Improving the Cluster Structure Extracted from OPTICS Plots

Erich Schubert and Michael Gertz

Heidelberg University
{schubert,gertz}@informatik.uni-heidelberg.de

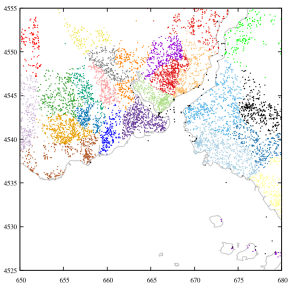
Abstract. Density-based clustering is closely associated with the two algorithms DBSCAN and OPTICS. While the first finds clusters connected at a single density threshold, the latter allows the extraction of a cluster hierarchy based on different densities. Extraction methods for clusters from OPTICS rely on an intermediate representation, known as the OPTICS plot. In this plot, which can be seen as a density profile of the data set, valleys (areas of higher density) are associated with clusters. Multiple methods for automatic detecting such valleys have been proposed, but all of them tend to produce a particular artifact, where some point is included in the cluster that may be far away from the remainder. In this article, we will discuss this commonly seen artifact, and propose a simple but sound way of removing the artifacts. At the same time, this change is minimally invasive, and tries to keep the existing algorithms largely intact for future study.

1 Introduction

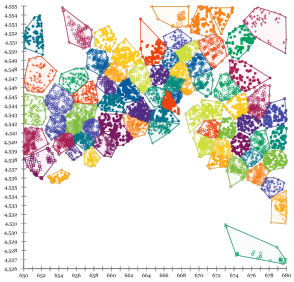
Cluster analysis is the task of discovering previously unknown structure in a data set, and belongs to the "unsupervised learning" subdomain of data mining where no labeled data is available yet. It can be used to infer an initial labeling of the data, but clustering results tend to not be reliable enough to automate such use. But it can be used in explorative data analysis to assist the user in discovering patterns that may lead to meaningful labels after formalization. For example, a clustering might suggest three subtypes of a disease to the user, and clinical verification may be able to confirm two of these three subtypes with appropriate thresholds to differentiate them. Here, the clustering itself does not need to produce "perfect" results, but it was used to find a hypothesis.

While clustering is considered an unsupervised task—it cannot learn from labels, in contrast to classification—it is not entirely free of assumptions on the data. In hierarchical clustering, the key assumption is that nearby objects should be in the same cluster, while far away objects belong to different clusters. Probably the most used clustering method, k-means clustering [24, 25, 16], is based on the assumption that the data consists of k groups, such that the sum of squared errors is minimized within clusters, and maximized inbetween clusters. While this is useful in many applications such as image quantization, it does not work very well when the clusters are of different shape and density, or if there are too many outliers in the data set.

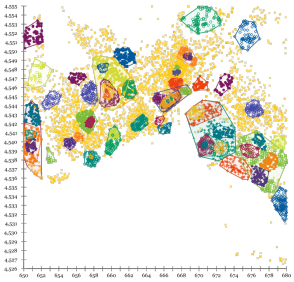
The key assumption of density-based clustering is that regions of similar density—when connected to each other—should belong to the same cluster. For this, the seminal algorithm DBSCAN [15, 33] introduced the concepts of "density reachability" and



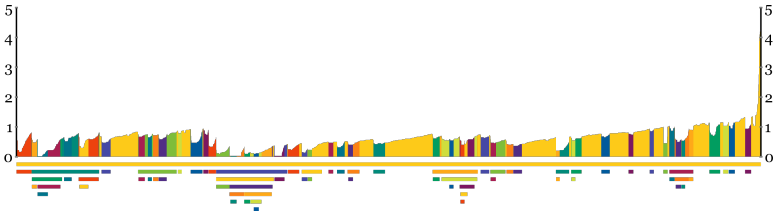
(a) Input data, with coastline and administrative districts



(b) Best k-means clustering, VRC criterion [11], $k = 75$



(c) Hierarchical density based clustering with OPTICS



(d) OPTICS plot, with valleys that correspond to clusters

Fig. 1: Istanbul tweets data set

"density connectedness", where clusters of DBSCAN are those objects, which form a density connected component. DBSCAN required the specification of two parameters: minPts and ε (plus the implicit parameter of a distance function) which together specify a minimum density requirement: points with more than minPts neighbors within a radius ε are considered dense and called "core points", and those neighbors are considered to be direct density-reachable from this seed points. The algorithm then computes the transitive closure of this reachability relation. Where DBSCAN used a binary predicate of density, OPTICS [6] only retains the minPts requirement: in OPTICS, a point is dense at a radius r if it has at least minPts neighbors within this radius. OPTICS is a greedy algorithm, which always processes the neighbors of the most dense points first (by organizing points in a priority queue), and produces an ordering of the points known as "cluster order". The diagram placing objects on the x axis according to their cluster order, and the density at which the point was reached on the y axis, is known as the "OPTICS plot". If a cluster exhibits a density clustering structure, this plot will exhibit valleys corresponding to the clusters. If there is a hierarchy of clusters, there may be smaller valleys inside larger valleys, as it can be seen in Fig. 1d. Such plots are easy to use for the researcher if the data set is not too large, and multiple algorithms have been proposed to automatically extract clusters: along with OPTICS, the authors proposed to detect valleys based on a minmum relative distance change of ξ [6]. In [29], the authors order maxima of the OPTICS plot by their significance, and only use significant maxima for splitting the data set. The main benefit of this approach is that it produces a simpler and thus easier to use cluster hierarchy than the ξ method. An angle-based extraction based on inflexion points was introduced in [8]. The main benefit of this approach is that it takes a larger context into account, where the ξ method would only look at the distances of two subsequent objects in the plot.

In Fig. 1 we introduce our running example. This data set contains 10,000 coordinates derived from real tweets in Istanbul. The raw coordinates can be seen in Fig. 1a, along with the coastline for illustration, and colored by administrative areas. Classic k-means clustering can be seen in Fig. 1b, with $k{=}75$ chosen by the Caliński-Harabasz criterion [11] (but this heuristic did not yield a clear "best" k). A hierarchical density-based clustering is shown in Fig. 1c. This clustering does not partition all data points equally, but it emphasizes region of higher density, possibly even nested. Many of these regions correspond to neighborhood centers and shopping areas. Fig. 1d shows the OPTICS plot for this clustering.

The remainder of this article is structured as follows: In Section 2, we examine related work. Then we introduce the basics of OPTICS in Section 3 and cluster extraction from OPTICS plots in Section 3.3. Then we will discuss the extraction problem effecting all methods based solely on the OPTICS plot, as well as introducing a simple remedy for this problem in Section 4. Finally, we will demonstrate the differences on some example data sets in Section 5, and conclude the article in Section 6.

2 Related Work

DBSCAN [15] is the most widely known density based clustering method. In essence, the algorithm finds all *connected components* in a data set which are connected at a given density threshold given by a maximum distance ε and the minimum number of points minPts contained within this radius. The approach has been generalized to allow other concepts of neighborhoods as well as other notions of what constitutes are cluster core point [28]. In [33], the authors revisit DBSCAN after 20 years, provide an abstract form, discuss the runtime in more detail, the algorithms limitations and provide some suggestions on parameter choices (in particular, how to recognize and avoid parameters). In OPTICS [6] the approach was modified to produce a hierarchy of clusters by keeping the minPts parameter, but varying the radius. Objects that satisfy the density requirement at a smaller radius connect earlier and thus form a lower level of the hierarchy. OPTICS however only produces an order of points, from which the clusters either need to be extracted using visual inspection [6], the ξ method [6], using significant maxima [29] or inflexion points [8]. The relationship of OPTICS plots to dendrograms (as known from hierarchical agglomerative clustering) has been discussed in [29].

Outside of clustering, the concepts of DBSCAN and OPTICS also were used for outlier detection algorithms such as the local outlier factor LOF [10] and the closely related OPTICS-OF outlier factor [9]. The concept of density-based clustering was also inspiration for the algorithms DenClue [20] and DenClue 2 [19], which employ grid-based strategies to discover modes in the density distribution. Both belong to the wider family of mode-seeking algorithms such as mean-shift clustering orignating in image analysis and statistical density estimation [17, 14]. The algorithm DeLiClu [3] employs an R*-tree [7] to improve scalability, by performing a nearest-neighbor self-join on the tree. This can greatly reduce the number of distance computations, but increases the implementation complexity and makes the algorithm harder to optimize. It also removes the need to set the ε parameter of OPTICS, as the join will automatically stop when everything is connected, without computing all pairwise distances.

```
SeedList ← empty priority queue
ClusterOrder ← empty list
repeat
    if SeedList.empty() then
        if no unprocessed points then stop
        (r, o) ← (∞, next unprocessed point)      // get unprocessed point
    else (r, o) ← SeedList.deleteMin()        // next point from seed list
    N ← RANGEQUERY(o, dist, ε)                    // get neighborhood
    ClusterOrder.add( (r, o, CoreDist(o, N)) )     // add to cluster order
    Mark o as processed
    if p is Core then foreach n ∈ N do             // explore neighborhood
        if n is processed then continue
        p ← ReachDist(n ← o)                       // compute reachability
        if n ∈ SeedList then SeedList.decreaseKey( (p, n) )          // update
                        else SeedList.insert( (p, n) )               // insert
```

Algorithm 1: OPTICS Clustering Algorithm [6]

FastOPTICS [30, 31] approximates the results of OPTICS using 1-dimensional random projections, suitable for Euclidean space. The concepts of OPTICS were transferred to subspace clustering in the algorithms HiSC [2] and DiSH [1], for correlation clustering in HiCO [4], and to uncertain data in FOPTICS [22]. HDBSCAN* [12] is a revisited version of DBSCAN, where the concept of border points was removed, which yields a cleaner theoretical formulation of the algorithm, even closer connected to graph theory. In [13] the authors propose a general extraction strategy for hierarchical clusters that works well with HDBSCAN*, assuming that the hierarchy essentially corresponds to a minimum spanning tree. HDBSCAN* can be accelerated using dual-tree joins [26].

3 OPTICS Clustering

We will now first briefly review the core ideas of OPTICS clustering. The first idea is the notion of density reachability, defining at which distance two neighbors are connected. Clusters are then connected components of this reachability.

3.1 Reachability Model of OPTICS

OPTICS [6] uses the concept of "reachability distance" for constructing clusters. This distance correspondy closely to the ε radius parameter of DBSCAN, and can be seen as an inverse density estimate: all points within a cluster are "density connected" with at least this density. Reachability is the maximum of two factors, the first being the actual distance of the two points, and the second being the core-distance of the origin point. In OPTICS [6], the core-distance was defined as

$$\text{core-dist}_{\varepsilon,\text{minPts}}(o) := \begin{cases} \text{UNDEFINED} & \text{if } |\{x \mid d(x,o) \leq \varepsilon_{\max}\}| < \text{minPts} \\ \text{minPts-dist}(o) & \text{otherwise} \end{cases}$$

where minPts-dist is the distance to the minPts nearest neighbor. In the following, we will omit the $\varepsilon_{\max}$ parameter and the associated UNDEFINED case from the original

definitions for brevity.[1] Furthermore, we can simply use infinity ∞ instead of a special UNDEFINED value. If $\varepsilon_{\max}$ is chosen large enough, this case does not occur, and we can substitute minPts-dist for the core-distance.

We can then define the *reachability* of a point p from a point o as:[2]

$$\mathrm{reachability}(p \leftarrow o) := \max\{\mathrm{dist}(o,p), \mathrm{minPts\text{-}dist}(o)\}$$

Reachability, while originally called "reachability distance" is not a distance as defined in mathematics, because it is not symmetric. This asymmetry is often causing confusion, and we thus will only refer to it simply as "reachability" here. Intuitively, this value is the minimum distance threshold ε that we need to choose, to make p density-reachable from o and thus part of the same DBSCAN cluster. OPTICS is based on the same idea of clusters being connected components based on a reachability threshold.

The use of core distances prevents (for large enough minPts) the undesirable chaining effect known as "single-link effect" that affects hierarchical clustering, in particular on noisy data. For practical purposes, minPts must thus be chosen not too small, but values such as $\mathrm{minPts}{=}5$ may be sufficient for small 2-dimensional data sets with little noise. The suggestions in [33] for choosing minPts in DBSCAN may be applicable.

3.2 Algorithmic Aspects of OPTICS

The OPTICS agorithm is an efficient algorithm to approximate the density clustering structure given by the reachability distance introduced above. It uses a greedy approach to find dense areas: it always processes the point with the currently lowest known reachability next (intuitively, the point with the highest expected density), removes it from the candidates and adds it to the output list, then updates the reachability of its yet unprocessed neighbors if (and only if) it decreased.

The output of OPTICS is an ordering of the database along with the reachability of each object, but not clusters in the traditional sense of data partitions. Instead, different visual and automatic methods can be used to extract clusters from this order. The cluster order is not fully deterministic because points with the same distance may be processed in any order. Thus, different runs may yield different results that nevertheless correspond to a highly similar cluster structure.

A key design aspect of the OPTICS algorithm is its efficiency *if* the database can accelerate radius queries. If the database can answer such queries for the given data set, distance function, and radius $\varepsilon_{\max}$ within $\log n$ time on average, then the run-time of OPTICS can be $n \log n$ instead of $O(n^2)$, making it suitable for large data sets.[3] For too large radius $\varepsilon_{\max}$, or "malicious" data sets, such run-times can usually not be guaranteed (c.f., [33]), so the worst-case run-time supposedly remains $O(n^2)$. In many practical applications, speedups of this magnitude can be observed empirically with indexes such as the k-d-tree and R*-tree [7].

[1] The $\varepsilon_{\max}$ parameter serves the purpose of allowing index acceleration, but does not contribute to the theoretical clustering model. Ideally, $\varepsilon_{\max}$ is chosen to not affect the result, practically we want to use the smallest $\varepsilon_{\max}$ that still gives satisfactory results for best efficiency.

[2] The definition in [6] had to handle the UNDEFINED case specially, we simply use ∞.

[3] We avoid the O notation here, because indexes usually can *not* guarantee $O(\log n)$ query time.

OPTICS uses a priority queue for processing points. Initially the queue is empty, and if a maximum distance $\varepsilon_{\max}$ is used, it may become empty again. If empty, an unprocessed point is chosen at random with a reachability of ∞. OPTICS polls one point o at a time from this priority queue, adds it to the cluster order with its current reachability, marks it as visited, and searches its neighborhood: For every neighbor p of o that has not yet been visited, $\mathrm{reachability}(p \leftarrow o)$ is computed, and the priority queue is updated, unless it already contains a better reachability for p.

The approach used by OPTICS bears similarities with Prim's algorithm [27] for computing the minimum spanning tree; except that for efficiency OPTICS only considers edges with a maximum length of $\varepsilon_{\max}$; and the output is a linear order instead of a spanning tree. Intuitively, OPTICS builds the cluster order by always adding the point next which is best reachable from all previous points (and drawing one at random, if no point is reachable or there is no previous point).

3.3 Cluster Extraction from OPTICS Plots

Density-based clusters correspond to "valleys" in OPTICS plots as seen in Fig. 1d, but the formal definition of a valley turns out to be more difficult than expected.

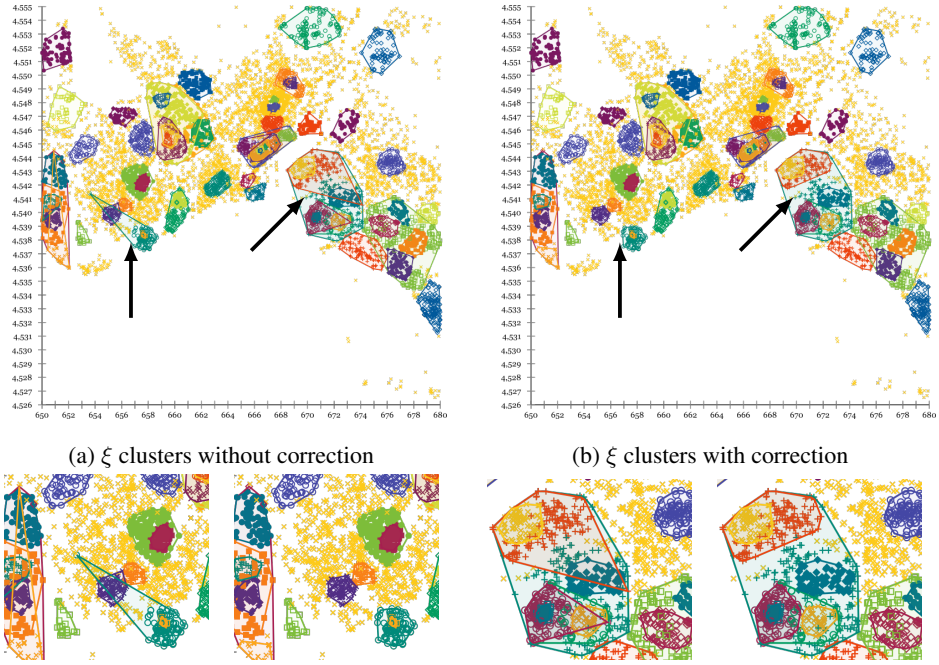
OPTICS [6] defined the notion of ξ-steep points if the reachability of two successive points in the cluster order differs by a factor of more than $1-\xi$. Downward steep points are candidates for the beginning of a cluster, and upward steep points are candidates for the end. The exact definitions of ξ-clusters involves handling of additional constraints, such as ensuring a minimum cluster size and taking the minimum in-between value into account for inferring the hierarchical structure. Details on this can be found in [6].

This extraction method operates only on the OPTICS plot only, not on the original data: it identifies subsequences that exhibit a steep gradient, and then selects the corresponding subsequence of the cluster order. It works as desired, and often would select those areas a human expert would select base on visual inspection of the OPTICS plot.

When running this algorithm on larger data sets, certain artifacts have been observed when inspecting the cluster in detail. This becomes most obvious when drawing the convex hulls of cluster of geographical points, as seen in Fig. 2a for Tweet locations in the Bosporus region (Fig. 2c and 2d are close-ups; lines are the convex hulls of the cluster points): some clusters have a spike extending from them that does not appear to be correct. Closer inspection revealed that these are always the last few objects within a cluster, and this problem can be provoked on a much smaller data set.

4 Using Predecessor Information for Improved Cluster Extraction

The implementation of OPTICS in ELKI [5] includes additional information in the data structures, useful for visualizing the clustering process of OPTICS: it not only tracks the reachability distance, but also the *predecessor* that had the smallest reachability. This was added to ELKI for the DiSH algorithm [1], an OPTICS variant for finding subspace cluster hierarchies. While DiSH used this information to compare the "subspace preference" of a point to its predecessor, it turned out that this is exactly the information needed for resolving ambiguity of wheter a point at the end of a valley still belongs to



(a) ξ clusters without correction

(b) ξ clusters with correction

(c) Details of artifacts
(left: uncorrected / right: corrected)

(d) Details of artifacts
(left: uncorrected / right: corrected)

Fig. 2: OPTICS without and with the proposed filtering

Data: $o(i)$: Object at position i
Data: $r(i)$: Reachability at position i
Data: $p(i)$: Predecessor of object at position i
Data: s: Start position of cluster within cluster order
Data: e: End position of cluster within cluster order (inclusive)

```
while s < e do
    if r(s) > r(e) then return (s, e) ;        // Rest must be consistent
    for i = s to e − 1 do                      // Search for predecessor
        if o(i) = p(e) then return (s, e) ;    // Found - consistent
    e ← e − 1 ;                                // Not found, shrink cluster
return null ;                                  // Completely inconsistent
```

Algorithm 2: Cluster refinement algorithm, to postprocess each cluster (s, e).

the cluster or not: if its predecessor is included in the cluster then the ambiguous point is also part of the cluster, but if the predecessor is in a parent cluster, the point should be part of the parent (or a subsequent cluster in the plot).

Algorithm 2 gives the pseudocode of this surprisingly simple refinement algorithm, which can be integrated easily into the ξ cluster extraction or other methods for extracting clusters from OPTICS plots. (The artifact is not specific to ξ clusters, but these artifacts will occur in any method based on valleys in the cluster order.) The predecessor of the last (by cluster order) element in the cluster is searched in the cluster. If

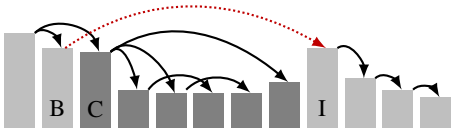


Fig. 3: Predecessors in the OPTICS plot.

it is found, the algorithm terminates, but if the predecessor is not found, then the last element is discarded and the algorithm continues. Because of the priority heap used in OPTICS, all points with a reachability less than the first point must have their predecessor in the cluster, so most points will cause successful termination. The necessary predecessor information can trivially be gathered while computing the OPTICS plot, and needs only $O(n)$ additional memory for storing the predecessors. Since we only have to look at a few elements at the end of each cluster, the runtime impact of this refinement is negligible compared to the cost of finding the neighbors.

Fig.2a visualizes the clusters without this algorithm, and Fig.2b after post-processing the clusters, with hardly visible differences. We indicate two (but not all) differences with arrows, and show these regions in more detail in Fig. 2c and Fig. 2d.

4.1 Theoretical Support for this Correction

This correction technique is well supported by theory if we realize that the OPTICS plot is a linearized version of a spanning tree. If there exist multiple subtrees in the data, only one subtree can immediately follow in the OPTICS cluster order. After the first subtree, there will be some object which is the first of the next subtree. This object will necessarily have a higher reachability than the first object of the first subtree (because of the priority queue). The converse is, however, not true, and we cannot rely on the reachability alone. Instead, we need to consider the predecessor of the point.

Fig. 3 illustrates this principle. Points C to I consitute a valley. But because the predecessor of I is B, and B is not part of the valley, it is not part of the cluster (the reachability in the plot is $\mathrm{reachability}(I \leftarrow B)$, and the reachability of I from any point between B and I is at least as much).

Line 2 is a simple shortcut to avoid having to search the predecessor: When the point s was added to the cluster order, it must have been the point with minimum reachability (because of the priority queue). If $r(s) > r(e)$, this implies that some predecessor of e in the cluster order between s and e must have caused the reachability of e to decrease to this value $r(e)$, and therefore this predecessor must be in the cluster. Note that for $r(s) \leq r(e)$ it is no longer guaranteed that $r(e)$ decreased after s was added. In such cases, we employ the linear search in Line 3. We could avoid this search by storing the cluster order position of each element with additional $O(n)$ memory, but the search has very low constant factors, and is not invoked very often.

While the proposed technique is an easy and effective modification of the existing cluster extraction techniques—and should be integrated to improve the results—it raises the question whether the linearization of the spanning tree into the cluster order is appropriate in the first place. Instead, it may be reasonable to use a spanning directly instead of the processing order used by OPTICS. In fact, the recently proposed

method HDBSCAN does exactly this: constructing a density based spanning tree of the data, and extracting clusters from this tree; unfortunately at the cost of increasing the complexity of the published algorithm to $O(n^2)$.

The observation that OPTICS plots relate to dendrograms is not completely new: In [29], the authors discuss how to convert an OPTICS cluster order into a dendrogram, or a dendrogram into a cluster order, but without realizing the role of the predecessor, and how this can be used here. And while OPTICS plots are a key contribution of the method, the roots of this visualization can be traced back to Ward [35], who used arrows of different length to indicate the clustering structure. Other early examples include skyline plots [36], the diagrams used by Johnson [21], and the "linear representations" of clusters in the book by Hartigan [18]. Icicle plots [23] closely resemble the OPTICS plot, but the main contribution claimed is to use the object labels for printing instead of blocks. These early methods will only scale to a few objects, as they usually require two characters or lines per object. The authors argue that these plots are easier to read off than the dendrograms more commonly used with hierarchical clustering. We believe we are the first to point out the close relationship between these icicle/skyscraper plots and OPTICS reachability plots. It is worth noting that the older plots plotted the separation distance *inbetween* of two points, whereas OPTICS reachability plots use the reachability distance of a point. In fact, the reachability is better interpreted as the separation from the points' predecessor(s), i.e., the cophenetic distance [34].

So while we should rather define OPTICS clusters as *subtrees* of the implicit spanning tree, rather than further using the current visual approach based on the cluster order and plot. But as mentioned before this ultimately leads to the HDBSCAN* [12] algorithm. In this paper, we rather discuss a small modification to OPTICS cluster extraction, that can easily be added to an existing implementation.

5 Experiments

We continue with the running example using Tweets in Istanbul introduced in Section 3. Fig. 2 shows the clusters without and with the proposed modification. In Table 1 we show the change in density caused by our filtering approach. While it would be possible for a cluster to disappear because of filtering (if it shrinks below the minimum cluster size), this did not occur here. Out of 70 clusters (hierarchical, including the top level cluster containing everything), only 7 clusters were modified, and a total of 11 points were assigned to the parent cluster instead. With most evaluation measures such as the adjusted Rand index (ARI), normalized mutual information (NMI), but also most internal measures, this change would only make a *negligible difference, as only 0.11% of points have changed the cluster assignment.*

Therefore, we evaluate the change in cluster area instead. For each cluster, we compute the convex hull, and compute the polygon area using the Shoelace formula. As we can see in Table 1, removing a single misassigned point reduced the area of the polygon up to 80%. For comparison, we give the *maximum* and the *average* area reduction possible by removing a single cluster member. As we can see, the filtered clusters are much less affected by random removal. We also include the four non-modified clusters with the largest possible area reduction as comparison. While we can see that while

Table 1: Change in cluster area by removing points.

	points		filtered				unfiltered		
Clu.#	before	change	area	**reduced**	max	rnd	area	max	rnd
#18	313	-4	3.09	-30.6%	-5.8%	-0.1%	4.46	-28.4%	-0.1%
#27	137	-1	0.79	-49.2%	-8.0%	-0.2%	1.56	-49.2%	-0.5%
#49	119	-1	1.23	-12.4%	-4.7%	-0.1%	1.40	-12.4%	-0.2%
#50	179	-1	2.23	-61.4%	-7.6%	-0.1%	5.78	-61.4%	-0.4%
#56	271	-1	7.43	-43.0%	-1.3%	-0.0%	13.04	-43.0%	-0.2%
#64	53	-2	1.04	-32.8%	-6.0%	-0.4%	1.54	-30.4%	-0.9%
#65	58	-1	1.27	-80.1%	-5.8%	-0.2%	6.37	-80.1%	-1.5%
#31	59	0					0.15	-13.4%	-0.6%
#43	94	0					1.71	-14.0%	-0.3%
#51	51	0					0.08	-17.2%	-0.9%
#68	81	0					6.03	-12.9%	-0.4%

these have a larger area reduction than cluster #49, they are also smaller, and we can thus expect the area to depend more on the individual points. This also shows in the larger area reduction when removing a random point. But most importantly, our approach is well supported by theory, as opposed to a heuristic threshold-based cluster pruning. Furthermore, cluster #51 is already unusually small, with just 0.077 km^2: the Sinan Erdem Dome, the tiny orange cluster embedded in the larger cluster (which is cluster #50, corresponding to Ataköy, Bakırköy) in the bottom right of Fig. 2c.

6 Conclusions

In this paper, we improve OPTICS clustering by storing the predecessor of each point, and taking this information into account during cluster extraction. The changes to the resulting clustering is small, usually just 0-2 samples per cluster. Because of this, this difference easily goes unnoticed when just looking at clustering evaluation measures such as the adjusted Rand index (ARI), or normalized mutual information (NMI). However because these points do not fit the resulting clusters well, they have a major impact on the volume of the cluster. This can easily be seen when using the convex hull on geographic data. The same effect however exists in other distances and in high-dimensional data as well. The proposed improvement is well supported by theory, as it prunes points based on the spanning tree used implicitly by OPTICS. Because the necessary improvements require only $O(n)$ additional memory and neglibile run-time, they should be used by default, and have in an earlier variant already been integrated in the open-source ELKI [32] data mining framework since version 0.7.0 as well as the `dbscan` R package since version 1.0-0. This article explains the theoretical foundation of this beneficial filtering technique.

Bibliography

[1] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek. "Detection and Visualization of Subspace Cluster Hierarchies". In: *Proc. DASFAA*. 2007, pp. 152–163.

[2] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek. "Finding Hierarchies of Subspace Clusters". In: *Proc. PKDD*. 2006, pp. 446–453.

[3] E. Achtert, C. Böhm, and P. Kröger. "DeLiClu: Boosting Robustness, Completeness, Usability, and Efficiency of Hierarchical Clustering by a Closest Pair Ranking". In: *Proc. PAKDD*. 2006, pp. 119–128.

[4] E. Achtert, C. Böhm, P. Kröger, and A. Zimek. "Mining Hierarchies of Correlation Clusters". In: *Proc. SSDBM*. 2006, pp. 119–128.

[5] E. Achtert, H.-P. Kriegel, E. Schubert, and A. Zimek. "Interactive Data Mining with 3D-Parallel-Coordinate-Trees". In: *Proc. SIGMOD*. 2013, pp. 1009–1012.

[6] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. "OPTICS: Ordering Points To Identify the Clustering Structure". In: *Proc. SIGMOD*. 1999, pp. 49–60.

[7] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. "The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles". In: *Proc. SIGMOD*. 1990, pp. 322–331.

[8] S. Brecheisen, H.-P. Kriegel, P. Kröger, and M. Pfeifle. "Visually Mining Through Cluster Hierarchies". In: *Proc. SDM*. 2004, pp. 400–412.

[9] M. M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander. "OPTICS-OF: Identifying Local Outliers". In: *Proc. PKDD*. 1999, pp. 262–270.

[10] M. M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander. "LOF: Identifying Density-based Local Outliers". In: *Proc. SIGMOD*. 2000, pp. 93–104.

[11] T. Caliński and J. Harabasz. "A dendrite method for cluster analysis". In: *Communications in Statistics* 3.1 (1974), pp. 1–27.

[12] R. J.G. B. Campello, D. Moulavi, and J. Sander. "Density-Based Clustering Based on Hierarchical Density Estimates". In: *Proc. PAKDD*. 2013, pp. 160–172.

[13] R. J.G. B. Campello, D. Moulavi, A. Zimek, and J. Sander. "A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies". In: *Data Min. Knowl. Disc.* 27.3 (2013), pp. 344–371.

[14] Y. Cheng. "Mean Shift, Mode Seeking, and Clustering". In: *IEEE TPAMI* 17.8 (1995), pp. 790–799.

[15] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proc. KDD*. 1996, pp. 226–231.

[16] E. W. Forgy. "Cluster analysis of multivariate data: efficiency versus interpretability of classifications". In: *Biometrics* 21 (1965), pp. 768–769.

[17] K. Fukunaga and L. Hostetler. "The estimation of the gradient of a density function, with applications in pattern recognition". In: *IEEE TIT* 21.1 (1975), pp. 32–40.

[18] J. A. Hartigan. *Clustering Algorithms*. New York, London, Sydney, Toronto: John Wiley&Sons, 1975.
[19] A. Hinneburg and H. H. Gabriel. "Denclue 2.0: Fast clustering based on kernel density estimation". In: *Proc. IDA*. 2007, pp. 70–80.
[20] A. Hinneburg and D. A. Keim. "An Efficient Approach to Clustering in Large Multimedia Databases with Noise". In: *Proc. KDD*. 1998, pp. 58–65.
[21] S. C. Johnson. "Hierarchical clustering schemes". In: *Psychometrika* 32.3 (1967), pp. 241–254.
[22] H.-P. Kriegel and M. Pfeifle. "Hierarchical Density-Based Clustering of Uncertain Data". In: *Proc. ICDM*. 2005, pp. 689–692.
[23] J. B. Kruskal and J. M. Landwehr. "Icicle Plots: Better Displays for Hierarchical Clustering". In: *The American Statistician* 37.2 (1983), pp. 162–168.
[24] S. P. Lloyd. "Least squares quantization in PCM". In: *IEEE TIT* 28.2 (1982), pp. 129–136.
[25] J. MacQueen. "Some Methods for Classification and Analysis of Multivariate Observations". In: *5th Berkeley Symposium on Mathematics, Statistics, and Probabilistics*. Vol. 1. 1967, pp. 281–297.
[26] L. McInnes and J. Healy. "Accelerated Hierarchical Density Based Clustering". In: *ICDMW*. 2017, pp. 33–42.
[27] R. C. Prim. "Shortest connection networks and some generalizations". In: *Bell system technical journal* 36.6 (1957), pp. 1389–1401.
[28] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications". In: *Data Min. Knowl. Disc.* 2.2 (1998), pp. 169–194.
[29] J. Sander, X. Qin, Z. Lu, N. Niu, and A. Kovarsky. "Automatic Extraction of Clusters from Hierarchical Clustering Representations". In: *Proc. PAKDD*. 2003, pp. 75–87.
[30] J. Schneider and M. Vlachos. "Fast Parameterless Density-based Clustering via Random Projections". In: *Proc. CIKM*. 2013, pp. 861–866.
[31] J. Schneider and M. Vlachos. "Scalable density-based clustering with quality guarantees using random projections". In: *WIREs DMKD* 31.4 (2017), pp. 972–1005.
[32] E. Schubert, A. Koos, T. Emrich, A. Züfle, K. A. Schmid, and A. Zimek. "A Framework for Clustering Uncertain Data". In: *Proceedings of the VLDB Endowment* 8.12 (2015), pp. 1976–1979.
[33] E. Schubert, J. Sander, M. Ester, H. Kriegel, and X. Xu. "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN". In: *ACM Transactions on Database Systems (TODS)* 42.3 (2017), 19:1–19:21.
[34] R. R. Sokal and F. J. Rohlf. "The Comparison of Dendrograms by Objective Methods". In: *Taxon* 11.2 (1962), pp. 33–40.
[35] J. H. Ward Jr. "Hierarchical grouping to optimize an objective function". In: *JASA* 58.301 (1963), pp. 236–244.
[36] M. Wirth, G. F. Estabrook, and D. J. Rogers. "A Graph Theory Model for Systematic Biology, with an Example for the Oncidiinae (Orchidaceae)". In: *Systematic Biology* 15.1 (1966), p. 59.