OPTICSvis: Visualizing the OPTICS algorithm

Sonja Biedermann* University of Vienna

Christian Permann† University of Vienna

ABSTRACT

OPTICS is a popular and robust density-based clustering algorithm often taught at undergraduate levels whose output often puzzles the uninitiated. To aid in teaching the algorithm, we design and implement a thorough visualization of all aspects of OPTICS, and allow for exploration of different parameterizations and their resulting output in the browser. We then evaluate user responses collected while using the tool.

1 INTRODUCTION

OPTICS is a popular and robust density-based clustering algorithm for spatial data. Although popular, the output generated by OPTICS has a tendency to appear somewhat cryptical—instead of generating a straightforward mapping of points to cluster-IDs, it outputs a list of points adjoined with meta data. This output format holds not one, but many clusterings which still need to be extracted.

This is done by picking a *cutoff*, which then separates all points that fall below this value into a cluster. As this is not easy to conceptualize, the output of the algorithm is usually visualized using a bar chart—as such, it can be argued that visualization is already a core part of working with OPTICS, and leads us to our motivation to implement a more sophisticated and interactive visualization design.

2 MOTIVATION

OPTICS is often taught to undergrad students among a group of other clustering algorithms such as k-Means and DBSCAN. A distinguishing feature of OPTICS among this group is that the output is not yet a clustering, but rather an intermediary output that still needs some parameterization—called the *cutoff*—before yielding a clustering that can be used to e.g. produce a colored scatter plot of the input data.

This parameterization is oftentimes puzzling to students. How is this value picked? The real-world answer to this is often to just try out a few values and observe the resulting clustering. Although some gut feeling as to picking this value develops in routine users of this algorithm, it is still hard to pass onto students, who are often rather reluctant to spend considerable time experimenting with algorithms on their own.

We want to offer a visualization dashboard that allows students to interactively try out the algorithm on data sets of their choice. Both the cutoff and the two other important parameters, ε and *minPts*, can be set and their effect observed. Lastly, we also introduce a dendrogram view that is relatively uncommon when dealing with OPTICS, as it is not an hierarchical algorithm per se, but still capable of producing hierarchies of clusterings that might be worthwhile to study. The hierarchies are a direct result of choosing different cutoff values and thus also serve to motivate the meaning of differing cutoff values.

*e-mail:sonja.biedermann@univie.ac.at

†e-mail:a01463926@unet.univie.ac.at

2.1 Background information

What follows are a few facts that are either needed to understand the rest of the paper or are distinguishing general characteristics of the project, such as tasks or user target groups.

2.1.1 OPTICS

As already discussed, OPTICS is a density-based clustering algorithm that works on spatial data. Density-based clustering algorithms typically classify points into either cluster points (possibly giving them a cluster ID) or noise, which is made up of points that belong to regions that are not dense enough to be considered clusters. OPTICS takes two parameters *minPts* and ε.

More concretely, OPTICS computes its output representation by jumping between points (the choice of the first point is left in the open) and performing ε-neighborhood queries. The ε parameter supplied to the algorithm is used to define the neighborhoods. Note that ε may very well be set to an outrageously high value, but will then result in an algorithmic complexity of $O(n^2)$ as every neighborhood query would likely return the entire data set.

The influence of the *minPts* parameter on the result is twofold: first, the *core distance* of a point is defined as the *minPts*th smallest distance to one of its neighbors. Secondly, a point that fails to have *minPts* neighbors is considered to be a non-core point, which is regarded as unreachable by the algorithm and will thus certainly be labeled as noise later on. The reachability distance of a point p, coming from a point o, is the maximum between the core distance of p or the distance between o and p [13].

The output is thus composed of the order in which the points were visited, as well as the reachability distance of each point. This can be nicely summarized using a bar chart, which is then called a reachability chart in OPTICS lingo.

OPTICS is often regarded as an extension of DBSCAN.

2.1.2 Data

Our visualization dashboard is capable of dealing with scalar, two dimensional data. We provide both predefined data sets that showcase the very robust behavior of OPTICS, e.g. by having data sets that form circles or spirals, objects that would not be correctly classified by the likes of k-Means and DBSCAN, and a text input field that allows the user to input her own space-delimited data.

Although higher dimensional data is not supported, more dimensions can be supplied and then two dimensions chosen. The other dimensions are not considered. Albeit being limited, this still allows to explore a higher dimensional data set in slices, and dimensionality reduction can be done using external tools if so desired.

2.1.3 Users

Our visualization design has a strong focus on education. As such, our targeted user groups are educational staff, such as lecturers, students (mainly undergrad level), but also researchers that have a more casual need for using OPTICS or may be casually evaluating the algorithm for further use. Our application is not designed to work well with larger data sets, which we consider non-casual use, although non-interactive elements are still functional—interactive elements, however, often rely on interaction patterns such as hovering over bars or points, which will become unfeasible if there are many bars or points. These aspects are, however, geared towards the

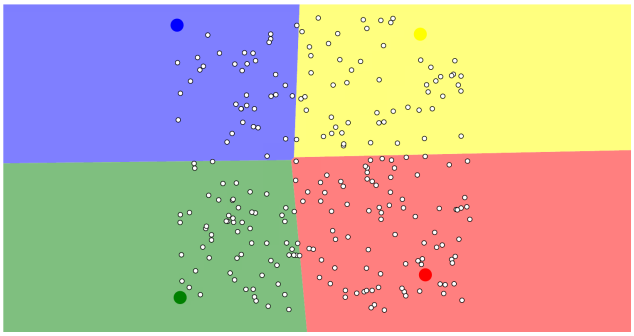

Figure 1: A *k*-Means visualization

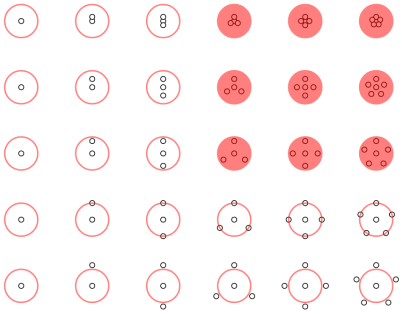

Figure 2: A DBSCAN visualization

educational experience that a student may have; and students are unlikely to want to figure out how an algorithm works by loading up a large data set. It is also of note that the realized product is a browser application and is thus severely limited in terms of computational power.

2.1.4 Tasks

As already mentioned, the main aspect of our visualization is an educational one. Thus, the main tasks include both aiding presentations in the classroom as well as helping students understand the algorithm and its output. An additional, but minor, task is allowing researchers to experiment with the algorithm by using the tool.

3 Related Work

Since the clustering of data is a fairly interesting, current but also hard problem, a few clustering visualizations have cropped up, most of them taking an educational angle. However, some also fulfill other tasks, such as allowing users to verify clustering output.

3.1 Visualizations

Harris [5, 6] implemented two visualizations for DBSCAN and k-Means, respectively. Both implementations display the input data (which can be chosen from a few predefined, but not very exciting, data sets) and overlay information derived from the algorithm, e.g. by coloring the points or displaying the ε neighborhood as an overlaid circle.

The *k*-Means visualization [6] allows to choose from three different centroid picking strategies—user defined, random or farthest from each other—and then allows the user to place as many centroids as she pleases. The rest of the visualization consists of pressing a button until the algorithm converges and watching the centroids move along with their regions. Figure 1 shows a screenshot.

The DBSCAN visualization [5] is similar. To illustrate the meaning of both the ε and *minPts* parameters, the author superimposes a grid of circles with radius ε and colors them red if they overlap more than *minPts* points. Figure 2 shows a screenshot.

ClustVis [10] is a tool for applying the Principal Component Analysis (PCA) method to user-defined data, and allows for generating and exporting both scatter plots of the resulting two dominant components, as well as heat maps. It allows tweaking the output by choosing different components, inversion of axes, as well as row scaling and different PCA methods, among others. The clustering occurs by correlation distance and average linkage. Figure 3 shows the resulting heat map of the default data set, with the dendrograms derived from the clustering process shown on the left and top. There is little explanation of the methods on the tool site, but altogether it seems to be a tool aimed at experts.

Clustervision [8] differs from the other tools by the fact that its aim is to enable users to verify the clustering output—as put by its tag line: *visual supervision of unsupervised clustering*. Clustervision

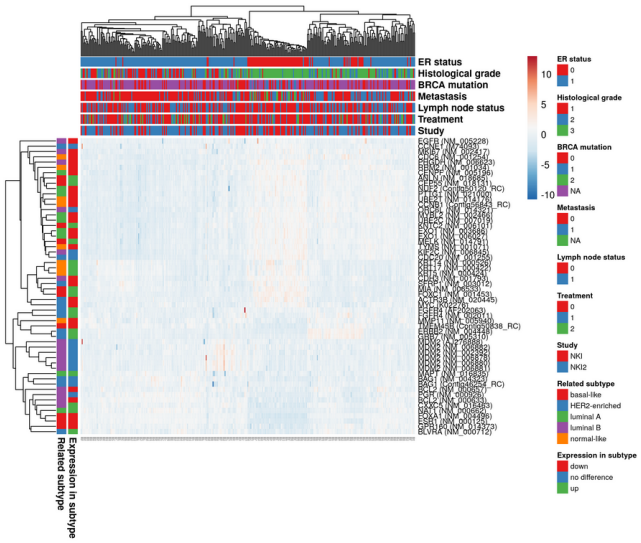

Figure 3: ClustVis heat map

allows comparison between many different clustering algorithms (among them are DBSCAN, Spectral Clustering, *k*-means and others). The authors argue that clustering is often done in an exploratory fashion—trying out, more or less blindly, different algorithms with different configurations. Any algorithm may provide a new insight into the data set—some algorithms may provide some insights that others fail to uncover. Thus, they propose a system that ranks different clustering results and urges the user to compare the solutions and to ask new questions. The tool looks to be excellent and very sophisticated[1], however we were unable to try it out beyond reading the paper.

3.2 Libraries

Our implementation is driven by the excellent d3 library [4], using multiple plugins. For lassoing the overlaid points on the density chart, we used d3-lasso [7]. The density estimation and contour computation is done by d3-contour [3]. The legend of the density chart is generated by d3-legend [9]. An important source of inspiration and know-how was `bl.ocks.org` [2], the corresponding sources are cited in the source code wherever relevant.

We have also used jQuery v3.2.1 [12] as well as Lodash v4.17.4 [11].

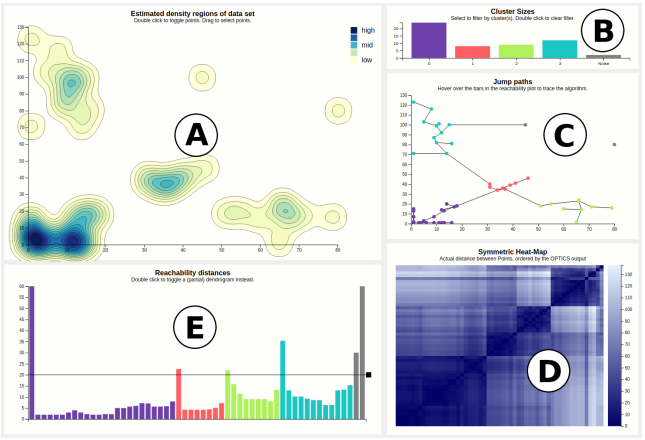

Figure 4: An overview of the implemented solution. (A) Density regions estimated from the input, (B) cluster sizes, (C) chart showing the path that the algorithm took, (D) heat map displaying inter-point distances, (E) reachability chart.

4 Approach

We split our work on the visualization design into two parts: first, we designed and implemented the static views, making sure that even without interactivity, they display useful information—providing an overview. Only afterward we added filtering, brushing and zooming to get at some details that are hidden in the overview or provide value during presentations in an educational setting (e.g. a classroom).

4.1 Views

Our visualization design encompasses 6 views showing either the input data or different aspects of the output data. The five main views which are visible from the start are shown in Figure 4. We will reference parts of this figure by the letters shown from here on.

Perhaps most central is the reachability chart (E), which is the staple bar chart that was proposed as the go-to visualization method by the authors of OPTICS [1]. We have extended this view to contain an interactive bar that corresponds to the cutoff value. Changes to this bar result in a recalculation of the output clustering. We have also introduced a second cutoff to separate sub-clusters from their super-clusters (e.g. even denser regions in an already dense region), however this was later removed due to unsatisfactory user feedback.

The top left view shows a density map of the input data (A). This serves as a reminder of what the OPTICS algorithm deals with—densities—and shows the general shapes of the data. The density map is estimated using a Gaussian kernel and will thus show circular tendencies that may not fit the data perfectly.

Going clockwise from this view, the next view shows a bar chart that summarizes the clustering by the cluster sizes (B). Very unbalanced values may be indicative of a bad configuration.

The next view visualizes the path that the algorithm took during execution (C). This is useful for tracing the algorithm on small data sets (e.g. when doing homework on the algorithm) and shows the decisions that the algorithm made during a run. Changes to the paths taken when changing the configuration do not seem significant as to the quality of the configuration.

The heat map (D) that is next is a rather novel way to help the user find the most natural clustering of the given data set as well as show hierarchical clusters. For this visualization, we apply the data set to both axes in the order that is given by the OPTICS algorithm. For each pair $(p_i, p_j) \in DB \times DB$, where DB is the data set or data base, we compute the distance $d(p_i, p_j)$ and map it to a hue, with darker hues bound to smaller distances. Note that this is the actual distance between points (given by e.g. the Euclidean distance function), and not the reachability as defined previously and shown in the reachability chart (E).

If the data set has a clustering structure that is apparent with the given configuration of ε and *minPts*, the clusters will become apparent as squares of darker color. Figure 4 (D) shows this for a small data set. Other characteristics are also visible—such as having circular objects in the data set, which will appear as diagonal lines in the heat map. Although this information is also visible in the reachability plot—where clusters appear as valleys—we have noted that the squares are easier to grasp for novices, and squares seem to generally be easier to perceive especially with sub-cluster structures, which will appear as nested groups of squares. This view supports a square brush which can be used to select regions of the data set and will highlight the corresponding data points in the reachability chart.

A particularly bad choice of parameterization will also become apparent in the heat map, as it will appear rather erratic.

The final view is hidden initially. By double clicking the reachability plot, a large dendrogram view will appear. Due to constraints in visualization estate, this dendrogram is flipped on its size and is also not technically a dendrogram—but rather a general tree, as leaves are not forcefully pulled to the last level.

A dendrogram is without doubt a good choice when visualizing hierarchies, which is our aim. However, it should also be noted that for large data sets, a dendrogram does not scale, and analyzing large hierarchies will quickly turn into an ordeal. Possible mitigations include e.g. collapsing parts of the tree or zooming, but the original problem still remains. Such was also argued in [1], where the authors make a point of highlighting that the reachability chart is vastly superior to tree-like outputs of other—usually hierarchical—algorithms. We want to point out that OPTICS is *not* a hierarchical algorithm, but merely *also* mirrors the hierarchical structures of the input in its output. Hierarchical clustering algorithms typically work by recursively merging (bottom-up) or splitting (top-down) a trivial clustering. OPTICS does not show this behavior.

The question is: how to source the hierarchy given the output, if it is not just given by how the algorithm does its clustering? We have chosen a rather simplistic approach, consisting of descendingly sorting the algorithm output by the core distance of each point, and then for every distance, extracting the clustering that results when setting the cutoff to this distance. If this clustering contains at least one cluster ID that is not present in the last clustering (if any), then some cluster must have split—and the current clustering is the next hierarchy level. We completely ignore points that get assigned noise status. However, since points drop out of the clustering, we may overlook clusterings that are significantly different, but have as many as or less clusters than the previous. Thus, the resulting hierarchy is only approximate, but a few radically different clusterings ("cornerstone clusterings") are guaranteed to be contained, and each lower hierarchy level is larger than the last, something that is always given when working with hierarchical clustering algorithms.

4.2 Interactivity

All views have interactive abilities.

The the actual points of the density regions chart (A) can be toggled by double clicking on the canvas. Doing this will show the cluster IDs of the points mapped to a color, and also allow the selection of points using a lasso, resulting in a highlighting of the corresponding bars in the reachability chart (E). The corresponding bars of the reachability chart will then be highlighted. This is a means to substantiate how each bar of the reachability correlates to a point in the data set to be used during e.g. in-class discussion of the algorithm.

The cluster sizes overview (B) can be used to filter the reachability chart. Hovering over a bar in the reachability chart will show the

[1] https://vimeo.com/232177941

ε-neighborhood of it in the density chart (if the points are toggled), as well as highlight the corresponding edge in the jump path chart (C). Zooming is available in both the heat map view (D) and the jump path chart. The heat map also has a square brush that can be used to select point groups, which will then be highlighted in the reachability chart. Selecting a level of the hierarchy tree will apply the corresponding cutoff value to the main visualization.

Most of these interaction patterns have the goal of explaining and providing context to the reachability chart, which is the chart that is most commonly associated with OPTICS, and which most newcomers to the algorithm are confused by. Others, e.g. zooming into the jump path chart, allow to see data that might be hidden by other overlapping data, or to filter down on particularly interesting clusters, particularly useful if the data set is quite large and the hovering actions on the reachability chart are no longer feasible if the whole data set is displayed.

When toggling the Settings tab for changing the configuration of the algorithm a scented widget is displayed, which shows the ratio of cluster points to noise. This can help identify a nonsensical configuration (e.g. one that results in a lot of noise) before the changes are committed (which triggers a rather expensive calculation along with an even more expensive re-rendering).

5 Implementation

The dashboard is implemented using JavaScript and the fantastic d3.js SVG and DOM manipulation library. The application is entirely client-side and only needs a file server to run. A current version is also available online[2].

The project also contains a complete implementation of OPTICS which is slightly tweaked to generate information used in some charts (e.g. the jump paths). This has the downside that our implementation might be a bit slower, although in general, the bottleneck seems to be the SVG rendering in the browser.

The application was tested on somewhat recent versions of Google Chrome and Mozilla Firefox and found to be in good working order in both. However, we noted that Firefox performs noticeably worse with regard to SVG rendering, so much so that it actively struggles loading some data sets. This seems more pronounced on Windows, whereas on Linux the issue is less pronounced (although still significantly worse than Chrome). We thus dearly recommend using it in Chrome and hope that Firefox will get its rendering act together sometime soon.

5.1 Challenges

One of the main challenges the implementation itself faces is the terrible performance of SVG rendering which is especially bad in some browsers. There are quite a few SVG elements to be rendered, especially so with larger data sets, which bogs down the entire application.

Our home-baked OPTICS implementation is also quite naïve and thus not all that quick. This is however overshadowed by the performance issues of SVG. The browser would go kaput before getting close to large enough input sizes.

Another issue was with forcing the heat map's brush to stay square and extracting a hierarchy from a given OPTICS output.

6 Results

Our application is to be used in conjunction with either already existing knowledge about clustering (especially density based clustering) or a classroom situation where somebody with this knowledge is present to further elaborate the algorithm.

We were unable to find another visualization that provides a similarly thorough visualization of OPTICS.

[2] https://biederfrau.github.io/opticsvis/app/index.html

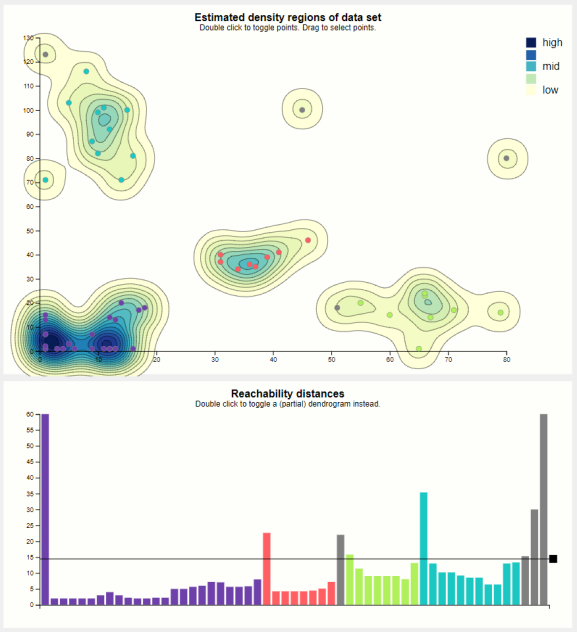

Figure 5: The resulting clustering when changing the cutoff

6.1 Scenario of Use

A scenario in which a lecturer at a university may use our implementation may look something like the following.

First the lecturer could show the default view, that appears when loading the tool, as shown in Figure 4, to his students. Here a good parameter set is pre-selected and we can clearly see the cluster structure on the data. The actual problem of clustering can be presented here as well as the specifics about density based clustering in comparison the methods like k-Means, that have probably been discussed in an earlier lecture.

The next logical step is now to talk about the estimated density, Figure 4 (A), regions of the data set where it is somewhat visually obvious where the clusters are. An interesting question for the students would now be if they think, the left- or rightmost point in the bottom right cluster should still be part of the cluster.

Moving the cutoff bar in the reachability plot, as shown in Figure 5, around, reveals that the left point is removed first when moving the cutoff bar down.

This can now be explained by hovering over the bars of the reachability plot and discussing how the reachability distance relates to the output of OPTICS, which is usually just this reachability data.

While moving the cutoff bar around it might also be interesting to see how it affects cluster points, Figure 4 (B), changing into noise and splitting clusters into smaller ones. A general overview might be had when looking at the cluster sizes plot.

Whenever needed the lecturer may also want to use the filtering in this plot to focus a cluster or highlight something using the lasso in the density plot or the brush in the heat map.

Explaining the algorithm step by step is supported by the jump paths plot, shown in Figure 6, where the lecturer can follow each jump, by moussing over the corresponding bar in the reachability plot and zooming in the jump paths plot. An important point to make here is that the reachability value in the corresponding chart is not necessarily equal to the actual distance between the two connected points. It is important to emphasize the influence of the *minPts* parameter on this value.

Next a general discussion may be had about what a good clustering actually is. For this it may be a good idea to look at the heat

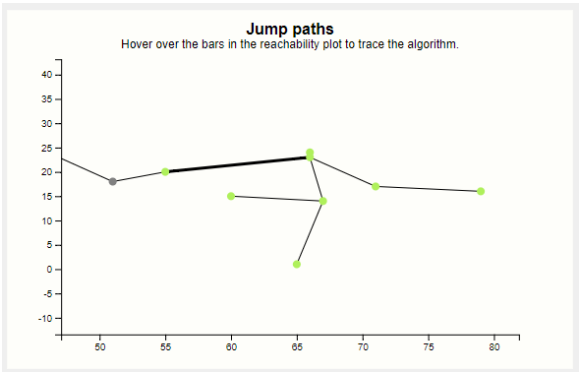

Figure 6: The jump paths plot, showing the steps the algorithm takes

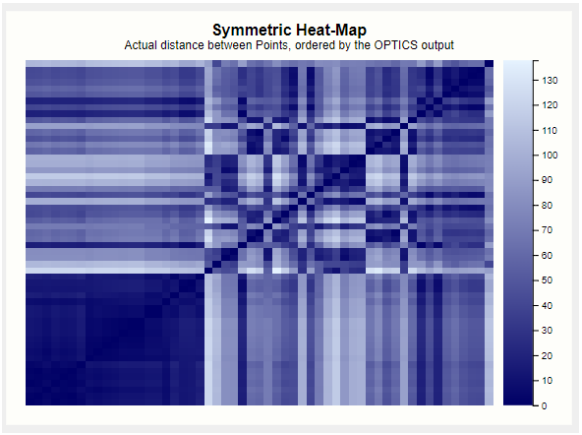

Figure 7: Heat map resulting from a bad configuration

map, as shown in Figure 4 (D).

A good clustering will usually produce a nice heat map, because points close to each other in the data space will also be close to each other in this plot and therefore produce a rectangle or other easily recognizable shapes. Here the lecturer can nicely show how a bad parameter set, as shown in Figure 7, may produce a bad result.

With those parameters, no choice of cutoff will produce a satisfactory clustering, unless this degenerate result was somehow desired.

At this point the lecturer can explain, that we may still want a result like this, e.g. as in Figure 8, if there is reasoning behind it—like explicitly not wanting clusters with less than 8 points reachable within the distance 10. This might throw out unwanted minor clusters and break weak links between clusters. Here would also be a good moment to talk about how reducing ε can improve the run time of the algorithm greatly.

When the parameters were chosen well and the lecturer wants to visualize and explore different clusterings, our dendrogram, as seen in Figure 9, can be toggled.

Clicking onto a column and leaving this view allows for quickly changing the cutoff to useful settings. The cutoff can then be adjusted a little bit to achieve the right filtering of noise points in the data.

6.2 Performance

We now evaluate both the computational as well as the visual performance of our application.

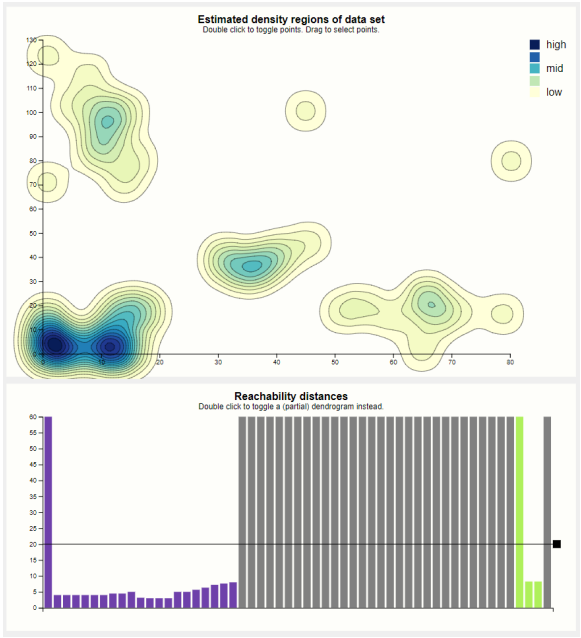

Figure 8: A clustering result that is usually regarded as bad

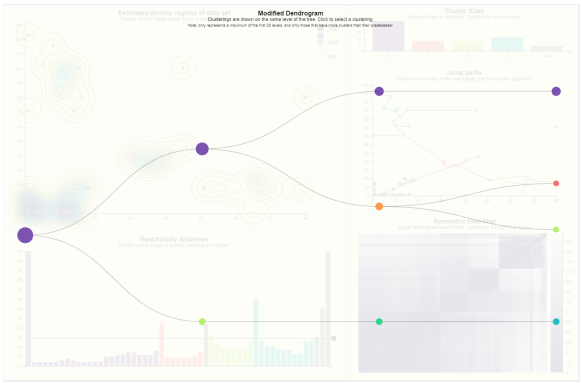

Figure 9: Dendrogram, showing cluster hierarchies

6.2.1 Computational Performance

Computationally our tool is currently not well optimized. Since we implemented OPTICS ourselves we took a trivial approach and the run time complexity for bad input parameters may reach $O(n^2)$. For a more sophisticated implementation a server backend would be useful as well as better optimized code.

A drawback of this is, that with current hardware working with a data set larger than 400 points will result in stutter when interacting with different aspects of our tool.

6.2.2 Visual Performance

We found that there are a few problems with our visual performance. If there are many points, they may overlap a lot in the plots. It is possible to work around this by zooming in the jump paths plot and looking at parts of the data at a time. Another possible problem could be, that the reachability plot may not be very nice anymore since the bars would be to thin to properly distinguish.

Other parts of our implementation should still work well though. The density plot should still give a good overview of the data, even with lots of points, since it abstracts from single points. The cluster sizes plot should look the same, as long as there is not also a huge

amount of clusters. The heat map should always look good because it gives a hierarchical overview and is zoomable. Shapes should still be recognizable.

6.3 Evaluation and Feedback

The feedback we got consists of the course feedback given by our lecturers and students during our presentation of our prototype as well as from other users we questioned about our reworked prototype.

6.3.1 Course Feedback

The first feedback we got was considered and implemented into our tool.

In the second feedback, we were told, that our heat map shows ugly lines between the actual squares. This was due to rendering artifacts from the browser and was easily fixed by changing CSS shape rendering property.

Another complaint was about the highlighting in the density plot. We have improved the highlighting behavior and don't think this is a problem any more. We may be misunderstanding this feedback though, it is possible that what was actually meant was that the highlighting does not work when the points are not visible. We want to keep this behavior, because the user would otherwise have no idea where his highlighting comes from or be generally confusing.

The last point made was questioning if our implementation visually scales well with large/realistic data sets. We think that it would, up to the point where the reachability distances plot cannot be properly read anymore because the bars get too thin. Another problem could be that the cluster sizes plot may get too crammed when there are too many clusters. We targeted our implementation to the educational aspect and feel that those problems are negligible because if someone wants to cluster a big data set they would want to use an optimized implementation anyway, and not a tool in the browser.

6.3.2 User Feedback

For user feedback we asked both users with and without knowledge on hierarchical clustering and OPTICS to evaluate our tool.

For the users with background knowledge, we decided to let them try out our tool and not interfere during this process, since we wanted to recreate the setting of someone that wants to get to know this algorithm. Some of those users told us afterwards that the second cutoff line was not in line with their expectations—usually, only one cutoff is present when extracting clusterings. Even after explaining that the secondary cutoff bar should serve to show hierarchical substructures, most users remained skeptical and argued that the way those substructures were shown—as dashed lines within the corresponing cluster size bars—was weak and hardly noticeable. This criticism is fair and thus it was gone.

A frequent question we were asked was how the density estimation works, especially by our one domain expert user. It may thus be a good idea to give additional information to some details of our implementation in a read-me or wiki page.

We also noted that some users did not use all of the possibilities of interaction available. When asking them why, the answers were that they did not know they were possible, like zooming in the heat map, or that they did not try them out because they felt that they would not need them, like filtering by clicking in the cluster sizes plot. It should be noted that most users were testing the tool on a rather small data set, and filtering might only become relevant on larger ones.

Besides that, we got positive feedback on the looks of the tool and its usability. The performance was somewhat of a concern for the users. Some users did not like how the fan of the computer turned on when they were using the tool.

For the other users, we changed the setting and decided to give them a very brief explanation on density based clustering and OPTICS first. We did this to create a teaching scenario, and when we let them work with our tool we proposed trying specific settings and explained the different plots. When the users felt that they understood our tool we asked them if they felt that the tool made it easier to understand the algorithm and if so, why. All of them said that the tool was helpful.

The estimated densities plot was usually mentioned as the most useful, because it gives a good idea of the clustering without further investigating the how or why. The dendrogram was also thought to be useful, the users especially liked being able to select a clustering from there. We were also told, that moving the cutoff around intuitively teaches how points get assigned to clusters.

We also asked them what they did not like and were told different things. One person argued that the density regions plot and the jump paths plot could be drawn as one plot. We decided to keep it as is though, because joining them together may cause too much information overlapping, which may be confusing.

Another person said, that the cluster-noise ratio widget is not very useful. We somewhat agree that if does not give the best overview of what to expect from the new parameters, but we think that it still gives an idea on how drastic the changes will be. Lastly some users would have liked more default data sets. For that reason we added a few new sets with interesting patterns that showcase the algorithm.

7 Discussion

We will now discuss both strengths and weaknesses of our solution, as well as summarize the most important things that we learned during this project.

7.1 Strengths and Weaknesses

The biggest strengths of our tool include allowing for quick exploration of cluster hierarchies in small data sets and helping with learning and teaching about the OPTICS algorithm and density based clustering in general. We think, that it also helps understanding if the algorithm is fit for the problem the user aims to solve, especially by showing additional meta information that may give additional insights. (dendrogram/heat map) In addition our Default data sets give a nice overview over the capabilities of OPTICS for other data sets.

Where our tool may be lacking is the ability to learn about how OPTICS works without any knowledge about density based clustering. It may also not be showing enough information when working with multidimensional data and it could therefore be difficult to understand the different plots. Also, our tool is slow when working with large data sets, which makes it less useful for scientific use.

7.2 Lessons Learned

During this project we learned different new things.

On top of what we learned about the d3 library in the visualization class, working with it on a custom project where we had to realize our own ideas made us dive deeper into its uses. This taught us how to work with it, what it is capable of as well as what it is not capable of and how to overcome those instances.

We learned a lot about creating nice visualizations, not only through the feedback we got, but usually from ourselves, when we did not like how something looked. Reworking what we did not like then showed us how not to visualize some things, which we will be able to apply in later projects.

We also learned about teamwork, especially because our team coordinated very well. We worked with the versioning tool git and besides sometimes needlessly having to merge code snippets, we think our task separation and joining of code parts worked flawlessly.

Task	Sonja	Christian
Milestone 1		
Website (Content)	85%	15%
Idea	15%	85%
Solution Sketch	50%	50%
Milestone 2		
Chart explanation	90%	10%
Mockup 1	0%	100%
Mockup 2	0%	100%
Mockup 3	100%	0%
Visualization Techniques	100%	0%
Conclusion/Scenario of use	0%	100%
Milestones	70%	30%
Milestone 3		
Skeleton with Settings	90%	10%
Charts (non interactive)	40%	60%
Filter-Interaction	50%	50%
Zoom/Pan-Interaction	50%	50%
Highlight-Interaction	60%	40%
Algorithm	10%	90%
Web summary	50%	50%
Milestone 4		
Dendrogram	100%	0%
Bug fixes	40%	60%
Additional Data-Sets	50%	50%
Paper	50%	50%

Table 1: Separation of tasks

8 Separation of Tasks

Table 1 details how the work during the project was split between us. In general, the work was split evenly and fairly with each of us trying to focus on things that suited us better.

9 Conclusion

We designed and implemented a thorough visualization of the OPTICS clustering algorithm to be used in an educational setting. We provided a brief introduction to the matter, elaborated all views and their interaction patterns and evaluated the finished application prototype using feedback from users of different knowledge levels who were either computer science students or lecturers.

We acted on most of the feedback and improved upon our past visualization design. Finally, we reflected on the things that we learned doing this group project and provided a description of the separation of tasks.

References

[1] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, June 1999. doi: 10.1145/304181.304187

[2] M. Bostock. `Blocks`. `https://bl.ocks.org`. Last-Accessed: 2018-01-17.

[3] M. Bostock. `d3-contour`. `https://github.com/d3/d3-contour`. Last-Accessed: 2018-01-17.

[4] M. Bostock, V. Ogievetsky, and J. Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, Dec. 2011. doi: 10.1109/TVCG.2011.185

[5] N. Harris. Visualizing DBSCAN Clustering. `https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/`. Accessed: 2018-01-13.

[6] N. Harris. Visualizing k-Means Clustering. `https://www.naftaliharris.com/blog/visualizing-k-means-clustering/`. Accessed: 2018-01-13.

[7] S. Kokenes. `d3-lasso`. `https://github.com/skokenes/d3-lasso`. Last-Accessed: 2018-01-17.

[8] B. C. Kwon, B. Eysenbach, J. Verma, K. Ng, C. deFilippi, W. F. Stewart, and A. Perer. Clustervision: Visual supervision of unsupervised clustering. *IEEE Transactions on Visualization and Computer Graphics*, PP(1):1–1, 2018.

[9] S. Lu. `d3-legend`. `http://d3-legend.susielu.com/`. Last-Accessed: 2018-01-17.

[10] T. Metsalu and J. Vilo. Clustvis: a web tool for visualizing clustering of multivariate data using principal component analysis and heatmap. *Nucleic Acids Research*, 43(W1):W566–W570, 2015. doi: 10.1093/nar/gkv468

[11] J.-P. Sirois and Z. Hall. Lodash. `https://lodash.com/`. Last-Accessed: 2018-01-17.

[12] The jQuery Foundation. jQuery. `https://jquery.com/`. Last-Accessed: 2018-01-17.

[13] Wikipedia. OPTICS algorithm. `https://en.wikipedia.org/wiki/OPTICS_algorithm`. Last-Accessed: 2018-01-17.