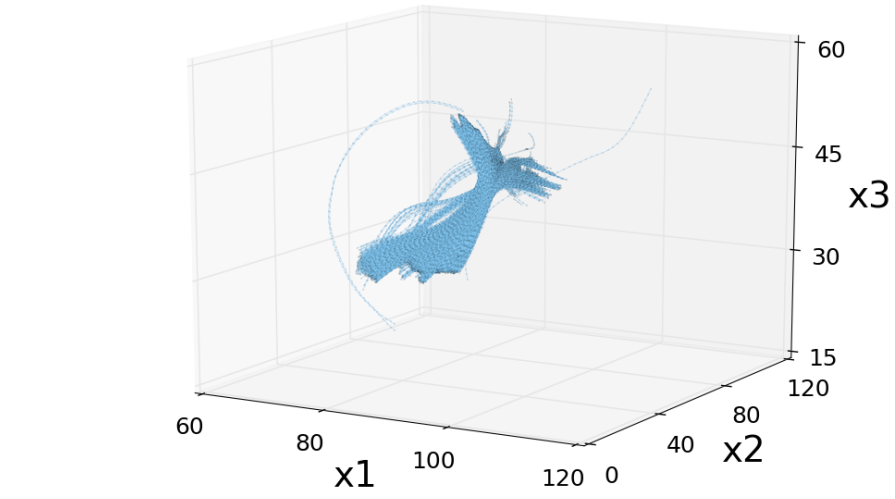
(b)



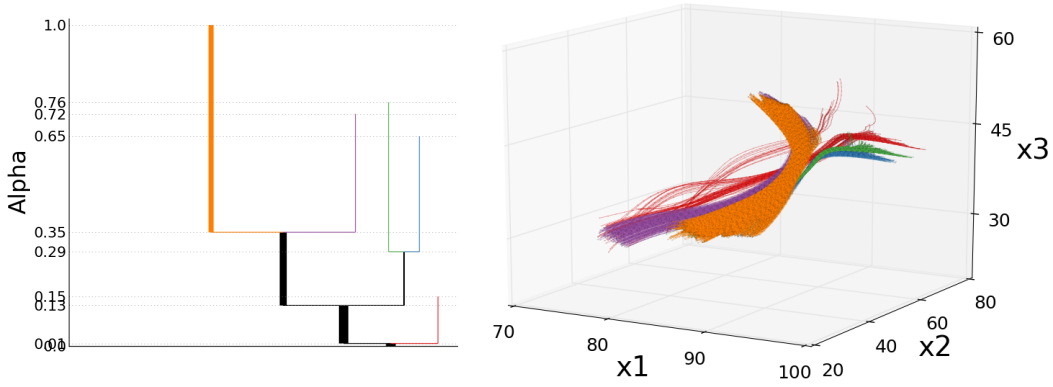
(c)

Figure 4.7: (a) 51,126 fiber streamlines projecting from the lateral frontal cortex to striatum mapped from the CMU-30 template DWI imagery. (b) Dendrogram for the α level set tree constructed from the pseudo-densities of lateral frontal cortex streamlines. (c) All-mode clusters from the lateral frontal cortex level set tree, colored to match the nodes of the dendrogram. There are 34,982 streamlines in the set of all-mode clusters.

that assume convex cluster shapes, but the pseudo-density/level set tree approach successfully extracts the patterns with minimal assumptions.



(a)



(b) (c)

Figure 4.8: (a) 3,038 fiber streamlines projecting from the orbitofrontal cortex to the striatum were mapped from CMU-30 template DWI imagery. (b) Dendrogram from the α level set tree constructed from the pseudo-densities of orbitofrontal cortex streamlines. (c) All-mode clusters from the orbitofrontal cortex level set tree, colored to match the nodes of the dendrogram.

4.4.3 Whole brain fiber streamlines

Although the level set tree results are good for ROI-specific fiber streamlines, it is common practice in deterministic fiber tractography is to simulate streamlines that cover the whole brain, usually with several hundred thousand fibers (Hagmann

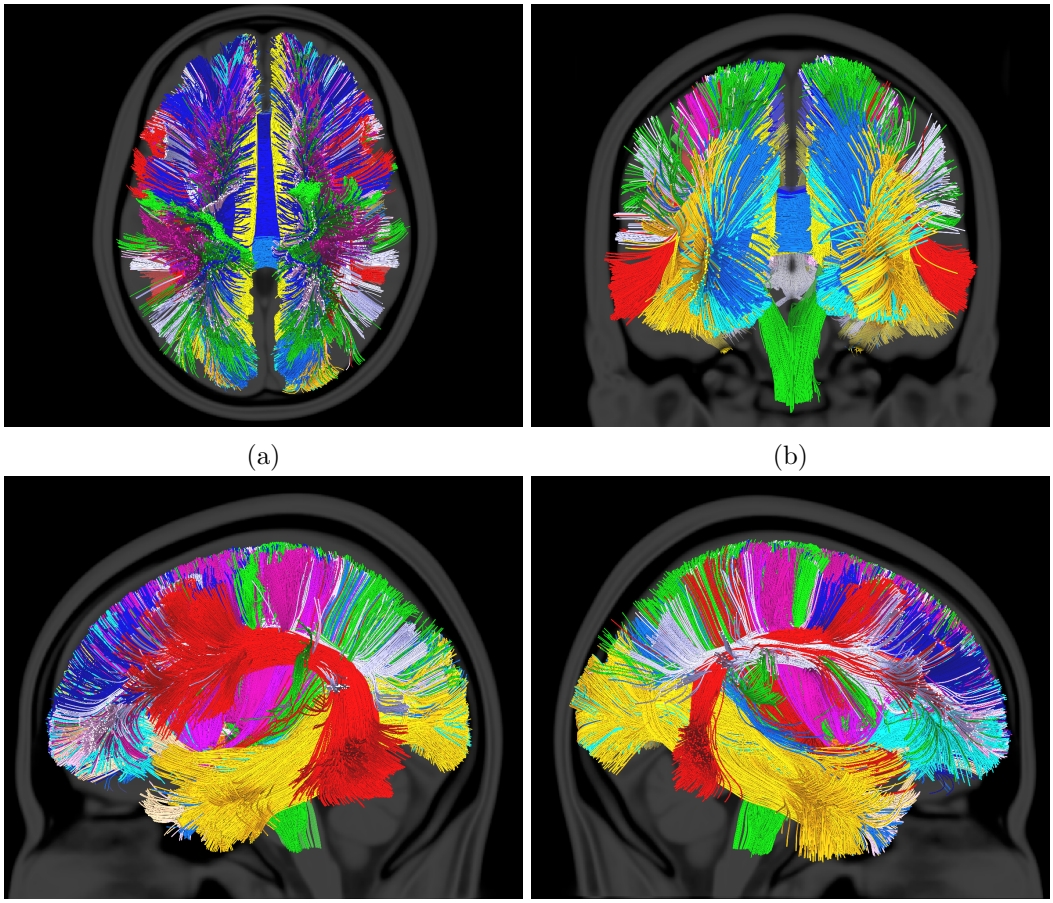
et al., 2006). The ability of pseudo-densities and level set trees to represent clustering behavior over multiple data "resolutions" suggests our method should transfer well from the smaller single ROI data to large whole brain problems.

For this section we use the CMU-60 dataset, publicly available (along with subject and acquisition details) at http://www.psy.cmu.edu/~coaxlab/?page_id=423. This set contains DWI imagery for 60 neurologically healthy volunteers (a superset of the CMU-30 group), including 24 lean (BMI < 25), 24 overweight (BMI $\in [25, 30)$) and 10 obese (BMI ≥ 30) subjects (the BMI category of 2 subjects is unknown), as well as the CMU-60 template that is the average imagery over all 60 participants. Our *whole-brain* dataset consists of 100,000 fiber streamlines simulated from the CMU-60 template.

Whole brain datasets pose several challenges on top of those discussed already in this section and in previous chapters. With the ROI-specific streamline sets, we could compute all pairwise distances in a reasonable about of time (on the order of several hours) and keep the results (about 20 GB) in memory on a modern desktop computer, but with 100,000 streamlines this is no longer possible. To overcome this we use the ball tree method for finding k-nearest neighbors, but this technique requires a distance that is a true metric. Because D_{mam} does not satisfy the triangle inequality it is disqualified and we resort to D_{mdf} instead (discussed earlier in this section).

For whole brain data, validation of the level set tree topography and clustering is especially difficult. Not only is there a lot of noise in the imagery and the streamline simulation process, but the *in vivo* nature of the data means it is impossible to assign a true group to each streamline. To evaluate the quality of level set trees for modeling whole brain fiber streamlines we enlisted a neuroanatomical expert with three years of clinical tractography experience to construct a *validation set* of streamlines. Like the *whole-brain* data, this set was simulated from the CMU-60

template imagery, but under the constraint that each streamline belongs to one of 30 known white matter tracts. The 82,934 curves in this dataset are shown in 4.9, and the tracts are listed in Figure 4.13.



(a) (b)

(c) (d)

Figure 4.9: Fiber streamlines in the whole-brain *validation set*, constructed from the CMU-60 template DWI data by a neuroanatomical and fiber tractography expert to include only streamlines that belong to anatomically valid white matter fiber tracts. Tracts are indicated by color. The views are (a) axial, (b) coronal, (c) left, and (d) right.

Through qualitative comparisons of the level set trees and all-mode clustering on the we established the optimal tree parameters for the validation set to be $k = 300$ and $\gamma = 150$. To save computation time we use a regular grid of 4,000 α values, rather than estimating connectivity for the entire set $\left\{\frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}, 1\right\}$. The dendrogram

for the α tree is shown in Figure 4.10 and the all-mode clusters are shown in brain space in Figure 4.11, although overplotting effects make it difficult to evaluate the quality of the result.

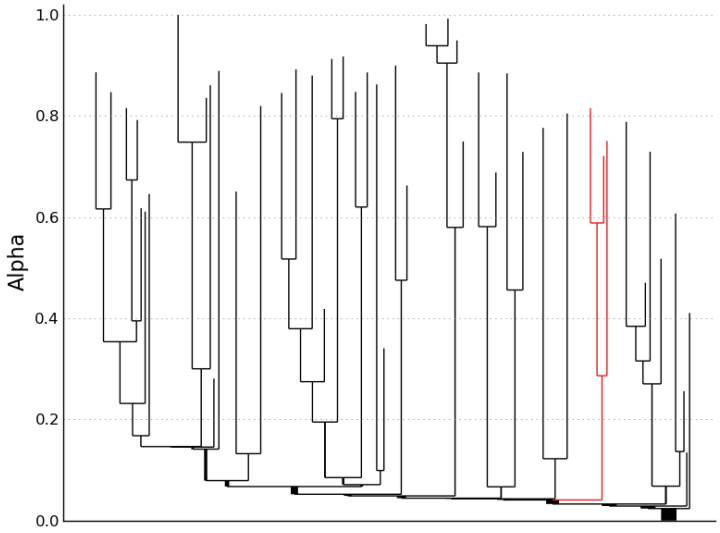


Figure 4.10: Dendrogram for the α tree constructed from pseudo-densities for the fiber streamlines in the whole-brain validation set, estimated with $k = 300$ and $\gamma = 150$. The subtree highlighted in red is explored further in Figure 4.12.

The GUI tool described in Section 3.1 helps us to evaluate the level set tree result visually and intuitively, by identifying the cluster tree node associated with a given branch of the dendrogram. Figure 4.12 illustrates a traversal through the subtree associated with the left arcuate fasciculus (AF_L) and superior longitudinal fasciculus (SLF_L).

In addition to qualitative evaluation with the GUI tool, several quantitative measures suggest the level set tree successfully captures the topography of the validation dataset. The all-mode labeling has 47 high-density clusters that contain 56.3% of the data and these clusters match the true groups well as measured by either adjusted rand index (ARI) (Hubert and Arabie, 1985), adjusted mutual information index (AMI) (Vinh et al., 2010), or classification accuracy. The latter metric is the

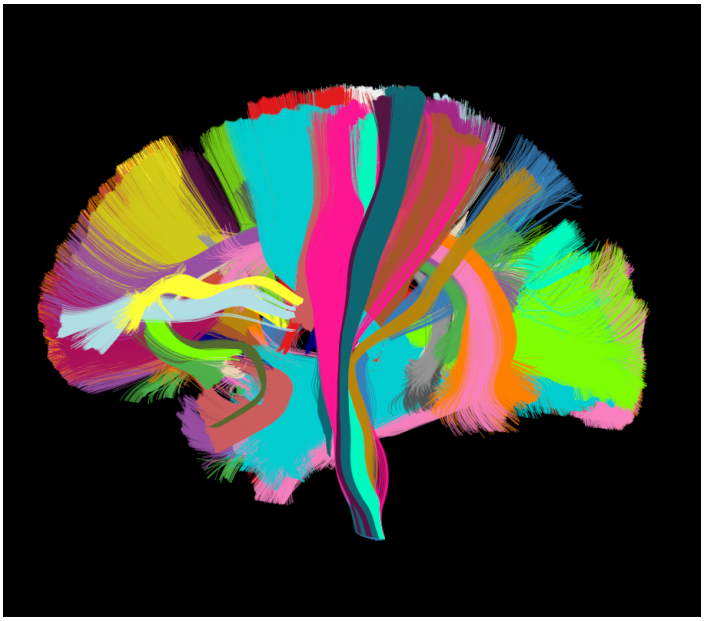


Figure 4.11: All-mode clusters for the validation fiber streamline set. These clusters contain 56.3% of the 82,934 streamlines in the whole validation set.

fraction of points with the same cluster and group label, under the optimal permutation of cluster labels, which is computed with the Hungarian algorithm (Kuhn, 1955; Munkres, 1957).

As a quick and straightforward comparison, the QuickBundles algorithm yields lower accuracy on all three measures, although it must be noted that it leaves far fewer streamlines unlabeled and runs faster (Table 4.2). The classification accuracy of two methods is shown in heatmap form in Figure 4.13; after alignment of cluster and group labels with the Hungarian algorithm, the diagonal of the heatmap shows the fraction of points with matching cluster and group labels, ignoring unclustered background points. The QuickBundles result has a greater fraction of off-diagonal observations than level set tree clustering, indicating tracts that are split across multiple clusters or clusters that capture multiple true tracts. At the same time there is clearly a higher fraction of data assigned to clusters (note that both methods have 47 clusters by coincidence; the neighborhood parameter for QuickBundles was

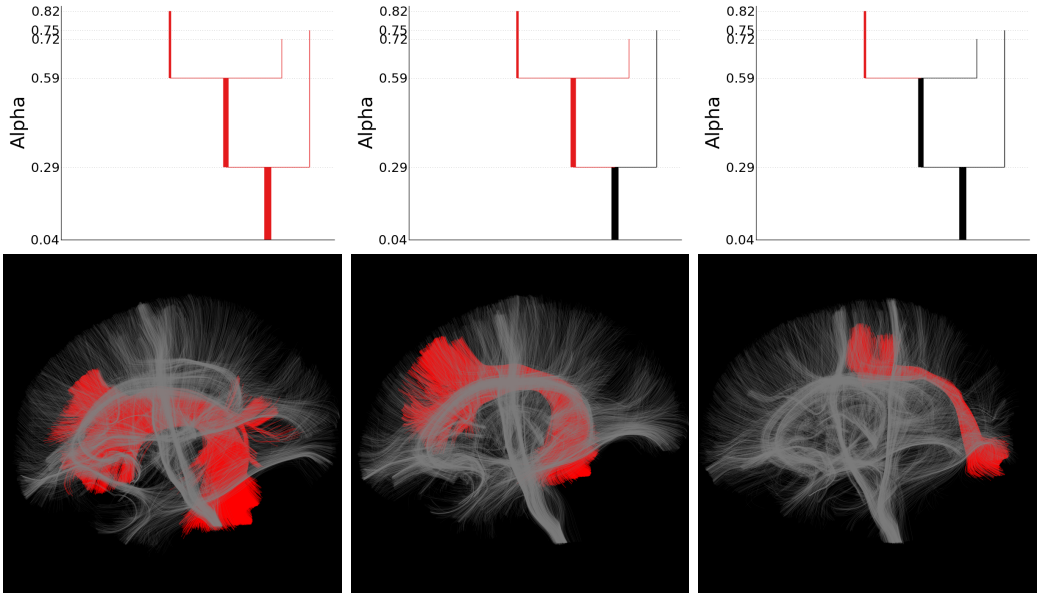


Figure 4.12: The subtree highlighted in red in Figure 4.10 is explored further. The top row shows the dendrogram of the subtree, with branches highlighted at successively higher α levels for further investigation. The streamlines associated with the selected branches are shown immediately below; this subtree corresponds to the left arcuate fasciculus and superior longitudinal fasciculus.

chosen independently to match typical usage in Garyfallidis et al. (Garyfallidis et al., 2012)).

Table 4.2: Classification accuracy for level set tree all-mode clustering and QuickBundles, in the validation set of fiber streamlines with known anatomical tract labels. ARI is the adjusted Rand index, AMI is the adjusted mutual information index, and accuracy is classification accuracy after optimal alignment of cluster labels with the Hungarian algorithm. The fraction of streamlines not assigned to a cluster is reported in the final column.

	Metric			
Method	ARI	AMI	Accuracy	Unclassified
LST all-mode	.823	.863	.775	.437
QuickBundles	.664	.804	.704	.003

The whole-brain streamline set contains 100,000 whole-brain streamlines simulated from the template ODF map without exogenous information about member-

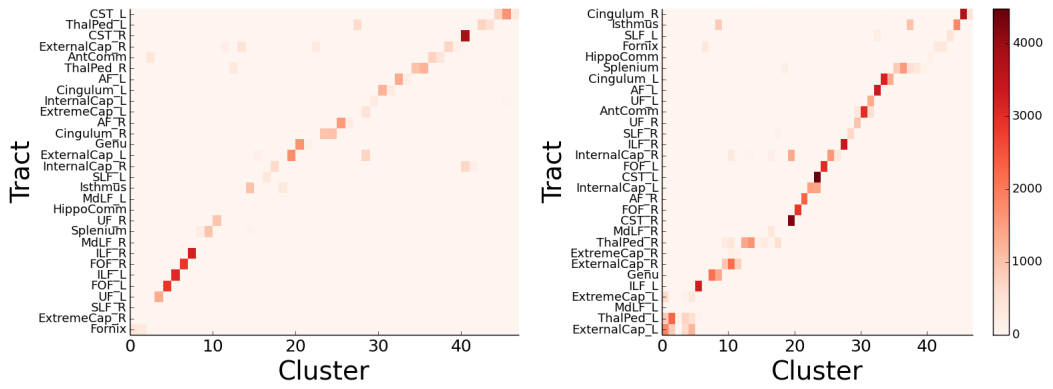


Figure 4.13: (a) Confusion matrix for level set tree all-mode clusters vs. true fiber tracts in the validation streamline set, represented as a heatmap. The cluster labels were permuted to be optimally aligned with the tracts using the Hungarian algorithm after discarding low-density unclustered points. More intense shades indicate larger entries in the matrix. (b) Confusion matrix for the QuickBundles result with the validation set of streamlines, using the same process and normalization as in panel a.

ship in "true" white matter tracts. The level set tree for this dataset was constructed with proportionally the same smoothing and pruning parameters as the validation results ($k = 361$, $\gamma = 180$). The dendrogram for the α tree is shown in Figure 4.14 and the all-mode clusters in Figure 4.15.

The level set tree results with this dataset are considerably noisier; according to a neuroanatomy expert, most clusters do not correspond to true white matter fiber tracts. On the other hand, the clusters do appear to be coherent and reasonable collections of fiber streamlines, and there are some notable matches between clusters and true fiber tracts. One of these—involving the left arcuate fasciculus (AF_L) and left inferior longitudinal fasciculus (ILF_L) is illustrated with the branch selection GUI tool in Figure 4.16. Note that although the subtrees for these tracts are slightly different in the whole-brain and validation dendrograms, the location of the subtree is roughly similar in each tree. In general, the clusters of the whole-brain tree show

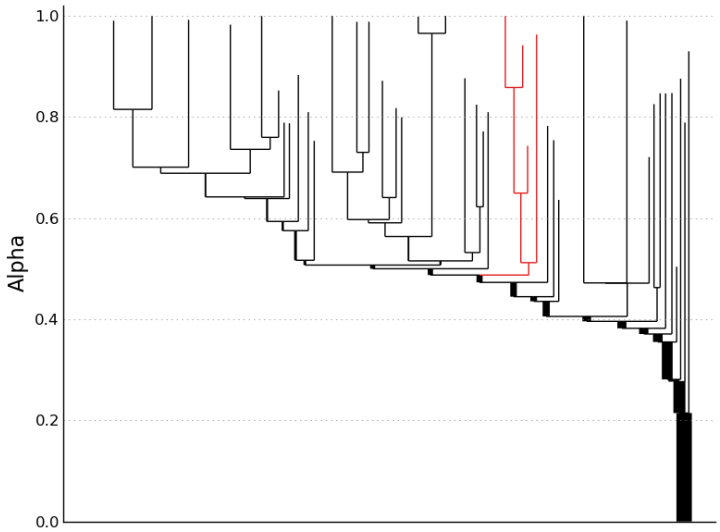


Figure 4.14: Dendrogram for the α tree constructed from pseudo-densities for fiber streamlines in the whole-brain set, estimated with $k = 381$ and $\gamma = 180$. The subtree highlighted in red is explored further in Figure 4.16.

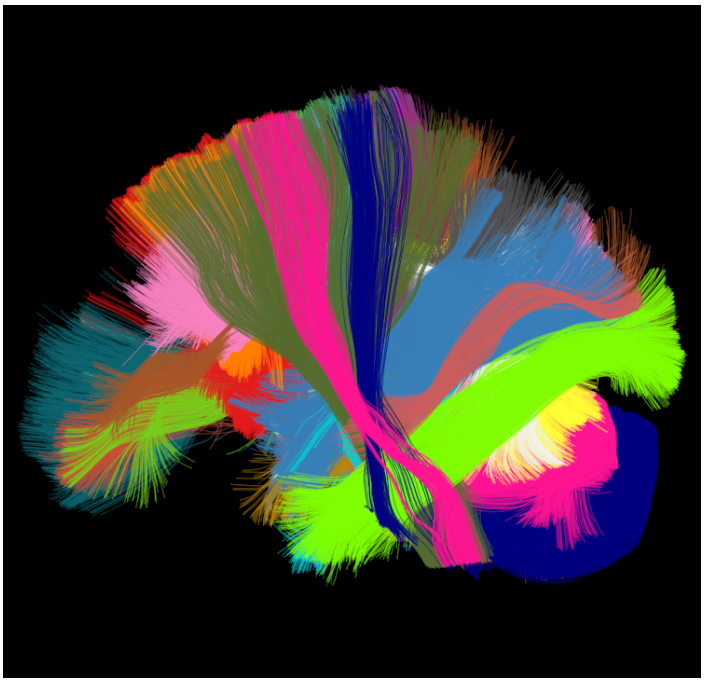


Figure 4.15: All-mode foreground clusters for the whole-brain fiber streamline set.

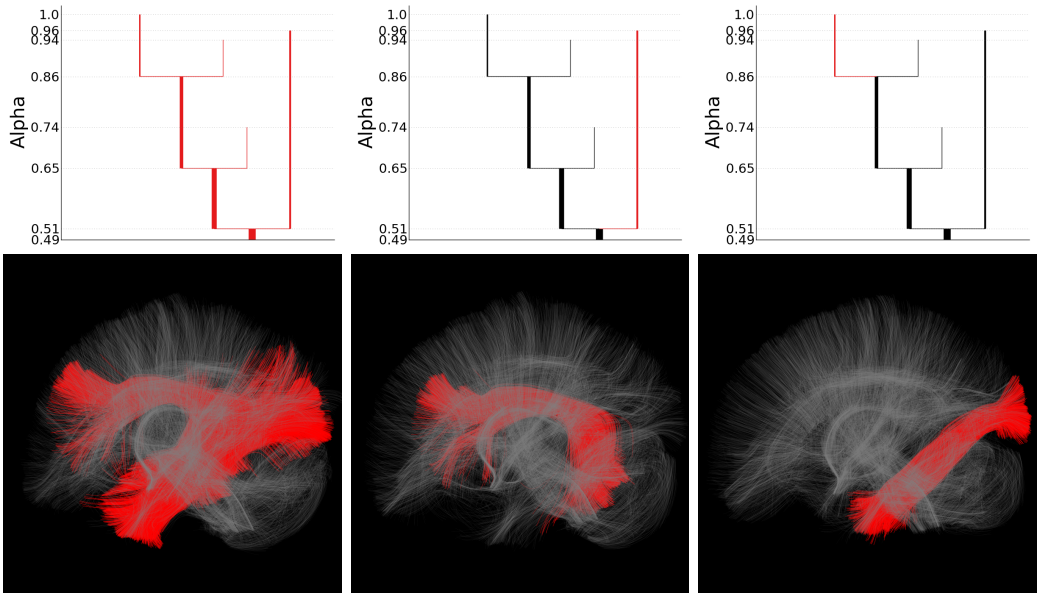


Figure 4.16: The subtree highlighted in Figure 4.14 is explored further. The top row shows the dendrogram of the selected subtree, with branches highlighted at successively higher α levels for further investigation. The streamlines associated with each branch are shown immediately below; this subtree includes fibers from both the left arcuate fasciculus and left inferior longitudinal fasciculus, which are separated at the leaf nodes.

substantially less persistence than those in the validation tree, most likely due to the larger amount of global noise in the whole-brain data. This idea of comparing two (or more) level set trees, and level set tree consistency more generally, is considered in Chapter 5.

Chapter 5

Inference

A critical statistical task for any of the analyses in previous chapters is the description of how accurately a sample level set tree estimates the true tree. Because level set trees are complicated objects, a direct numerical result remains elusive, but we propose several qualitative methods to assess the tree stability. Our ultimate goal is to treat level set trees as statistics in their own right, and to use them for inference about underlying distributions.

The fundamental tool for assessing the stability of estimated level set trees is a bootstrap-type procedure. From the original data $\mathbb{X}_n$, we draw v samples—with or without replacement—and compute the estimated level set tree for each sample. From this *orchard*[1] Υ of bootstrapped trees we can compute a large number of qualitative and numerical measures of stability or conduct rudimentary statistical inference and prediction.

5.1 Descriptive plots

The simplest thing to do with an orchard is to plot all dendrograms on the same canvas. For a small v, this gives a surprisingly clear picture of level set tree stability,

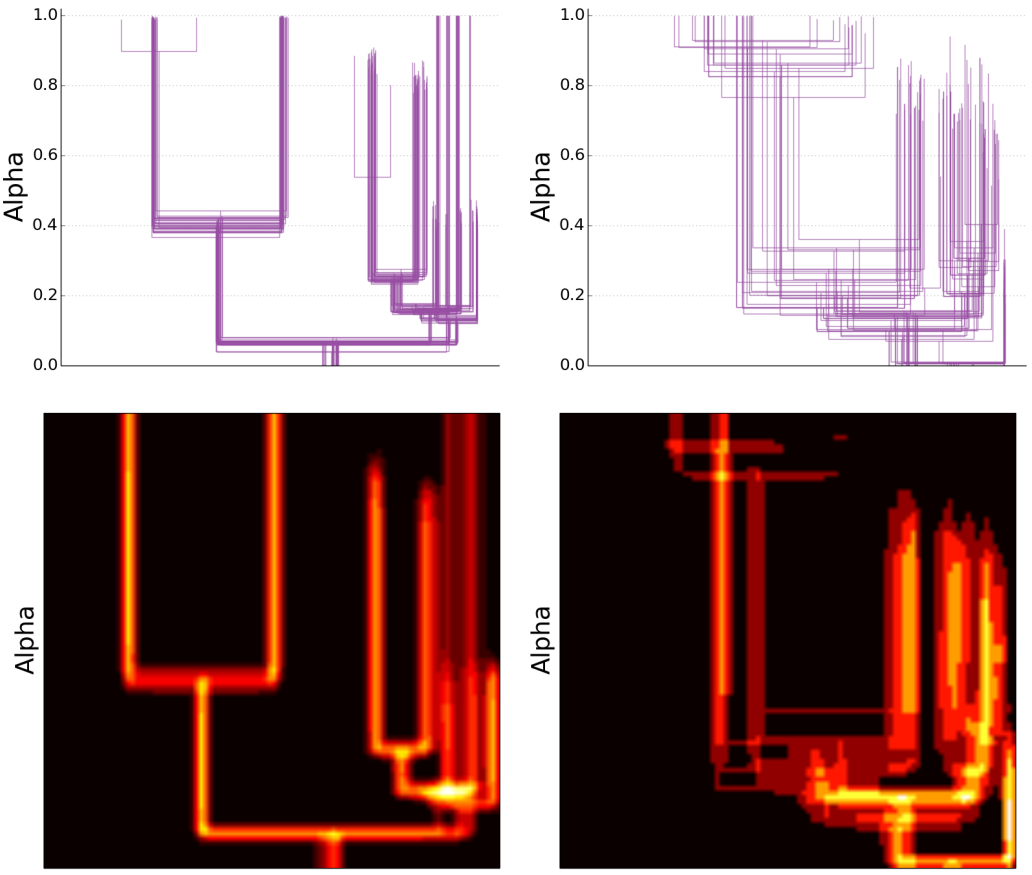
[1] Larry Wasserman deserves credit for this name.

which intuitively guides our degree of confidence in the accuracy of the original tree estimate. Figure 5.1 shows orchard plots for the level set trees on the single ROI fiber streamline datasets. For the lateral frontal cortex data we drew $\upsilon = 27$ subsamples (without replacement) from the data, each containing 15,000 streamlines (from the total 51,126), and for the orbitofrontal data we drew $\upsilon = 23$ subsamples (without replacement) of size 1,500 (from the total 3,038). These values of υ were chosen to match the number of participants whose scans contributed to the final template imagery for each ROI.

It is immediately clear that there is far less variation in the topography of the lateral frontal cortex trees, especially in the α levels at which tree branches split or vanish. While the trees of the orbitofrontal orchard clearly tend to have the same structure, the levels of the interesting features vary substantially.

For orchards with a large number of trees or large tree variation, the usefulness of the orchard plot is limited by overplotting problems. The *orchard intensity* plots of Figures 5.1a and 5.1b are a way to smooth out a noisy orchard plot into a more readable form. A 100×100 mesh is superimposed on the canvas of the orchard plot, and we count the number of line segments (vertical and horizontal) that intersect each cell of this mesh. A Gaussian filter is applied to smooth the counts across cells, and the counts are then visualized as an image. For the single ROI fiber streamline trees, the intensity plots do not add much information to the orchard plots, but they do emphasize that there are several regions of the orbitofrontal tree that actually are consistent across the bootstrap trees. In particular, the initial split occurs at almost exactly the same level in each bootstrap tree and yields two clusters of almost identical mass each time (one of which is the red cluster in Figure 4.8).

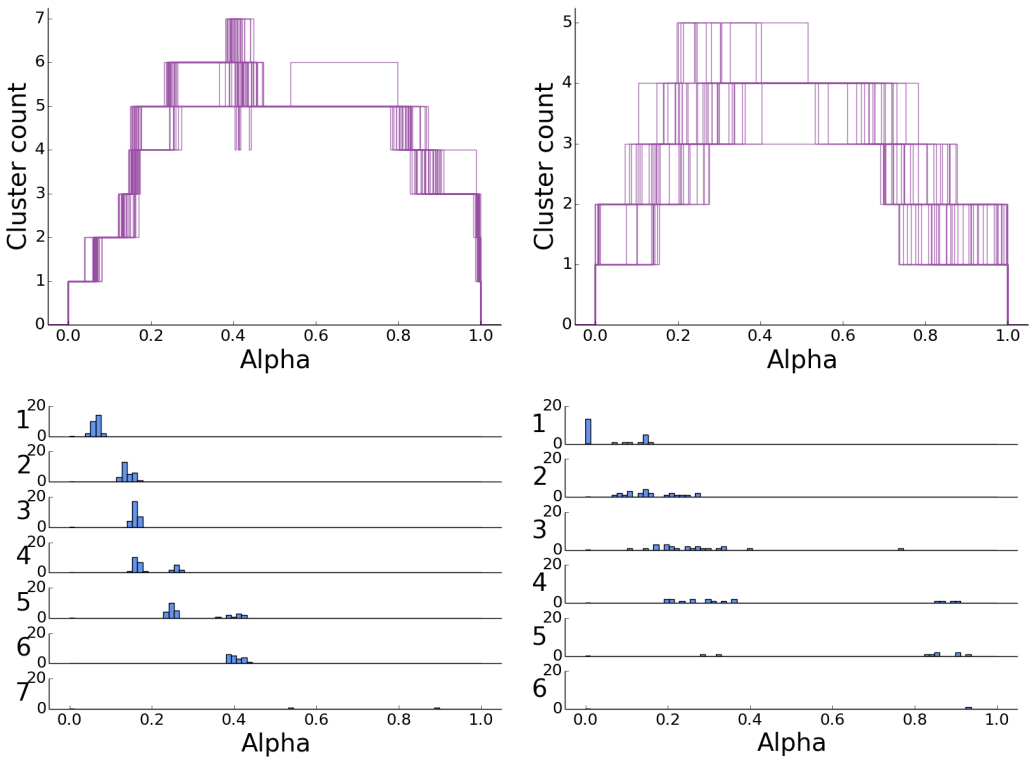
Based on the supposition that variation in the birth and death levels (λ' and λ'' or α' and α'') of tree branches is of primary interest, we also draw *mode functions* and *split histograms* for the orchards. The mode function simply counts the number of



(a) (b)

Figure 5.1: Orchard and intensity plots for subsamples from (a) lateral frontal cortex streamlines and (b) orbitofrontal cortex. The lateral frontal cortex orchard includes trees from $\upsilon = 27$ subsamples of 15,000 streamlines, and the orbitofrontal cortex orchard has $\upsilon = 23$ trees estimated for subsamples of 1,500 streamlines. The orchard plot simply plots all dendrograms in an orchard on the same canvas, while the intensity plot discretizes this canvas into a 100×100 grid, counts the number of trees that intersect each cell of the grid, and smooths the counts with a Gaussian filter.

high-density clusters as α varies from 0 to 1 (or as λ varies from 0 to $\max_x \widehat{f}$)(Azzalini and Torelli, 2007). The split histogram imposes a grid on the α scale and counts the number of splits that occur across the whole orchard in each interval of the grid. This mode functions and split histograms for the single ROI streamline orchards



(a) (b)

Figure 5.2: Mode functions and split histograms for subsamples from (a) lateral frontal cortex streamlines and (b) orbitofrontal cortex streamlines. The mode function indicates the number of clusters at each level of the α level set tree, while the split histogram indicates the location of α levels where branching occurs in all of the level set trees in an orchard.

(Figure 5.2) lead to the same conclusions as the orchard and intensity plots do: there is much more variation in the orbitofrontal cortex orchard than the lateral frontal cortex orchard, so we are more confident in the accuracy of the estimated level set tree for the lateral frontal cortex streamlines.

5.2 Test-retest comparisons

The plots in Figures 5.1 and 5.2 illustrate that randomness in a sample leads to uncertainty in the level set tree estimator. It is important to put this uncertainty

into context by comparing it to the uncertainty caused by measurement error, i.e. the uncertainty inherent in image acquisition and fiber streamline simulation. For the single ROI streamline data discussed in the previous section, we show that the measurement error in fact dwarfs the noise of the level set tree estimator, supporting the idea that level set trees are in fact a robust method.

For a subset of participants in the CMU-30 group, fiber streamlines were reconstructed for two separate scans separated by approximately six months. Figure 5.3 shows the level set trees constructed for the lateral frontal projections from each scan in several subjects of this subset, as well as the all-mode clusters. These clusters exhibit a high-degree of similarity between test and retest trials, with the consistent exception of one or two clusters that only appear in one of the two scans (highlighted in gray in Figure 5.3).

The level set trees likewise reflect similar structure across scans, but the non-overlapping tree nodes appear to exaggerate the differences between trees. For example, panels E and F in Figure 5.3 show the foreground fiber streamlines and level set trees for two scans of a single subject. The blue, green, cyan, violet, and yellow clusters match well across scans and appear to share very similar topography. However, panel E contains a cluster on the right side of the plot (in gray) that is not present in panel F, while panel F contains its own unmatched cluster on the left side of the plot (also in gray). Note that each branch's (or cluster's) color was manually defined to match in the test and retest within each subject, but not necessarily across subjects.

While some level set tree features reflect the overall similarity between test and retest streamlines—in panels E and F for example, the number of leaves is the same and the yellow, cyan, and violet clusters are more similar to each other while the blue cluster is much different—the overall shape of the trees tends to be quite different. These variations reflect actual differences between the test and retest data, not just

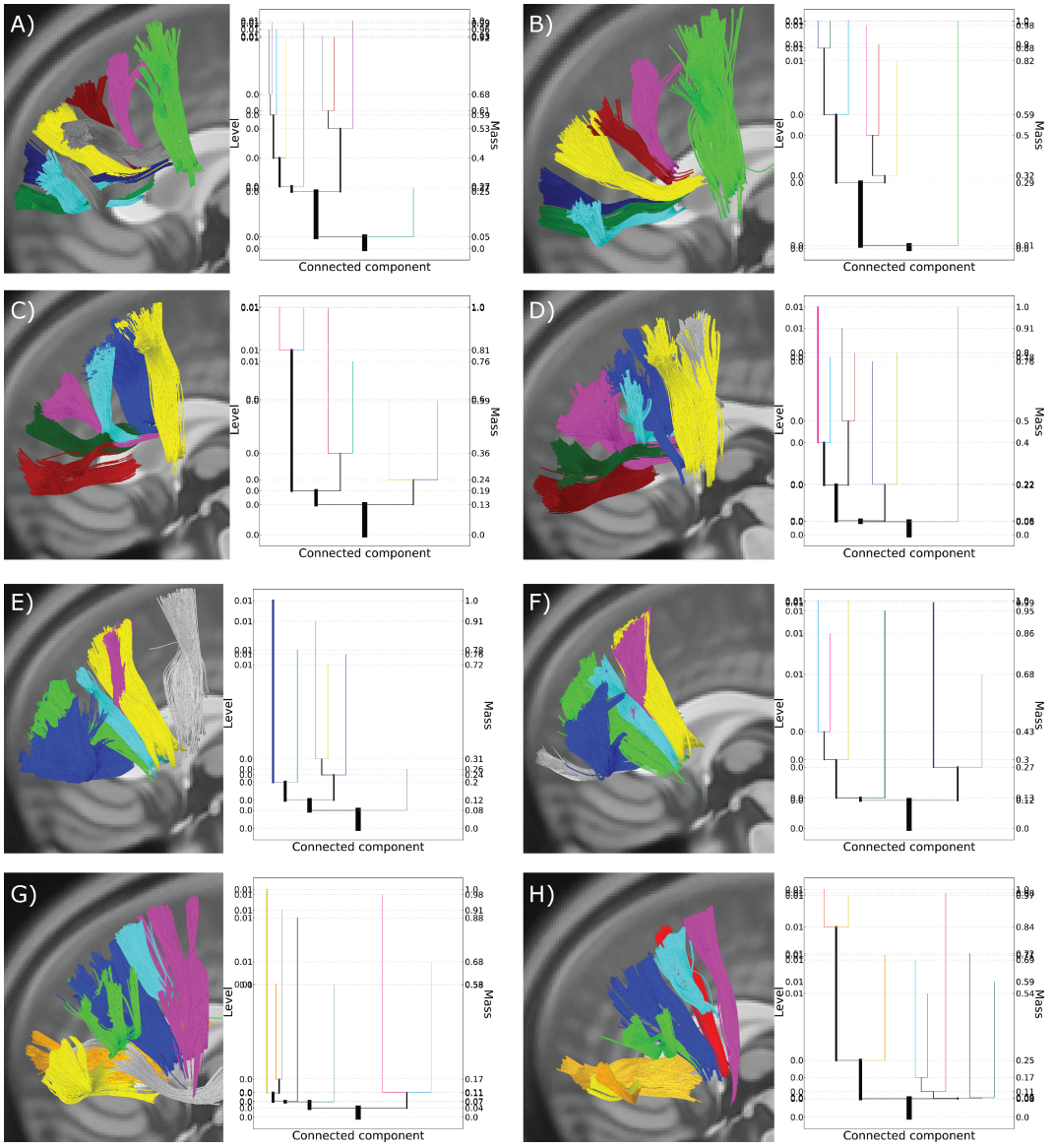


Figure 5.3: α level set tree dendrograms and all-mode clusters for the test-retest fiber streamline experiment. Each row shows the results for a single participant, with columns 1 and 2 showing the all-mode clusters and dendrogram from the first scan and columns 3 and 4 showing the same for the follow-up scan. Clusters that are matched across scans are colored the same; those that do not have a match are colored gray.

variability of the level set tree procedure. Not only are some clusters present in only one of the two data sets, but differing tree shapes and branching locations indicate

that the probability content and relative hierarchy of even similar-looking clusters is not the same across scans. Nevertheless, the all-mode clusters are actually quite consistent across scan sessions, demonstrating the robustness of the level set tree methodology for clustering and segmentation purposes.

5.3 Tree distances

Moving from qualitative descriptions of the uncertainty about individual level set trees to numerical descriptions and tree-based inference requires the definition of a distance between a pair of trees. We propose several ways to do this, which can be roughly broken down into those that use only data topography (i.e. the tree structure) and those that take into account the location of the data as well. The former category allows comparisons between very different distributions that may even lie in spaces of different dimension, which may be a benefit or a disadvantage depending on a practitioner's goal. To the best of our knowledge, all of these methods are novel and a great deal of work remains to determine the properties of each one. For our preferred method (paint mover distance), we present preliminary results suggesting how tree-based inference could work.

5.3.1 Paint mover distance

The paint mover distance D_{pmd} takes as input two cluster trees $\mathcal{R}_1$ and $\mathcal{R}_2$, which we assume are each binary with a single root. For each tree construct a dendrogram *signature* $\mathcal{S} = \{((z_0, y_0), \kappa_0'' - \kappa_0'), ..., ((z_{|\mathcal{R}|}, y_{|\mathcal{R}|}), \kappa_{|\mathcal{R}|}'' - \kappa_{|\mathcal{R}|}')\}$, where (z_i, y_i) is the position of dendrogram branch i on a plot canvas, $\kappa_i'' - \kappa_i'$ is the littoral mass associated with node i of the cluster tree, and $|\mathcal{R}|$ is the number of nodes in $\mathcal{R}$. The vertical coordinate y_i is simply the midpoint of the birth and death density or mass levels that define the tree branch; for the λ-tree this is $(1/2)(\lambda_i' + \lambda_i'')$, for the α-tree it is $(1/2)(\alpha_i' + \alpha_i'')$, and for the κ-tree it is$(1/2)(\kappa_i' + \kappa_i'')$.

The horizontal coordinate for branch i is set according to mass-based silos with boundary-positioned branches. The procedure for constructing these silos is described in detail in Section 2.5, although the binary tree assumption here eliminates the possibility that $|\texttt{kid}_i| > 2$. Once the signatures $\mathcal{S}_1$ and $\mathcal{S}_2$ are constructed, D_{pmd} is the earth mover distance (D_{emd}) between them (Rubner et al., 1998)[2]:

$$D_{pmd}(\mathcal{R}_1, \mathcal{R}_2) = D_{emd}(\mathcal{S}_1, \mathcal{S}_2) \tag{5.1}$$

The name paint mover distance is based on the fact that for κ-trees, the distance is the "work" required to erase and re-draw one dendrogram to match another.

Given a metric ground distance, D_{pmd} is also a metric. Suppose three cluster trees $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_3$ with signatures $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$. Most of the metric properties of the paint mover distance follow immediately from the metric properties of the ground distance and the earth mover distance (Rubner et al., 1998):

1. non-negativity

$$D_{pmd}(\mathcal{R}_1, \mathcal{R}_2) = D_{emd}(\mathcal{S}_1, \mathcal{S}_2) \geq 0; \tag{5.2}$$

2. symmetry

$$D_{pmd}(\mathcal{R}_1, \mathcal{R}_2) = D_{emd}(\mathcal{S}_1, \mathcal{S}_2) = D_{emd}(\mathcal{S}_2, \mathcal{S}_1) = D_{pmd}(\mathcal{R}_2, \mathcal{R}_1); \tag{5.3}$$

3. triangle inequality

$$\begin{aligned} D_{pmd}(\mathcal{R}_1, \mathcal{R}_3) + D_{pmd}(\mathcal{R}_3, \mathcal{R}_2) &= D_{emd}(\mathcal{S}_1, \mathcal{S}_3) + D_{emd}(\mathcal{S}_3, \mathcal{S}_2) \\ &\geq D_{emd}(\mathcal{S}_1, \mathcal{S}_2) \\ &= D_{pmd}(\mathcal{R}_1, \mathcal{R}_2). \end{aligned} \tag{5.4}$$

[2]for two distributions with equal mass, earth mover distance is also known as Mallow's or Wasserstein distance (Levina and Bickel, 2001)

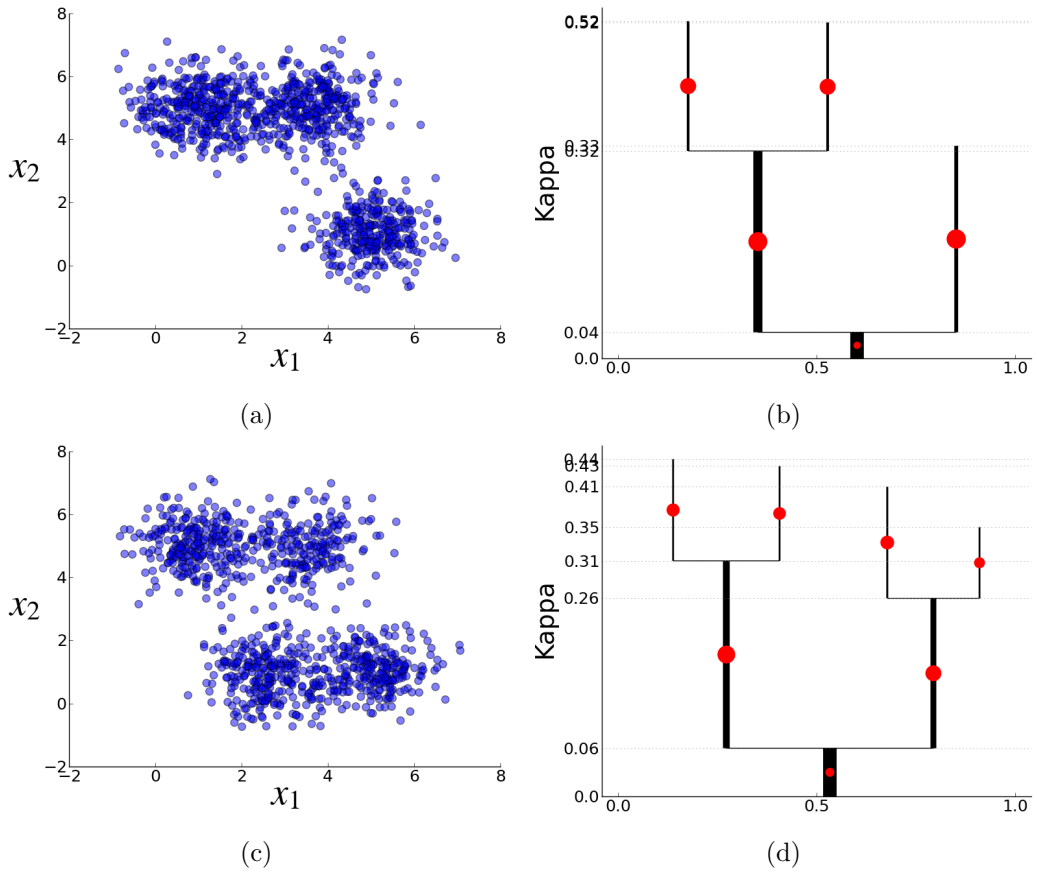


Figure 5.4: Paint mover signatures for Gaussian mixtures. (a) A sample of 1,000 points from a mixture of three Gaussians. (b) The κ dendrogram for the 3-Gaussians sample, with the signature represented by red discs. Each disc consists of coordinates in the unit square and a weight, represented by the disc's size. (c) A sample of 1,000 points drawn from a mixture of four Gaussians. (d) The paint mover signature overlaid on the κ dendrogram for the 4-Gaussian sample.

Proving the identity of indiscernibles relies on the dendrogram construction rule as well as the metric properties of the ground and earth mover distances. By construction, there are no vertical inversions in the dendrogram: if node j is a child of node i, then $y_j > y_i$. Also by construction, the whitespace silos form hard horizontal boundaries for tree families: for branch i with silo $[a_i, b_i]$, all nodes with $y_j > y_i$ and $z_j \in [a_i, b_i]$ *must* be descendants of node i.

Based on these observations, the following procedure reconstructs a *unique* den-

drogram (and by extension, a cluster tree) from a given signature $\mathcal{S}$. Consider the elements of $\mathcal{S}$ in order of increasing vertical position. For each signature element $(z_i, y_i), M_i)$, do the following three steps.

- Draw a vertical line segment from $(z_i, y_{\texttt{parent}_i})$ to $(z_i, y_{\texttt{parent}_i} + 2(y_i - y_{\texttt{parent}_i}))$.
- Identify the silo $[a_i, b_i]$. The root silo is fixed at $[0, 1]$. Since we traverse the points from the bottom-up, we know the silo of node i's parent to be $[a_{\texttt{parent}_i}, b_{\texttt{parent}_i}]$. Because the dendrogram is constructed so that each branch is positioned horizontally on the boundary of its child silos, if $z_i < z_{\texttt{parent}_i}$ then node i's silo is $[a_{\texttt{parent}_i}, z_i]$, otherwise its silo is $[z_i, b_{\texttt{parent}_i}]$.
- Identify the child nodes of i, $\texttt{kid}_i$. Find the set of signature points j such that $z_j \in [a_i, b_i]$ and $y_j > y_i$. From this set, label the lowest point on each side of node i as a member of $\texttt{kid}_i$. If the set is empty, node i is a leaf of the tree. For each node $j \in \texttt{kid}_i$, draw a horizontal line segment from $(z_i, y_{\texttt{parent}_i} + 2(y_i - y_{\texttt{parent}_i}))$ to $(z_j, y_{\texttt{parent}_i} + 2(y_i - y_{\texttt{parent}_i}))$.

The existence of this procedure proves that $\mathcal{S}_1 = \mathcal{S}_2 \implies \mathcal{R}_1 = \mathcal{R}_2$. This fact allows us to finish the proof that paint mover distance is a metric.

$$\begin{aligned} D_{pmd}(\mathcal{R}_1, \mathcal{R}_2) = 0 &\implies D_{emd}(\mathcal{S}_1, \mathcal{S}_2) = 0 \\ &\implies \mathcal{S}_1 = \mathcal{S}_2 \\ &\implies \mathcal{R}_1 = \mathcal{R}_2. \end{aligned} \tag{5.5}$$

The first statement is true by definition of paint mover distance, the second holds because earth mover distance is a metric, and the final one holds because of the existence of the tree reconstruction procedure. We typically use Manhattan distance as the ground distance to emphasize the idea that mass can only travel horizontally and vertically on a dendrogram canvas, but limited empirical experience suggests

other ground distances work equally well. The choice of dendrogram scale may be more important, and is the subject of ongoing research.

To illustrate the effectiveness of level set trees and paint mover distance as tools for inference, we show that the method can be used to distinguish samples drawn from four different Gaussian mixture distributions. For each distribution we drew 30 samples and constructed the α level set tree for each one. An example sample and the orchard plot for all samples from each distribution are shown in Figure 5.5.

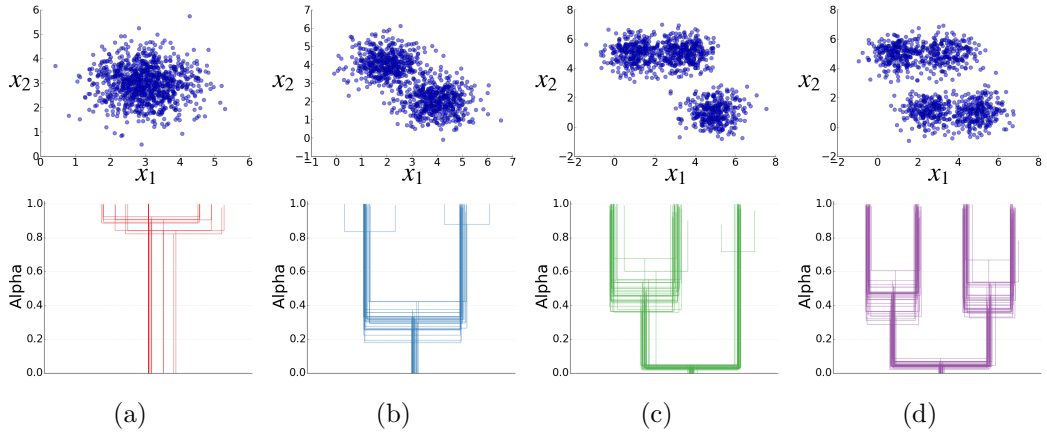


Figure 5.5: Gaussian mixture samples and level set tree orchards for paint mover distance. The top row shows an example simulation from a mixture of 1, 2, 3, or 4 Gaussian distributions. The level set trees for 30 simulated samples drawn from each mixture distribution are shown in the orchard plots in the bottom row.

The 120 total α trees were then treated as individual observations in a meta-clustering task. We computed the paint mover distance between each pair of trees and estimated the kNN pseudo-density and kNN connectivity graph with $k = 10$. Finally, we estimated the α level set tree, pruned it at $\gamma = 10$, and retrieved the all-mode clusters. These clusters perfectly assign the samples to their correct distributions (Figure 5.6 and Table 5.1), demonstrating the effectiveness of the level set tree-paint mover distance combination in discriminating between at least very simple distributions. Note also that the simpler the underlying distribution, the

less variation there is in the orchard, and the more persistent is the branch of the meta-tree.

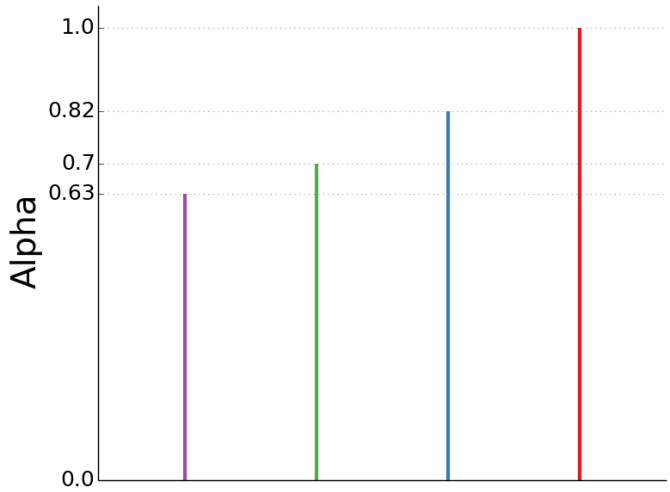


Figure 5.6: Dendrogram for the meta-tree constructed on level set trees for Gaussian mixtures. The branch colors indicate the all-mode clustering, and match the orchard colors in Figure 5.5.

Table 5.1: Confusion matrix for PMD all-mode clustering of Gaussian mixtures

	Cluster			
Distribution	0	1	2	3
1	30	0	0	0
2	0	30	0	0
3	0	0	30	0
4	0	0	0	30

We also used the paint mover distance to better quantify the variation in the level set trees for whole-brain fiber streamlines. 50,000 streamlines were simulated for each of 10 obese participants and 12 lean participants in the CMU-60 group. Based on results suggesting an association between obesity and reduced white matter integrity (Verstynen et al., 2012b, 2013), we hypothesized that each group would

have different fiber streamline organization (Bolzenius et al., 2013) and that this difference would be reflected in smaller paint mover distances between the trees in each group than between groups. For comparison, we also constructed 10 bootstrap samples of 50,000 streamlines from the whole-brain streamline data set (note: this set was simulated from template imagery that averaged the scans of all CMU-60 participants, including the 22 individuals considered here).

The orchard and intensity plots for the obese, lean, and bootstrap trees are shown in Figure 5.7. There appears to be as much tree variation within the obese and lean groups as there is between them, especially in comparison to the relatively small amount of variation in bootstrap trees. The bootstrap plots in fact show a surprisingly degree of consistency, particularly in the branches on the far left (corresponding to the right corticospinal tract and the right corona radiata) and upper right (corresponding to the right and left cerebellum).

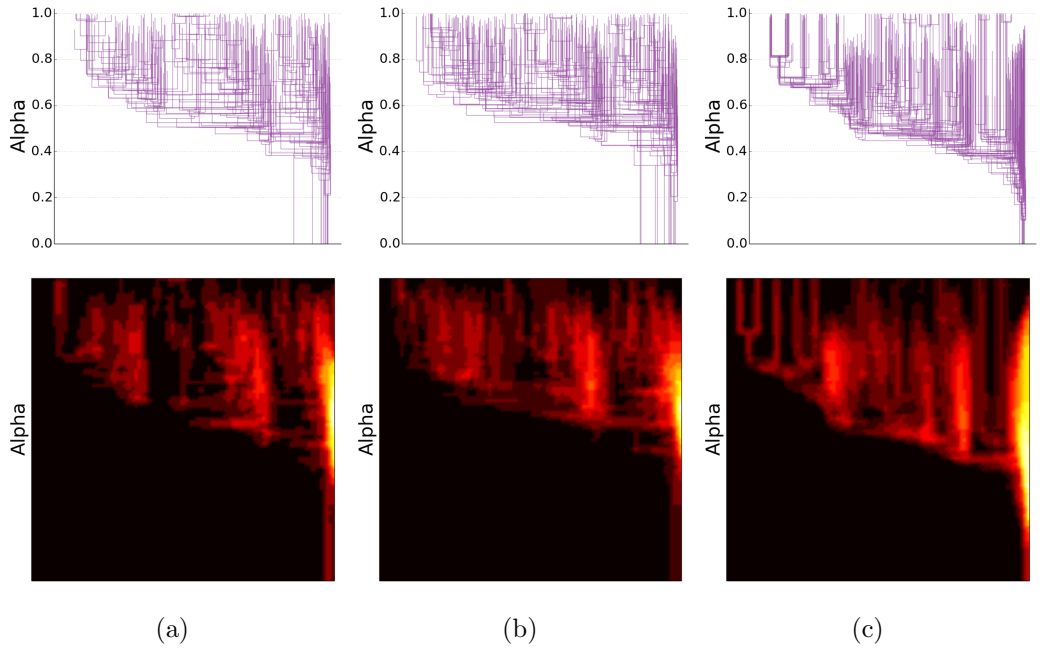


Figure 5.7: Orchard and intensity plots for (a) 10 obese, (b) 12 lean, and (c) 10 bootstrapped whole-brain fiber streamline datasets, each containing 50,000 streamlines.

The bar chart in Figure 5.8 confirms our qualitative interpretation of the orchard and intensity plots. The pairwise paint mover distances within the obese and lean groups are just as large as the paint mover distances between pairs of one obese and one lean tree. As a check, we compute the paint mover distance between each lean, obese, and bootstrap tree and the level set tree from the whole-brain fiber streamline set, and these do match the within- and between-group distances very closely. By contrast, the distance between pairs of level set trees from the Gaussian discrimination experiment when one the two trees are from different mixtures is three times as large as the obese vs. lean distances.

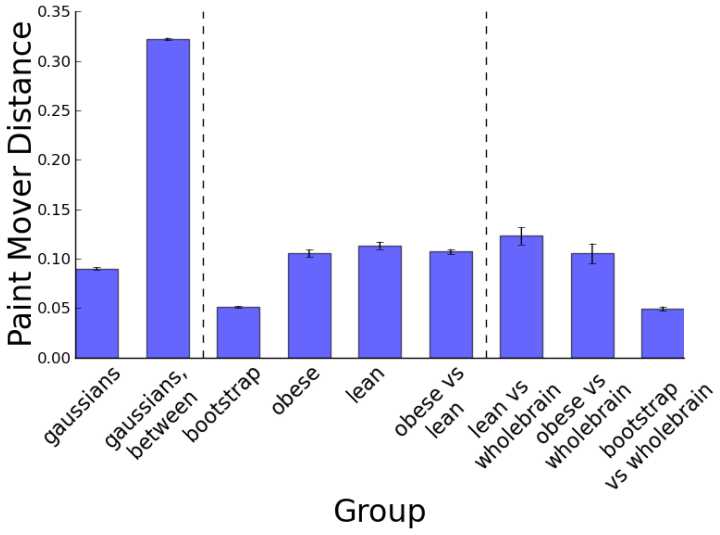


Figure 5.8: Mean and standard error of paint mover distances between and within various groups of level set trees. *gaussians* refers to the distances between level set trees constructed from the same Gaussian mixture distribution in Figure 5.5. *gaussian, between* refers to distances between level set trees from different Gaussian distributions in Figure 5.5. *bootstrap*, *obese*, and *lean* show the mean and standard error of pairwise distances between trees within the bootstrapped, obese, and lean orchards of Figure 5.7, and *obese vs. lean* indicates the average paint mover distance between all pairs of trees in the Cartesian product of the lean and obese orchards. For comparison, *lean vs. whole-brain*, *obese vs. whole-brain*, and *bootstrap vs. whole-brain* show the mean distances between the lean, obese, and bootstrapped trees and the level set tree from the whole-brain streamline set.

In sum, it appears that the paint mover distance accurately quantifies our intuitive description of the orchard and intensity plots, but that the differences between the obese and lean groups are too small—if they exist at all—to be detected by the paint mover distance or the descriptive plots. We surmise that this is due either to noise in the fiber streamline simulation process or an incorrect hypothesis—differences in white matter integrity may not affect the location of simulated streamlines—or both.

5.3.2 Other topographical distances

Paint mover distance is only one of many possible topography-based distances between level set trees. Another simple option is to compute any functional distance (e.g. ℓ_2 distance) between the mode functions for two trees, although for λ and α trees this is not a metric. Because there is not a 1-1 relationship between mode functions and level set trees, the identity of indiscernibles is violated:

$$D(\mathcal{T}_1, \mathcal{T}_2) = 0 \not\Rightarrow \mathcal{T}_1 = \mathcal{T}_2. \tag{5.6}$$

Like paint mover distance, the *canvas distance* treats the dendrogram plot canvas as the fundamental space. In this case, we compute the Euclidean distance between the intensity plots created separately for each of two trees. The properties of this distance are still unexplored and left for future research.

5.3.3 Alpha-earth mover distance

Another entire category of distances treats level set trees as simple containers for high-density clusters computed over a succession of α or λ density levels. The α-*earth mover distance* (αEMD) measures the distance between two level set trees as the distance between underlying clusters, defined by cutting the level set tree on a grid of α levels. Let α vary over a finite set of J values in $[0, 1]$. For each

value α_j, retrieve the level α_j high-density clusters and define the *signature* to be $\mathcal{S}^j = \{(x^{(1)}, \widehat{M}^{(1)}), \ldots, (x^{(K_j)}, \widehat{M}^{(K_j)})\}$, where $x^{(i)}$ is the modal observation of high-density cluster i and $\widehat{M}^{(i)}$ is the mass of the same cluster. Note that the number of clusters K_j varies for each value of α_j, but because the clusters are retrieved on α-upper level sets, the total mass is the same between each tree for each value of α_j. The signatures at three α levels for two example Gaussian mixture samples are shown in Figure 5.9.

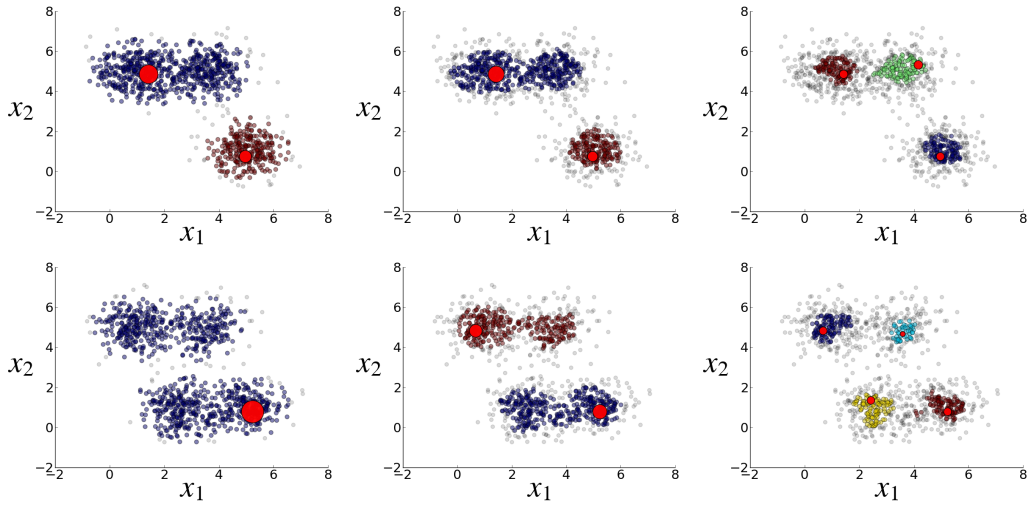


Figure 5.9: α earth mover distance signatures for 3- and 4-component Gaussian mixtures, at three example α levels. Each red disc is an element of the signature, and consists of coordinates in the feature space and a weight. The coordinates are the mode of the corresponding cluster and the weight is the mass of the cluster.

αEMD is the mean of earth mover distance over all values of α:

$$D_\alpha(\mathcal{T}_1, \mathcal{T}_2) = \frac{1}{J} \sum_{j=1}^{J} D_{emd}\left(\mathcal{S}_1^j, \mathcal{S}_2^j\right). \tag{5.7}$$

Figure 5.10 and Table 5.2 show the results of the Gaussian discrimination experiment using αEMD instead of paint mover distance. The meta-tree is clearly noisier for αEMD, and the all-mode clustering fails to distinguish the 3- and 4-component

distributions.

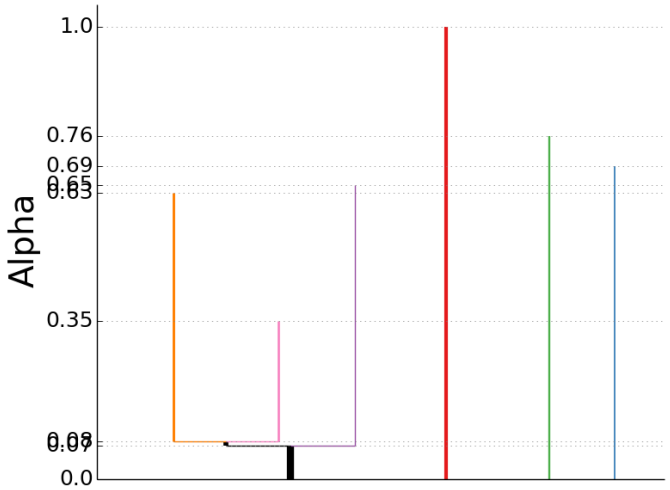


Figure 5.10: Dendrogram for the Gaussian demo α earth mover distance meta-tree, estimated with $k = 10$ and $\gamma = 5$. Color indicates branches selected as all-mode clusters.

Table 5.2: Confusion matrix for αEMD all-mode clustering of Gaussian mixtures

	Cluster						
Distribution	0	1	2	3	4	5	background
1	30	0	0	0	0	0	0
2	0	13	17	0	0	0	0
3	0	0	0	9	12	8	1
4	0	0	0	1	9	11	9

Chapter 6

The Chaudhuri-Dasgupta algorithm

The DeBaCl package also includes the first implementation (that we are aware of) of the Chaudhuri-Dasgupta level set tree algorithm (Chaudhuri and Dasgupta, 2010). As described in Chapter 1, this algorithm is particularly notable because the authors prove finite-sample convergence rates (where consistency is in the sense of Hartigan (1981)). The algorithm is a generalization of single linkage hierarchical agglomerative clustering (and Wishart (1969)), reproduced here for convenience in Algorithm 1.

Algorithm 1: Chaudhuri and Dasgupta (2010) level set tree estimation procedure.

Input: $\{x_1, \ldots, x_n\}$, k, β

Output: $\widehat{\mathcal{T}}$, a hierarchy of subsets of $\{x_1, \ldots, x_n\}$

$r_k(x_i) \leftarrow$ distance to the k'th neighbor of x_i;

for $r \leftarrow 0$ **to** ∞ **do**

  $G_r \leftarrow$ graph with vertices $\{x_i : r_k(x_i) \leq r\}$ and edges $\{(x_i, x_j) : \|x_i - x_j\| \leq \beta r\}$;

  Find the connected components of G_r;

$\widehat{\mathcal{T}} \leftarrow$ dendrogram of connected components of graphs G_r, ordered by inclusions;

To translate this program exactly into a practical implementation, we must find a finite set of values for r such that the graph G_r can only change at these values. When $\beta = 1$, the only values of r where the graph can change are the edge lengths in the graph $e_{ij} = \|x_i - x_j\|$ for all i and j. Let r take on each value of e_{ij} in descending order; in each iteration remove vertices and edges with larger k-neighbor radius and edge length, respectively.

When $\beta \neq 1$, the situation is trickier. First, note that including r values where the graph *does not* change is not a problem, since the original formulation of the method includes all values of $r \in \mathbb{R}^{+0}$. Clearly, the vertex set can still change at any edge length e_{ij}. The edge set can only change at values where $r = e_{ij}/\beta$ for some i, j. Suppose $e_{u,v}$ and $e_{r,s}$ are consecutive values in a descending ordered list of edge lengths. Let $r = e/\beta$, where $e_{u,v} < e < e_{r,s}$. Then the edge set $E = \{(x_i, x_j) : \|x_i - x_j\| \leq \beta r = e\}$ does not change as r decreases until $r = e_{u,v}/\beta$, where the threshold of βr now excludes edge (x_u, x_v). Thus, by letting r iterate over the values in $\bigcup_{i,j}\{e_{ij}, \frac{e_{ij}}{\beta}\}$, we capture all possible changes in G_r.

In practice, starting with a complete graph and removing one edge at a time is extremely slow because this requires $2 * \binom{n}{2}$ iterations of the connected component search. The DeBaCl implementation includes an option to initialize the algorithm at the kNN graph instead, which is a substantially faster approximation to the Chaudhuri-Dasgupta method. This shortcut is still dramatically slower than DeBaCl's geometric tree algorithm, which is one reason why we prefer the latter. Illustrations of the exact and approximate Chaudhuri-Dasgupta trees, along with the DeBaCl geometric tree (the algorithm used in this thesis) are shown in Figure 6.1.

Note the long tree trunk that contains no information and hides the interesting features at the top of the tree. The kNN approximation shortens the trunk, which allows the leaves to seen more easily. Note also that the radius scale on the

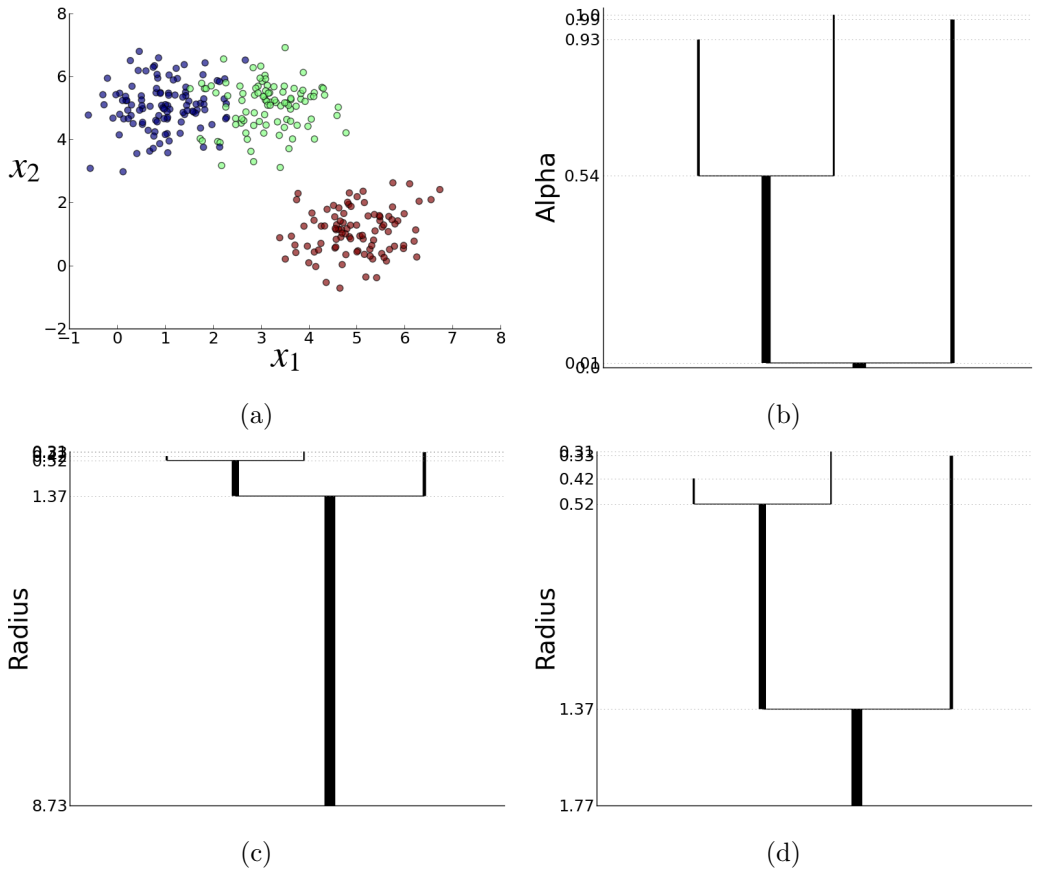


Figure 6.1: (a) A sample of 300 points in $\mathbb{R}^2$ from a mixture of 3 Gaussians, colored according to group membership. (b) The α level set tree, estimated with our standard algorithm ($k = 15$). (c) The exact Chaudhuri-Dasgupta level set tree ($\beta = 1$). (d) The kNN approximation to the Chaudhuri-Dasgupta level set tree, which removes successively shorter edges starting from the kNN similarity graph, rather than the complete graph.

Chaudhuri-Dasgupta trees is very difficult to interpret, especially in comparison to the α and κ scales of the geometric tree. Lastly, even though this demonstration sample has only $n = 300$ points, the Chaudhuri-Dasgupta tree takes over three minutes to compute (on a standard desktop PC, using just one thread). The kNN approximation reduces this to 70 seconds, but our preferred geometric tree algorithm takes only 0.05 seconds.

The generalized single linkage method proposed by Stuetzle and Nugent (2010)

is also an edge-removal method that requires $\mathcal{O}(n^2)$ connectivity estimates, but the authors show that finding the connected components of the minimal spanning tree of the connectivity graph is an equivalent and much faster shortcut. It is worth noting that this shortcut does not work for the Chaudhuri-Dasgupta algorithm.

Let T be the minimal spanning tree of similarity graph G. For generalized single linkage, the edges of G have weights that are based on estimated densities, so G_λ and T_λ are the subgraphs of G such that all vertices *and* edges have density level greater than λ. Stuetzle and Nugent show that for generalized single linkage any two nodes v_i and v_j are are in the same component of G_λ if and only if they are in the same component of T_λ. Implicit in the proof is the (correct) assumption that the vertex sets of G_λ and T_λ are identical, but for the Chaudhuri-Dasgupta algorithm this is not the case. Consequently, v_i and v_j can be in different components in T_λ, but in the same component in G_λ. Figure 6.2 shows an example where this situation occurs, but it remains to be seen how much this type of problem degrades the accuracy of the algorithm. It may be the case that minimal spanning tree's substantially better speed justifies the loss in accuracy.

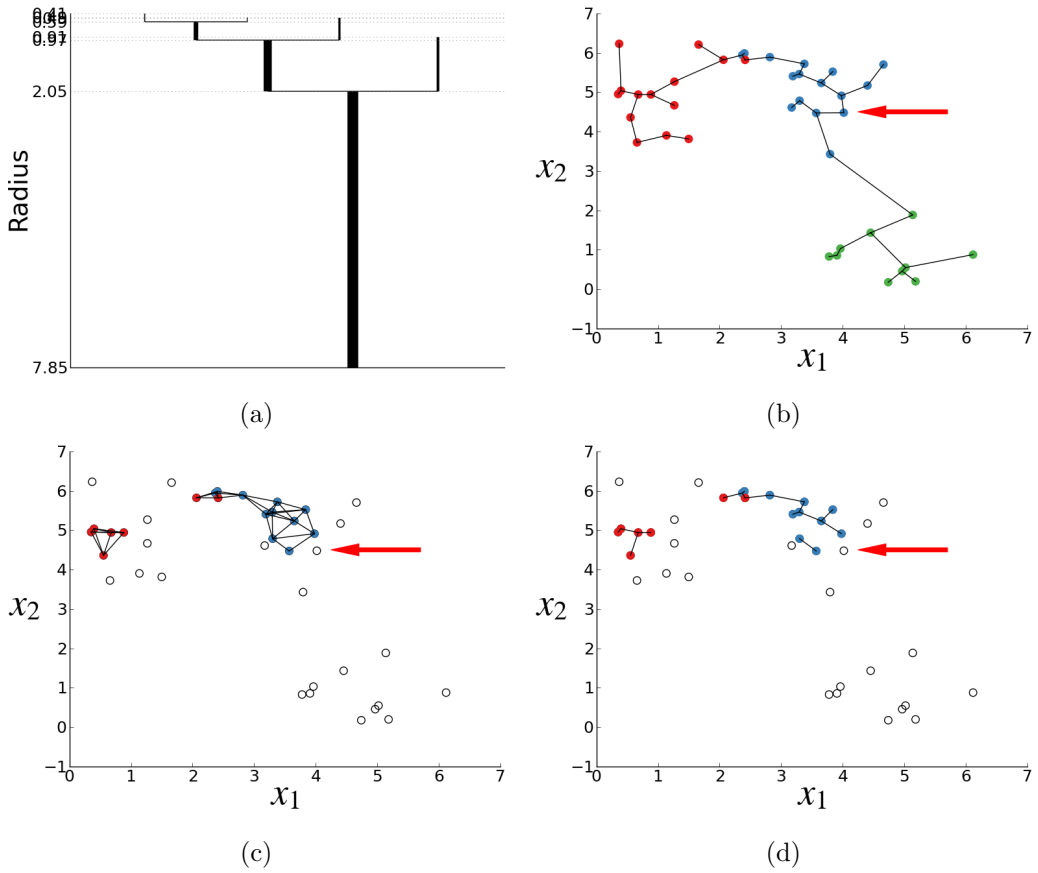


Figure 6.2: Minimal spanning tree counterexample for Chaudhuri-Dasgupta tree. (a) The Chaudhuri-Dasgupta tree for a mixture of 3 Gaussians in $\mathbb{R}^2$, with $n = 40$ and $\beta = 1$. (b) The data, colored by component membership, with the minimal spanning tree overlaid. (c) The complete subgraph G_r at radius threshold $r = 0.7$, showing two connected components. (d) The minimal spanning tree subgraph $T_{0.7}$, showing three connected components.

Chapter 7

Conclusion

The level set tree idea is nearly four decades old, but is only now coming into its own as a statistical tool. Many results have been published in the intervening years about clustering at fixed density levels, but the first strong theoretical results about the accuracy of estimators for the entire level set tree appeared only in the last few years, in Chaudhuri and Dasgupta (2010), Kpotufe and Luxburg (2011), and Rinaldo et al. (2012). We have extended the theoretical literature to make the level set tree approach more useful as a tool for applied statistics, by summarizing existing estimation techniques, proposing several methodological extensions to the Kpotufe-von Luxburg algorithm, describing ideas for inference with level set trees, and providing high-quality open source level set tree software.

Many of the practical innovations in this thesis remain unsupported by statistical theory, leaving a great number of open questions for future research. We separate these into three (overlapping) categories: computational efficiency, statistical performance of level set tree estimators, and tree-based inference.

7.1 Computational efficiency

A critical factor in the usefulness of a data analysis method is computational speed and memory efficiency, so it is important to find the fastest possible algorithm for level set tree estimation. Because our standard algorithm is a point iteration method, it is considerably faster than several existing edge iteration and histogram-based implementations, and is able to estimate trees with several tens of thousands of data points in seconds or minutes.

As described in Section 2.4.2, Najman and Couprie propose a tree estimation procedure based on the union-find algorithm that is considerably faster than using a depth- or breadth- first search to estimated connected components (Najman and Couprie, 2006). It remains to transform this procedure from an image processing routine into a more general statistical tool. While the Najman-Couprie method is extremely fast at $\mathcal{O}(n\alpha(n))$, there are shortcuts that may speed up tree estimation even further with an acceptable loss of accuracy. For example, even when estimating connectivity with a depth-first search, the procedure can be very fast if several graph vertices are removed in each iteration (rather than one). Combining this shortcut with the Najman-Couprie method would create an extremely fast but approximate tree estimator; this could be useful, for example, in choosing the smoothing parameter k by estimating the tree for many different values of k.

Another way that some tree estimators improve speed is to reduce the similarity graph G to its minimal spanning tree before iterating through subgraphs (see Stuetzle and Nugent (2010), for example). We showed in Chapter 6 that this shortcut does not produce the correct tree for the Chaudhuri-Dasgupta edge iteration algorithm, but Figure 6.1 also suggests that the minimal spanning tree-based level set tree is a good approximation to the Chaudhuri-Dasgupta tree. An interesting challenge is to quantify how close the approximation is and to devise a algorithm that approximates the Chaudhuri-Dasgupta level set tree to an arbitrary degree.

Another computational improvement would be to translate our `DeBaCl` software suite into a compiled language like C or C++. It is currently written in Python, which maximizes readability and logical clarity, but is not optimized for speed and memory efficiency.

7.2 Statistical performance

In this thesis we introduced several extensions to the Kpotufe-von Luxburg level set tree estimator, and we illustrated the value of these extensions primarily by applying them to simulations and real clustering challenges. A great deal of work remains to quantify the statistical properties of these methods. For example, the consistency results of Chaudhuri and Dasgupta (2010) and Kpotufe and Luxburg (2011) are based on the λ tree. Because the map $\alpha \mapsto \lambda_\alpha$ is one-to-one and monotonic, it seems highly likely that the α tree is also consistent (although the details may prove tricky). The κ tree, on the other hand, admits *inversions*: given two nodes i and j in the cluster tree, it is possible for $\kappa_i' > \kappa_j'$ even though $\lambda_i' < \lambda_j'$. As a result, a pressing and challenging question is whether we can find a consistency result and rates of convergence for the κ tree.

In the same vein, the existing theory is based on the existence of a density function. Using a pseudo-density estimator to build level set trees for non-Euclidean data works well because accurate tree estimation requires only the correct order relationships between observations. But without a *bona fide* pdf, it is far from obvious how to extend level set tree theory to the functional setting. It is not even clear how to define the population level set tree or a meaningful concept of consistency, because these are based on the assumption of a dominating measure for the volume of sets.

The fact that level set trees require only the correct order of data points also suggests trees can work well even with high-dimensional data where density estimation is problematic. Several existing theoretical results also suggest that level set trees

might work in the high-dimensional setting, and we demonstrated that this is indeed the case by applying the level set tree to a population genetics dataset where each individual's genetic information was measured at over 11,500 covariates. A better quantification of the relationship between the accuracy of density estimation and estimation of the order statistics of $\widehat{f} \cap \mathbb{X}_n$ is required to understand when exactly level set trees will work with high-dimensional data and how accurate they can be in that setting.

Finally, it may be possible to find answers to these questions by leveraging the close connection between level set trees and other density-based clustering methods. New developments in the mean shift algorithm in particular may yield insight into the statistical performance of level set trees and provide ideas for faster computation.

7.3 Tree inference

The idea of using level set trees for statistical inference is particularly new, and the scope for future work is enormous. In Section 5.3.1 we showed that the paint mover distance is a metric and illustrated in a very simple simulation that it can be used to discriminate between different distributions. A substantial amount of work remains to understand the statistical properties of the paint mover distance, both in theory and in practice, and the relationships between paint mover distance, canvas distance, and α-earth mover distance. One particularly thorny problem is the description of an "average" level set tree based on these distances alone.

A very promising idea in this direction borrows the concept of a persistence *landscape* from the field of persistent homology (Bubenik, 2012). The persistence landscape is a continuous piecewise linear function constructed by turning each point of a persistence diagram into a triangle whose base is on the real line, then calculating the point-wise maximum of the triangles along the real line.

We can construct a very similar function for the level set tree. Working through

nodes of the cluster tree from lowest to highest birth level (with any index), draw a line segment connecting the top of each dendrogram branch to the top of the branches on either side which have higher birth levels. If the branch already has two segments connecting it to lower branches, it is a leaf and is left alone. Draw segments connecting the top of the outermost leaves to the horizontal axis.[1]

The tree landscape defines a map from a d-dimensional density to an unnormalized 1-dimensional density that preserves the critical values of the original function. With an appropriate bootstrapping procedure we can construct an average landscape and a landscape confidence band, which can then be used both to understand the uncertainty in a single level set tree and for statistical inference (Chazal et al., 2013). we note that Oesterling et al. (2011) and Oesterling et al. (2013) propose a slightly different construction of the density landscape for data visualization and interactive exploration, but their scheme for landscape construction is somewhat convoluted and they do not pursue the idea of using the landscape for inference about underlying populations.

7.4 Final thoughts

- Choosing the connectivity and bandwidth parameter k and pruning parameter γ in level set tree estimation remains an open area of research. In practice, however, it is often surprisingly easy to find a small range for each parameter where the tree reveals interesting data structure. γ is generally chosen to fit a relevant scientific idea about the minimum plausible or useful cluster size. k is usually chosen first to be relatively large, yielding a very smooth density and a very simple level set tree. k is then reduced until the resulting tree fragments into many very small disconnected clusters. The population genetics application of Section 3.4 shows a somewhat simplified example of this. Level

[1] This idea came out of a conversation with Fabrizio Lecci, Alessandro Rinaldo, and Larry Wasserman.

set trees tend to be very similar for values of k between these two extremes.

- It is not really appropriate to ask when the level set tree method works or fails. The level set tree simply describes the structure of a density function; if the assumption of a pdf is reasonable, then the level set tree is applicable. We have shown in this thesis and in other work (not described here) that level set trees can be used even when the density assumption is *not* reasonable, as is the case with infinite-dimensional functional data and lattice-based arbitrary functions.

  Our view is that the level set tree should be estimated and used for exploratory data analysis in every statistical application, just as practitioners use histograms and scatterplot matrices to get an intuitive sense for the data. The ability of the level set tree dendrogram to act as a platform for interactive data exploration makes the method especially useful for this purpose.

- Despite the fact that the level set tree is an elemental statistical object, it is a difficult concept to describe. We suspect this is due in part because the notion of the population tree is often mixed up with the enumeration of tree estimation procedures, and we have tried in this thesis to separate the two. In fact, the idea of a true level set tree is what makes the method a statistical one in the first place.

  To make matters worse, various flavors of the level set tree and estimation algorithms have been reinvented by several different fields. The different terminology used by each field leads to a great deal of duplicated effort, and as the method gains popularity it will benefit from standardization of nomenclature. The literature review in Chapter 1 is intended to facilitate this process by providing an overview of some of the most popular methods, with an emphasis on organizing techniques based on their relationship to the underlying

density or pseudo-density function (or lack thereof).

Bibliography

Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. OPTICS: Ordering Points To Identify The Clustering Structure. In *Proceedings of the ACM SIGMOD'99 International Conference on Management of Data*, Philadelphia, PA, 1999. [8]

David Arthur and Sergei Vassilvitskii. K-means++: The Advantages of Careful Seeding. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007. doi: 10.1145/1283383.1283494. [35]

Adelchi Azzalini and Nicola Torelli. Clustering via Nonparametric Density Estimation. *Statistics and Computing*, 17(1):71–80, 2007. doi: 10.1007/s11222-006-9010-y. [4, 34, 35, 71]

Sivaraman Balakrishnan, Srivatsan Narayanan, Alessandro Rinaldo, Aarti Singh, and Larry Wasserman. Cluster Trees on Manifolds. *arXiv [stat.ML]*, (arXiv:1307.6515v1), 2013. [6, 40]

Jon Louis Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517, 1975. doi: 10.1145/361002.361007. [18]

Patrick Billingsley. *Probability and Measure*. Wiley, 2012. ISBN 978-1118122372. [3, 48]

Jacob D Bolzenius, David H Laidlaw, Ryan P Cabeen, Thomas E Conturo, Amanda R McMichael, Elizabeth M Lane, Jodi M Heaps, Lauren E Salminen, Laurie M Baker, John Gunstad, and Robert H Paul. Impact of Body Mass Index on Neuronal Fiber Bundle Lengths Among Healthy Older Adults. *Brain Imaging and Behavior*, 7:300–6, 2013. doi: 10.1007/s11682-013-9230-7. [81]

Peter Bubenik. Statistical Topology Using Persistence Landscapes. *arXiv*, (arXiv:1207.6437v1), 2012. [96]

Benoit Cadre, Bruno Pelletier, and Pierre Pudlo. Estimation of Density Level Sets with a Given Probability Content. *Journal of Nonparametric Statistics*, 25(1):261–272, 2013. [7]

Hamish Carr, Jack Snoeyink, and Ulrike Axen. Computing Contour Trees in All Dimensions. *Computational Geometry*, 24:75–94, 2003. [6]

Miguel A Carreira-Perpinan. Acceleration Strategies for Gaussian Mean-Shift Image Segmentation. In *Proc. of the 2006 IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, pages 1160–1167, New York, New York, USA, 2006. IEEE. [8]

Miguel A Carreira-Perpinan and Weiran Wang. The K-Modes Algorithm for Clustering. *arXiv*, (arXiv:1304.6478v1), 2013. [8]

Kamalika Chaudhuri and Sanjoy Dasgupta. Rates of Convergence for the Cluster Tree. In *Advances in Neural Information Processing Systems 23*, pages 343–351, Vancouver, BC, 2010. [4, 5, 6, 87, 93, 95]

Frederic Chazal, Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, Aarti Singh, and Larry Wasserman. On the Bootstrap for Persistence Diagrams and Landscapes. *arXiv*, (arXiv:1311.0376v1), 2013. [97]

Yizong Cheng. Mean Shift, Mode Seeking, and Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995. doi: 10.1109/34.400568. [8]

Ronald R Coifman and Stéphane Lafon. Diffusion Maps. *Applied and Computational Harmonic Analysis*, 21:5–30, 2006. doi: 10.1016/j.acha.2006.04.006. [35]

Dorin Comaniciu and Peter Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002. [8]

Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, third edition, 2009. [19]

Andrew Crossett, Brian P Kent, Lambertus Klei, Steven Ringquist, Massimo Trucco, Kathryn Roeder, and Bernie Devlin. Using Ancestry Matching to Combine Family-Based and Unrelated Samples for Genome-Wide Association Studies. *Statistics in medicine*, 29(28):2932–45, 2010. doi: 10.1002/sim.4057. [40]

Antonio Cuevas, Manuel Febrero, and Ricardo Fraiman. Estimating The Number of Clusters. *The Canadian Journal of Statistics*, 28(1973):1–17, 2000. [7]

Luc P Devroye and T J Wagner. The Strong Uniform Consistency of Nearest Neighbor Density Estimates. *The Annals of Statistics*, 5(3):536–540, 1977. [16]

Bogdan Draganski, Ferath Kherif, Stefan Klöppel, Philip a Cook, Daniel C Alexander, Geoff J M Parker, Ralf Deichmann, John Ashburner, and Richard S J Frackowiak. Evidence for Segregated and Integrative Connectivity Patterns in The Human Basal Ganglia. *The Journal of Neuroscience*, 28(28):7143–52, July 2008. doi: 10.1523/JNEUROSCI.1486-08.2008. [56, 57]

Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Knowledge Discovery and Data Mining*, pages 226–231, 1996. [7, 35]

Frederic Ferraty and Philippe Vieu. *Nonparametric Functional Data Analysis*. Springer, 2006. ISBN 9780387303697. [9, 48, 49, 50]

Jerome H Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An Algorithm for Finding Best Matches in Logarithmic Expected Time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977. [18]

Keinosuke Fukunaga and Larry D Hostetler. The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975. [8]

Eleftherios Garyfallidis, Matthew Brett, Bagrat Amirbekian, Christopher Nguyen, Fang-cheng Yeh, Yaroslav Halchenko, and Ian Nimmo-smith. Dipy - a Novel Software Library for Diffusion MR and Tractography. In *17th Annual Meeting of the Organization for Human Brain Mapping*, pages 1–5, 2011. [56]

Eleftherios Garyfallidis, Matthew Brett, Marta Morgado Correia, Guy B Williams, and Ian Nimmo-Smith. QuickBundles, a Method for Tractography Simplification. *Frontiers in neuroscience*, 6, 2012. ISSN 1662-453X. [47, 56, 64]

Ann M Graybiel and Clifton W Ragsdale. Histochemically Distinct Compartments in the Striatum of Human, Monkey, and Cat Demonstrated by Acetylthiocholinesterase Staining. *Proceedings of the National Academy of Sciences of the United States of America*, 75(11):5723–5726, 1978. [57]

P Guevara, C Poupon, D Rivière, Y Cointepas, M Descoteaux, B Thirion, and J Mangin. Robust Clustering of Massive Tractography Datasets. *NeuroImage*, 54 (3):1975–1993, 2011. doi: 10.1016/j.neuroimage.2010.10.028. [48]

Suzanne N Haber and Brian Knutson. The Reward Circuit: Linking Primate Anatomy and Human Imaging. *Neuropsychopharmacology*, 35(1):4–26, 2010. doi: 10.1038/npp.2009.129. [30, 57]

Suzanne N Haber, Ki-Sok Kim, Philippe Mailly, and Roberta Calzavara. Reward-Related Cortical Inputs Define a Large Striatal Region in Primates that Interface with Associative Cortical Connections, Providing a Substrate for Incentive-Based Learning. *The Journal of Neuroscience*, 26(32):8368–76, August 2006. doi: 10.1523/JNEUROSCI.0271-06.2006. [56]

Patric Hagmann, Lisa Jonasson, Philippe Maeder, Jean-philippe Thiran, Van J Wedeen, and Reto Meuli. Understanding Diffusion MR Imaging Techniques: From Scalar Diffusion-weighted Imaging to Diffusion Tensor Imaging and Beyond. *Radiographics*, 26(Supp):205–224, 2006. [30, 47, 59]

Patric Hagmann, Leila Cammoun, Xavier Gigandet, Reto Meuli, Christopher J Honey, Van J Wedeen, and Olaf Sporns. Mapping the Structural Core of Human Cerebral Cortex. *PLoS Biology*, 6(7):1479–1493, 2008. [47]

J A Hartigan. Consistency of Single Linkage for High-Density Clusters. *Journal of the American Statistical Association*, 76(374):388–394, 1981. [2, 3, 5, 39, 87]

John Hartigan. *Clustering Algorithms*. John Wiley & Sons, 1975. ISBN 978-0471356455. [1, 2, 3, 5, 35]

Trevor Hastie, Andreas Buja, and Robert Tibshirani. Penalized Discriminant Analysis. *The Annals of Statistics*, 23(1):73–102, 1995. [49, 50]

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, second edition, 2009. doi: 978-0387848570. [35]

Lawrence Hubert and Phipps Arabie. Comparing Partitions. *Journal of Classification*, 2:193–218, 1985. [62]

Brian P Kent, Alessandro Rinaldo, Fang-Cheng Yeh, and Timothy Verstynen. Mapping Topographic Structure in White Matter Pathways with Level Set Trees. *PLoS ONE (submitted)*, 2013. [30, 56]

Jussi Klemelä. Visualization of Multivariate Density Estimates With Level Set Trees. *Journal of Computational and Graphical Statistics*, 13(3):599–620, 2004. doi: 10.1198/106186004X2642. [4, 22]

Jussi Klemelä. Algorithms for Manipulation of Level Sets of Nonparametric Density Estimates. *Computational Statistics*, 20(2):349–368, 2005. doi: 10.1007/BF02789708. [4]

Samory Kpotufe and Ulrike Von Luxburg. Pruning Nearest Neighbor Cluster Trees. *Proceedings of the 28th International Conference on Machine Learning*, 105:225–232, 2011. [6, 16, 93, 95]

H W Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955. [63]

F Kent Kuiper and Lloyd Fisher. 391: A Monte Carlo Comparison of Six Clustering Procedures. *Biometrics*, 31(3):777–783, 1975. [2]

Chris Landsea, James Franklin, and Jack Beven. The Revised Atlantic Hurricane Database (HURDAT2). Technical Report February, U.S. National Hurricane Center, 2013. [52]

Elizaveta Levina and Peter Bickel. The Earth Movers Distance is the Mallows Distance: Some Insights from Statistics. *Proceedings of the 8th IEEE Conference on Computer Vision*, 2:251–6, 2001. [76]

Jun Z Li, Devin M Absher, Hua Tang, Audrey M Southwick, Amanda M Casto, Sohini Ramachandran, Howard M Cann, Gregory S Barsh, Marcus Feldman, Luigi L Cavalli-Sforza, and Richard M Myers. Worldwide Human Relationships Inferred from Genome-wide Patterns of Variation. *Science*, 319(5866):1100–4, 2008. doi: 10.1126/science.1153717. [40]

Ting Liu, Andrew W Moore, and Alexander Gray. New Algorithms for Efficient High-Dimensional Nonparametric Classification. *Journal of Machine Learning Research*, 7:1135–1158, 2006. [18]

Stuart P Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. doi: 10.1109/TIT.1982.1056489. [1]

J MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1:281–297, 1967. [1]

Markus Maier, Matthias Hein, and Ulrike von Luxburg. Optimal Construction of K-Nearest-Neighbor Graphs for Identifying Noisy Clusters. *Theoretical Computer Science*, 410(19):1749–1764, 2009. doi: 10.1016/j.tcs.2009.01.009. [7, 16]

Giovanna Menardi and Adelchi Azzalini. An Advancement in Clustering via Nonparametric Density Estimation. *Statistics and Computing*, 2013. doi: 10.1007/s11222-013-9400-x. [6]

Michael C Minnotte and David W Scott. The Mode Tree: A Tool for Visualization of Nonparametric Density Features. *Journal of Computational and Graphical Statistics*, 2(1):51–68, 1993. [8]

Bart Moberts, Anna Vilanova, and Jarke J Van Wijk. Evaluation of Fiber Clustering Methods for Diffusion Tensor Imaging. *IEEE Visualization*, pages 65–72, 2005. [54, 57]

Andrew W Moore. The Anchors Hierarchy : Using the Triangle Inequality to Survive High Dimensional Data. In *Uncertainty in Artificial Intelligence Proceedings*, pages 397–405, 2000. [18]

James Munkres. Algorithms for the Assignment and Transportation Problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1):32–8, 1957. [63]

Laurent Najman and Michel Couprie. Building the Component Tree in Quasi-Linear Time. *IEEE Transactions on Image Processing*, 15(11):3531–9, November 2006. ISSN 1057-7149. [19, 94]

Lauren J O'Donnell, Alexandra J Golby, and Carl-Fredrik Westin. Fiber Clustering Versus the Parcellation-Based Connectome. *NeuroImage*, 2013. doi: 10.1016/j.neuroimage.2013.04.066. [47, 54, 57]

Patrick Oesterling, Christian Heine, Heike Janicke, Gerik Scheuermann, and Gerhard Heyer. Visualization of High-Dimensional Point Clouds Using Their Density Distributions Topology. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1547–1559, 2011. [6, 22, 97]

Patrick Oesterling, Christian Heine, Gunther H Weber, and Gerik Scheuermann. Visualizing nD Point Clouds as Topological Landscape Profiles to Guide Local Data Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 19 (3):514–26, March 2013. doi: 10.1109/TVCG.2012.120. [22, 97]

Stephen M Omohundro. Five Balltree Construction Algorithms. Technical report, International Computer Science Institute, Berkeley, California, 1989. [18]

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. [36]

Mathew D Penrose. Single Linkage Clustering and Continuum Percolation. *Journal of Multivariate Analysis*, 53:94–109, 1995. [5]

Wolfgang Polonik. Measuring Mass Concentrations and Estimating Density Contour Clusters - An Excess Mass Approach. *The Annals of Statistics*, 23(3):855–881, 1995. [7]

R Core Team. R: A Language and Environment for Statistical Computing. Technical report, R Foundation for Statistical Computing, Vienna, Austria, 2012. [36]

Clifton W Ragsdale and Ann M Graybiel. A Simple Ordering of Neocortical Areas Established by The Compartmental Organization of Their Striatal Projections. *Proceedings of the National Academy of Sciences of the United States of America*, 87:6196–9, August 1990. [57]

Joseph W Richards, Peter E Freeman, Ann B Lee, and Chad M Schafer. Exploiting Low-Dimensional Structure in Astronomical Spectra. *The Astrophysical Journal*, 691(1):32–42, 2009a. doi: 10.1088/0004-637X/691/1/32. [37]

Joseph W Richards, Peter E Freeman, Ann B Lee, and Chad M Schafer. Accurate Parameter Estimation for Star Formation History in Galaxies Using SDSS Spectra. *Monthly Notices of the Royal Astronomical Society*, 399(2):1044–1057, 2009b. [37]

Alessandro Rinaldo and Larry Wasserman. Generalized Density Clustering. *The Annals of Statistics*, 38(5):2678–2722, 2010. doi: 10.1214/10-AOS797. [7, 39]

Alessandro Rinaldo, Aarti Singh, Rebecca Nugent, and Larry Wasserman. Stability of Density-Based Clustering. *Journal of Machine Learning Research*, 13:905–948, 2012. [4, 9, 13, 93]

Noah A Rosenberg. Standardized Subsets of the HGDP-CEPH Human Genome Diversity Cell Line Panel, Accounting for Atypical and Duplicated Samples and Pairs of Close Relatives. *Annals of Human Genetics*, 70(Pt 6):841–7, 2006. doi: 10.1111/j.1469-1809.2006.00285.x. [40]

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The Earth Mover's Distance as a Metric for Image Retrieval. Technical report, Stanford University, 1998. [76]

Jianbo Shi and Jitendra Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. [1]

P H A Sneath. The Evaluation of Cluster Methods. In A.J. Cole, editor, *Numerical Taxonomy*. Academic Press, London, 1969. [2]

Bharath K Sriperumbudur and Ingo Steinwart. Consistency and Rates for Clustering with DBSCAN. *JMLR Workshop and Conference Proceedings*, 22:1090–1098, 2012. [7]

Ingo Steinwart. Adaptive Density Level Set Clustering. *Journal of Machine Learning Research: Workshop and Conference Proceedings - 24th Annual Conference on Learning Theory*, pages 1–35, 2011. [4, 7]

Werner Stuetzle and Rebecca Nugent. A Generalized Single Linkage Method for Estimating the Cluster Tree of a Density. *Journal of Computational and Graphical Statistics*, 19(2):397–418, 2010. doi: 10.1198/jcgs.2009.07049. [5, 14, 89, 94]

Timothy D Verstynen, David Badre, Kevin Jarbo, and Walter Schneider. Microstructural Organizational Patterns in The Human Corticostriatal System. *Journal of Neurophysiology*, 107:2984–95, 2012a. doi: 10.1152/jn.00995.2011. [56, 57]

Timothy D Verstynen, Andrea M Weinstein, Walter W Schneider, John M Jakicic, Dana L Rofey, and Kirk I Erickson. Increased Body Mass Index Is Associated With a Global and Distributed Decrease in White Matter Microstructural Integrity Timothy. *Psychosomatic Medicine*, 74(7):682–690, 2012b. [80]

Timothy D Verstynen, Andrea Weinstein, Kirk I Erickson, Lei K Sheu, Anna L Marsland, and Peter J Gianaros. Competing Physiological Pathways Link Individual Differences in Weight and Abdominal Adiposity to White Matter Microstructure. *NeuroImage*, 79:129–37, 2013. doi: 10.1016/j.neuroimage.2013.04.075. [80]

Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010. [62]

Ulrike von Luxburg. A Tutorial on Spectral Clustering. *Max Planck Institute for Biological Cybernetics*, TR-149(August), 2006. [35, 37]

V J Wedeen, R P Wang, J D Schmahmann, T Benner, W Y I Tseng, G Dai, D N Pandya, P Hagmann, H D'Arceuil, and A J de Crespigny. Diffusion Spectrum Magnetic Resonance Imaging (DSI) Tractography of Crossing Fibers. *NeuroImage*, 41:1267–77, 2008. doi: 10.1016/j.neuroimage.2008.03.036. [47]

Van J Wedeen, Douglas L Rosene, Ruopeng Wang, Guangping Dai, Farzad Mortazavi, Patric Hagmann, Jon H Kaas, and Wen-Yih I Tseng. The Geometric Structure of The Brain Fiber Pathways. *Science*, 335:1628–34, 2012. doi: 10.1126/science.1215280. [47]

D Wishart. Mode Analysis: A Generalization of Nearest Neighbor Which Reduces Chaining Effects. In Alfred John Cole, editor, *Proceedings of the Colloquium on Numerical Taxonomy held in the University of St. Andrews*, pages 282–308, 1969. [5, 87]

Anthony Wong and Tom Lane. A kth Nearest Neighbour Clustering Procedure. *Journal of the Royal Statistical Society: Series B*, 45(3):362–368, 1983. [5]

Fang-Cheng Yeh, Timothy D Verstynen, Yibao Wang, Juan C Fernández-Miranda, and Wen-Yih Isaac Tseng. Deterministic Diffusion Fiber Tracking Improved by Quantitative Anisotropy. *PLoS ONE*, 8(11), 2013. doi: 10.1371/journal.pone.0080713. [47]

Xiao-Tong Yuan, Bao-Gang Hu, and Ran He. Agglomerative Mean-Shift Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 24(2):209–219, 2012. [8]