Scatterplot Visualization of Hierarchically Clustered Data Points

Daniel Gruncl*
Supervised by: Ing. Ladislav Čmolík, Ph.D.†

Faculty of Electrical Engineering
Czech Technical University in Prague

Abstract

The focus of this paper is real-time visualization of and interaction with scatterplots with hundreds of thousands of data points, where the points are organized into a hierarchy of clusters. We present a technique that automatically selects a color palette for the clusters of the selected level. The level of the cluster hierarchy and the color palette are dynamically adjusted by zooming the scatterplot. Furthermore, the technique improves visibility of the displayed clusters by reducing occlusion of the overlapping clusters. We demonstrate the visualization technique using two real medical datasets containing 2D coordinates of hundreds of thousands points.

Keywords: hierarchical data, cluster visualization, scatterplot, dynamic color palette

1 Introduction

Applications of different clustering algorithms in biomedicine play a key role in advancement of data analysis. Namely, protein-protein interactions (PPI)[9] and protein expression, genomic sequence analysis, MRI image analysis etc., require different cluster analysis and assumptions. Hierarchical clustering as a method of cluster analysis aims to build a hierarchy of similar clusters to identify informative natural clusters of observations, adds additional complexity to this problem. The visualization aims to present the properties of such datasets, that is the hierarchical structure, the density of the clusters and their mutual overlaps as well. This is important for physicians to distinguish substantial overlap in diseases spectrum [1].

The clustering algorithms can be divided into hierarchical and partitioning. Hierarchical clustering algorithms can be subdivided into agglomerative (bottom-up clustering) and divisive (top-down), which perform recursive partitioning. Since the boundaries of the clusters cannot be objectively defined, hundreds of clustering algorithms have been proposed, each with different priorities. Thus, it cannot be said which algorithms are better or worse, as the algorithm's performance is often dependent on the characteristics of the information demanded as well as on the dataset itself.

To present the data, reduction from n-dimensional to 2D or 3D space is needed. Based on the deformation of space caused by the dimension reduction, the dimension reduction algorithms can be divided into linear, nonlinear, and those that have been implemented in both linear and nonlinear variants. PCA is a commonly used linear method that flattens the data along axes of minimal variance. t-SNE, a nonlinear method, aims to separate clusters in the data and avoid overlap of clusters of different categories, which may distort the data as the clusters might have overlapped in the original data. MDS has been implemented both as linear and nonlinear. This method reduces dimension while minimizing distortion of mutual distances between data points. An overview of the clustering and dimension reduction algorithms is provided by Wenskovitch et al. [10].

In this paper, we present our progress on the scatterplot visualization of large datasets of medical observations represented as n-dimensional data points organized into a hierarchy of clusters. Hierarchy is given as a tree, whose leaves contain points in 2D space. The hierarchy subdivides data points into clusters based on their mutual proximity. At the same time, some of the nodes in the hierarchy maintain the assignment of all their child nodes to a certain population. These nodes are disjunctive and provide complete coverage, meaning that every point in the data set is associated with exactly one population. Therefore, while the nodes of the hierarchy signify spatial proximity and clustering in the data points, some of the nodes also categorize the clusters into populations. Overlaps of clusters of different populations signify interactions between the populations.

The challenges of visualizing such observations are the scale of the data, namely the high count of points, causing heavy overdraw, and the high number of categories, which is only amplified by the hierarchical structure of clusters. The next challenge is the overlap of points of different categories, which makes the separation of points difficult. The characteristics of the data also rule out the utilization of position as a visual channel, as the data are represented as a set of hundreds of thousands points, whose original n-dimensional position is projected into 2D using dimension reduction techniques (e.g., PCA, NMF, t-SNE). This leaves us to separate clusters only by color since high point

*gruncdan@fel.cvut.cz
†cmolikl@fel.cvut.cz

count limits utilization of visual channels such as size and shape of points as well. We propose a visualization technique that allows for the analysis of such data. The proposed approach has three main contributions:

1. Real-time visualization rendering, allowing interactive close-up examination of visualized data.
2. Color separability of clusters and their identification in the hierarchy.
3. Identification of cluster density and their overlaps.

2 State-of-the-art

The overdrawing issue can be reduced by data abstraction techniques, such as binning, contouring the density function (that is, converting dense parts of the clusters into areas while outliers are displayed as points), or subsampling of the data.

Heimerl et al. [5] designed a technique of binning data points into a honeycomb grid, with each cell containing a bar graph or pie chart representing the distribution of individual categories in the cells. The background of the cell can also be saturated to communicate the density of the data points. However, the often complex structure of clusters in our data and rapid changes in cluster density, would require a high number of cells to keep the visualization representative. Breaking continuous areas of single color into a number of small icons, coupled with the much higher number of categories than Heimerl et al. [5] planned for, would lead to visual clutter and decrease the discernability of categories significantly.

Chen et al. [4] in their article demonstrate subsampling by sampling a density function, reducing overdrawing while preserving density information and relative representation of data categories compared to original data. Visualization by contouring the density is also provided, but overlapping colored areas causes color mixing, thereby significantly reducing the discernability of colors while losing density information of as well. However, sampling a density function seems feasible and may be employed in further developments of our work.

The color separability of many categories can be improved by using a dynamic color palette, which takes advantage of situations where only some categories have significant representation on the screen. Such a technique was developed by Waldin et al. [8]. The technique, named Chameleon, also provides hierarchical subdivision of the color space, which too is desired for our purposes.

Chameleon is designed for coloring the internal structures of viruses and cells. Here, at the highest level of the hierarchy are the structures of the virus, such as the lipid envelope and the capsid, in which the individual proteins can be distinguished after zooming in, as well as the domains and the atomic structure of the proteins. When gradually zooming in, only one or a few structures from higher levels of the hierarchy can fit on the screen, so there is no need to divide the color space among all of them but only among the visible ones. Thus, for each structure on a higher level, a larger part of the color space can be allocated, which can be further divided to be allocated to lower structures, as can be seen in Figure 1.

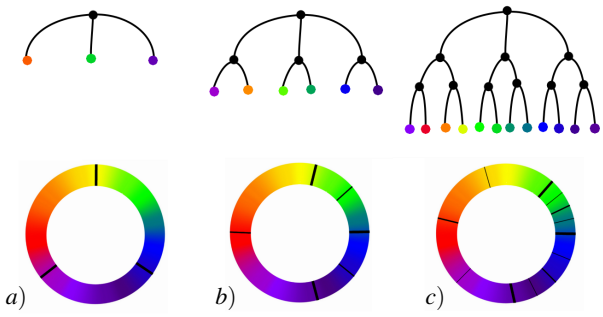

Figure 1: Hierarchical subdivision of color when zooming in on certain cluster. a) Color wedges are allocated only for the top level of hierarchy. b), c) With increased zoom, lower levels of hierarchy are visualized with distinct colors. The size of the color wedges corresponds to the hypothetical relative representation of individual clusters and subclusters on the screen.

3 Our Approach

We have decided to visualize the data as-is, without abstraction or resampling. A dynamic color palette with hierarchical coloring is employed to improve cluster discernabilty, while rendering points with transparency will allow for visualization of density and overlaps of the clusters. The visualization is GPU accelerated to achieve interactive real-time rendering.

3.1 Dynamic color palette

The method of coloring data points is based on the dynamic color palette developed by Waldin et al. [8], but with regards to the different characteristics of the data, several modifications have been made. The first modification is given by the fact that the data are individual points, not forming distinguishable continuous objects such as proteins. Therefore, it is impossible for the color spaces of individual hierarchy nodes to overlap, thereby reducing the space available to nodes at lower levels of the hierarchy. Furthermore, the number of hierarchy levels can be greater than it is in the case of visualizing a virus or cells. Therefore, the dynamic palette method [8] is applied at all levels of the hierarchy, except for the last one, where we do not have to consider the needs of any subclusters, and thus. maintaining color discrenibility remains the only priority. The color space is thus – in correspondence to the hierarchy – recursively divided into smaller and smaller sections (Figure 1).

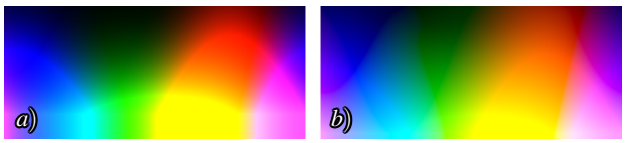

Figure 2: a) HCL implementation by Zeileis et al. [11]. b) HCL implementation from Catalano framework [3] satisfies the property of isoluminance better, notably in the dark green part of the spectra.

To prevent one dominant cluster from taking up all the color space, Waldin et al. [8] have imposed limit on the maximal size of the color wedge allocated to a single cluster. This way, the maximal size of one single color wedge at the k-th level of the hierarchy is a^k, where a, ranging from 0 to 1, is the size limit. To maintain the distinguishability of colors at lower levels of the hierarchy, the size limit a was set to a value higher than recommended by Waldin et al., namely 0.75 instead of 0.5. Also, because the number of subclusters may vary significantly across clusters, the size limit may be too high for some clusters with only a few subclusters. Thus, we have imposed an additional size limit at the top level of the hierarchy. The limit is the number of descendants n_i of a i-th cluster at the top level of the hierarchy multiplied by an appropriately chosen constant 0.08. In conclusion, the maximal size for cluster at k-th level of the hierarchy being descendant of i-th top-level cluster is

$$min(0.08n_i, 0.75) \cdot 0.75^{k-1}. \quad (1)$$

The original method sets size of color wedge allocated to individual clusters proportionally to the number of subclusters visible on the screen. In our case, even at a high level of zoom, most subclusters are present on the screen due to the wide scattering of the clusters. This results in almost no color space being released when zooming. Therefore, we have decided to size the color wedge proportionally to the number of points in the given cluster located on the screen. Allocating by the number of points is very aggressive, which achieves a good distinction of points according to their belonging in the hierarchy. On the other hand, in certain situations, the assigned colors may change significantly when the view changes, possibly confusing the user. The limits imposed on the maximal size of color wedges reduced this artifact. Other applicable methods are normalizing the number of points in individual clusters, weighting by the square root of the number of plotted points, or blending uniform and weighted color allocation.

3.2 Color model

HCL is a family of color models having channels hue, chroma and luminance. HCL models are isoluminant and perceptually uniform and thus often used in computer visualizations [8][4][7]. We have used the implementation of the HCL color model developed by Zeilies et al. [11], which contains HCL-to-RGB conversion chain. Part of the chain is XYZ-to-RGB conversion, where we have used Catalano's implementation [3] instead, since in our experience it better satisfies the property of isoluminance (see Figure 2). The HCL color model is implemented as CIELAB with axes $A*$ and $B*$ transformed to polar coordinates. The hue channel is used to differentiate the clusters, as it is the only one that does not visually imply sorting.

Figure 3: Top line: coloring using hue only. Bottom line: coloring using hue and brightness.

But, since the first level of hierarchy of visualized data contains over twenty nodes, comparing to the data, for which the dynamic color palette developed by Waldin et al. [8] was designed, where first level of hierarchy only contains up to ten nodes, the hue channel alone proved to be insufficient to reliably distinguish clusters of data points (see Figure 3). This is also due the used model CIELAB not being perfectly hue uniform [6] and in the area of the blue hues, distance between distinguishable hues is greater than it is for the other color hues.

Therefore, two channels, hue and luminance, are used in coloring the first level of the hierarchy. The chroma channel is not utilized because it interferes with the hue. Hue is used as the primary channel, where each group uses a different hue. Luminance is used as a complementary channel to avoid blending in with the background. Because the technique of color optimization used by Waldin et al. is only designed for optimizing color palette in one dimension, a low number (five) of fixed brightness levels is used, which are assigned to successive nodes in the cluster hierarchy. The order of the used luminance levels is permuted so that adjacent sections in the hue channel are not assigned adjacent luminance levels, thereby increasing the tonal distance of the colors assigned to successive clusters. All nodes in the hierarchy are colored using the same luminance level as their parent. The use of two channels improved the separability of clusters, as can be seen in Figure 3.

3.3 Data density

Visual encoding using the luminance channel excludes the rendering of points using transparency, because in the case of a white[1] (achromatic in general) background, luminance interferes with transparency, that is, the difference in the brightness of the colors of individual clusters is not discernible; darker clusters only look sparser. Also, when transparency is used, colors will be mixed, which will worsen the distinguishability of individual clusters, and new colors may even appear. Therefore, in order to

[1] Visualization uses black background by default, as it makes clusters stand out better, white background mode was added for print.

preserve the distinctness of the individual clusters, the rendering is performed without transparency.

However, this introduces two new problems: the dependence of the visualization results on the rendering order (Figure 4a, b), which distorts the information about clusters overlap and loss of the data density information.

The rendering order issue was resolved by performing a depth test and assigning a random depth to each point. This way, the overlap of the groups depends on their density; a denser group has a statistically greater chance that out of the number of points falling within one pixel on the screen, its point will have the smallest depth of all and thus will be displayed over the others. This technique also achieves proper visualization of cluster overlaps without mixing colors. Figure 4 compares the differences on two overlapping clusters, blue one linearly falling off in down direction and brown one linearly falling off in left direction.

We will now demonstrate that the probability of a cluster being on top in a given pixel corresponds to its relative representation at the pixel. The depth of cluster X at pixel p contributing to the pixel with l samples is the minimum of l samples of uniform random variable with range 0 to 1. The probability of one such sample being lower than $x \in \langle 0;1 \rangle$ is x. Therefore, probability of l samples being lower than x is x^l, which is the CDF function of cluster depth at the pixel. Thus, when cluster X is contributing to a pixel with l samples, and all other clusters (Y) combined contribute with k samples, probability of X being on the top is calculated as difference between two random variables with CDF functions

$$F_X(x) = x^l, \tag{2}$$

$$F_Y(y) = y^k. \tag{3}$$

The sum of two random variable is calculated as a convolution. The formula for sum can be modified to difference as such:

$$P(X+Y \leq z) = \int_{-\infty}^{\infty} f_X(x) F_Y(z-x) dx, \tag{4}$$

$$P(X-Y \leq z) = \int_{-\infty}^{\infty} F'_X(x) F_Y(z+x) dx, \tag{5}$$

although in case of difference, the formula is not commutative anymore. The formula can be further modified to

$$P(X \leq Y) = \int_0^1 l x^{l-1} x^k dx. \tag{6}$$

The result of the integral is

$$P(X \leq Y) = \frac{l}{l+k}. \tag{7}$$

As we can see now, when $k+l$ is normalized to 100, the probability of X being on the top is $l\%$.

As for visualizing the data density, the transparency settings of the plotted points have been made available to the user so that s/he can view the density if necessary, even though this distorts the information about the points belonging to groups. In this mode, it is necessary to disable the depth test so that no points are neglected and to disable the use of the luminance channel, as it would distort the perceptual transparency.

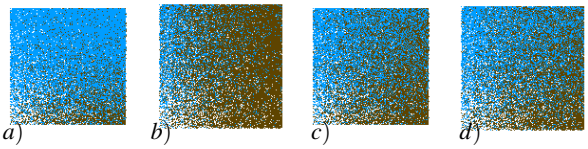

Figure 4: a), c) Blue over brown. b), d) Brown over blue. a), b) No depth test. c), d) Depth test with randomized point depth.

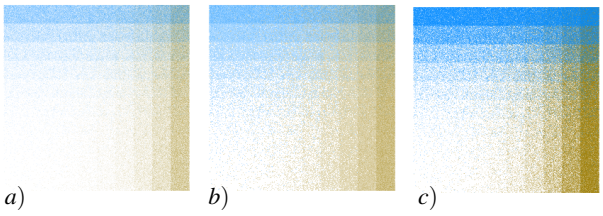

Figure 5: a) Single-pass rendering. b) Two-pass rendering. c) Single-pass with lower transparency causes order-dependent result. Unlike in Figure 4, an exponential falloff is used here.

When visualizing data density using transparency, we find that for high transparency settings, very sparse areas where there are only a few points, surrounded by a black background, are hard to see, and at low transparency settings, the difference in density between the rims and the centers of the clusters cannot be discerned. We cannot increase transparency, as due to alpha blending, clusters drawn last would cover earlier drawn clusters, as can be seen in Figure 5c, top right, where the blue cluster incorrectly covers the brown cluster. Therefore, a non-linear dependence between transparency and density is needed. We implemented this as rendering with transparency in two passes (comparison in supplementary video 2) .[2] At first, all points are rendered with a transparency lower than the user set, and on each screen pixel, only the first point that fell into the pixel is rendered. Then all the data is re-rendered over the previously drawn data using the transparency level set by the user. In both passes, the points are drawn in the same order so that the distortion given by color mixing is the same in both passes.

3.4 Color assignment order

Looking at Figure 6, we notice that some very close or overlapping clusters of points are colored in similar colors, thus merging and creating the impression of a single

[2] https://drive.google.com/drive/folders/1yLt3SFyOHZMTO00O3J20oP4wAha0KPrk?usp=sharing

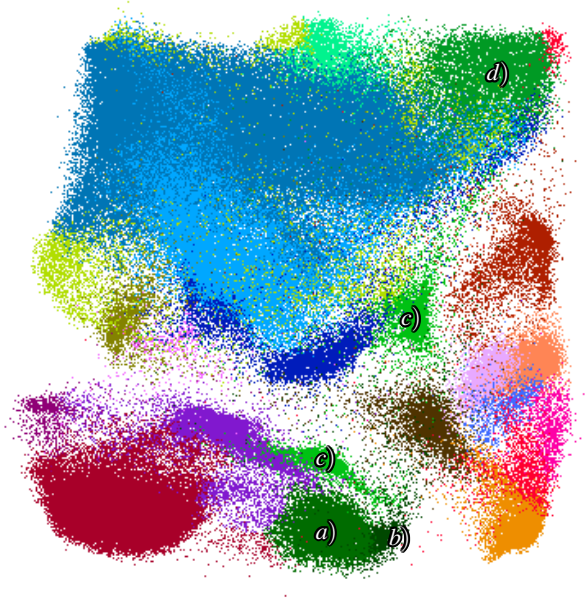

Figure 6: a),b) Co-located groups colored with a similar shade. c) Disjointed cluster. c) and d) might create impression of one disjointed cluster.

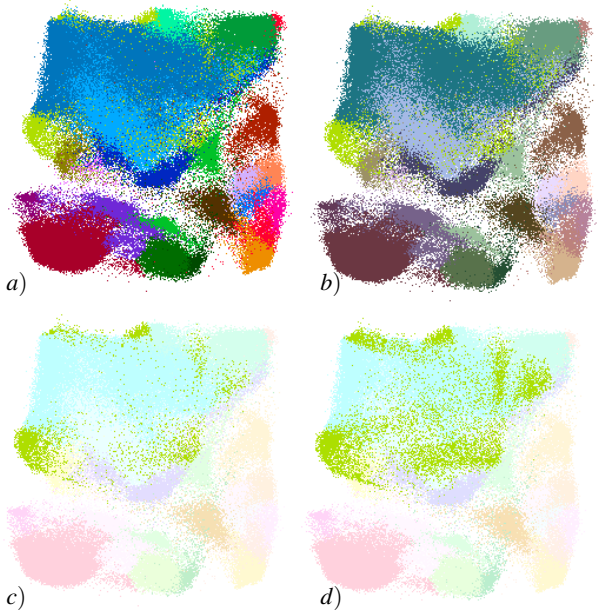

Figure 7: A comparison of context highlighting methods: a) No context highlighting. b) Highlighting with chroma. c),d) Highlighting with chroma and luminance. d) Highlighting cluster is brought to front, thus not overlapped by any cluster.

group. For example, the two clusters, a) and b), are colored in a similar shade of green. An easy distinction between those two clusters could be achieved by permuting the order of colors so that the co-located clusters are colored by colors distant in the color space, as proposed by Lu et al. [7]. But this may create the problem of two distant clusters dyed with a similar color creating the impression of one disjointed cluster, as shown in Figure 6 by clusters c) and d). Similar shades of color are easier to recognize if placed close to each other. Thus, we decided not to permute the color spaces.

3.5 Context selection

Waldin et al. [8] mention improving the discernability of user-selected area of interest by desaturation of the surroundings. In our case, the context selection aims to give a full overview of clusters of the hierarchy (see supplementary video 2, time 0:04) . However, since the mere desaturation of the surroundings turned out to be an insufficiently discriminating channel (see Figure 7b), the luminance channel was additionally used to brighten of the surroundings. Desaturation needs to be used together with brightening, as brightening alone makes colors appear more saturated. Saturated background is undesirable, as it draws users' attention away from the selected context. Desaturation and brightening are already sufficiently distinctive, but there remains the problem of covering the highlighted cluster with a denser cluster (7c), which can be solved by moving the highlighted cluster to the foreground, but this results in the suppression of the context, and we lose information about the background of the marked cluster (7d). The user is given the option to switch between these two methods since it cannot be conclusively said which one is better. Comparing to 7a, we can now see that the context selection gives us an overview of the cluster, which would otherwise remain unknown.

4 Results

An application visualizing clusters of real measured hierarchically grouped data in real-time was created. Running on AMD Ryzen™ 5 3500U APU with integrated Radeon™ Vega 8 GPU, we have measured 30 FPS on a dataset with 250,000 points and 68 leaf nodes in the hierarchy and 45 to 60 FPS on a dataset with 800,000 points and 50 leaf nodes, which shows that the number of categories has a higher performance impact than the number of points. This is due to synchronization between CPU and GPU, caused by querying the number of drawn points after each node drawn. Therefore, smooth running of the visualization on low-end hardware might require reworking the querying to be made for all nodes at once and thus making only one query per frame.

Due to the unavailability of hierarchy data, cluster data were additionally divided into subclusters using the Kmeans++ method [2], resulting in three levels of hierarchy. The application is written in the Java language and utilizes GPU accelerated rendering trough the JOGL library, which is a Java binding for OpenGL.

The visualization allows to distinguish individual clus-

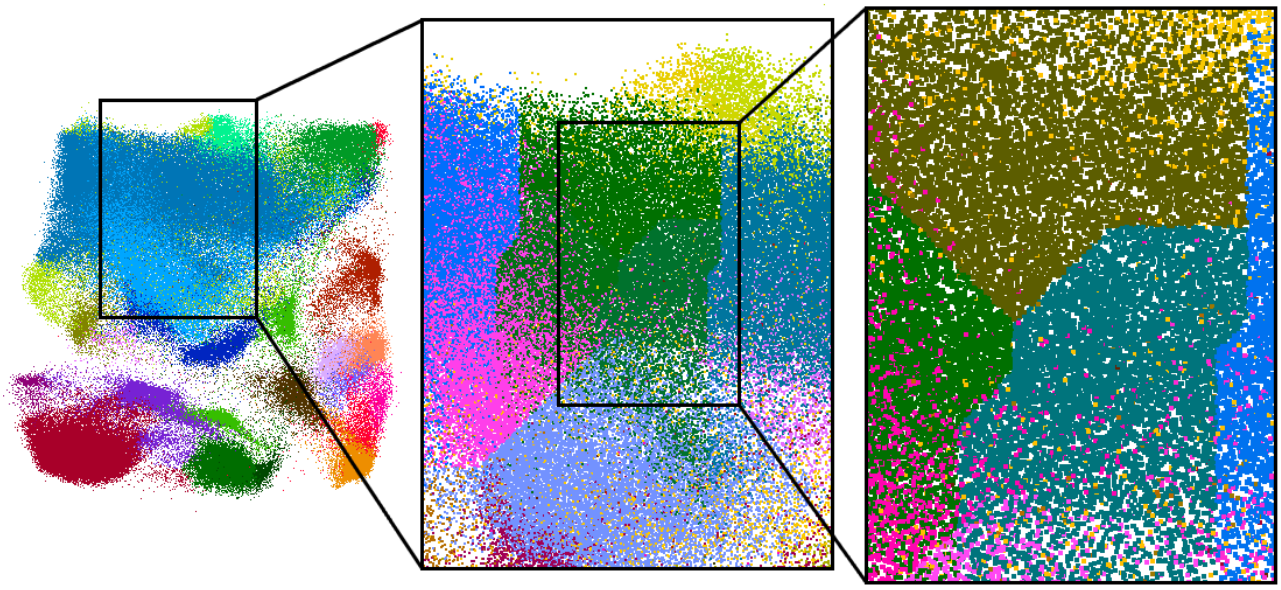

Figure 8: Hierarchical coloring.

ters in a multilevel hierarchy, as shown in Figure 8, where zoom at three consecutive levels is depicted. Individual clusters at any level of the hierarchy can be highlighted (Figures 9, 7) by double-clicking on the cluster of interest. To make selection easier and to allow the selection of an entire subtree or of a cluster overlapped by other clusters, a visualization of the hierarchy as a tree was added to the visualization (Figure 9). The selection is done by double-clicking on a node in the tree.

Figures 10b and 11b demonstrate improvements achieved by two-channel coloring and nonlinear transparency. A comparison of these two figures also shows differences between discernability-based visualization (Figure 10) and density-based visualization (Figure 11). The density based visualization allows for precise perception of the shapes of the two large blue clusters at the top left of the visualizations, but at a cost of lower discernability of the clusters' category.

4.1 Limitations

An unsolved issue is the color inconsistency of the dynamic color palette at high zoom. In most cases, the color difference between adjacent subclusters is satisfying, but typically in very scattered clusters, there are cases where the color shades of the subclusters are very far from each other, and significant color changes occur when the view is moved or zoomed in or out (supplementary video 1) . In a related issue, color differences between clusters having a relatively low number of points are small, making distinguishing them difficult, as can be seen in Figure 6, clusters c) and d). However, context highlighting can be used to resolve this issue.

Since only the division of groups at the population level of the hierarchy was available at the time of design, it was necessary to extend the hierarchy artificially for testing purposes. To subdivide the available data into a multi-level hierarchy, the K-means++ [2] method was used, which is a heuristic hierarchical clustering method often used in statistical data analysis.

5 Conclusion

A visualisation of hierarchically clustered multi-categorical data of medical measurements was created. Such a visualisation is important for medical diagnoses and development. The visualised properties are the density of the clusters, their mutual overlaps and the hierarchical structure.

The nodes of the hierarchy are separated by color with color resolution of twenty-six clusters. Due to the large number of categories, it was necessary to use two color channels, hue and brightness, to achieve resolution. After zooming in, up to seventy clusters can be distinguished with the use of the dynamic color palette. It takes advantage of the fact that the vast majority of these seventy clusters are not visible on the screen when zoomed in. When zooming in, it was necessary to increase the size of the rendered points, which is proportional to the degree of zoom, in order to maintain color resolution. In the application, it is possible to specify at which zoom levels coloring with different colors should be activated at individual depths of the hierarchy.

Due to the mutual exclusion of cluster discriminability and density visualization, two visualization modes were created. Density-focused mode renders points with transparency, while discriminability-focused mode renders fully opaque points. Both modes respect the mu-

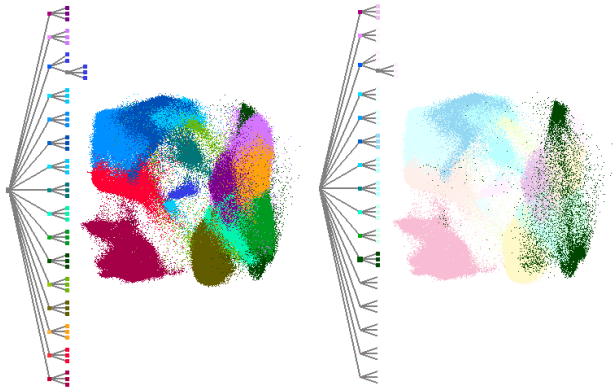

Figure 9: Context selection and hierarchy visualization. The selected hierarchy can also be seen it the tree.

tual overlap of the clusters, i.e., that where several clusters overlap, the color distribution corresponds to the density of the individual clusters. In the case of opaque rendering of points, the overlap visualization was achieved by activating the depth test and randomizing of the point depth.

5.1 Further developments

Visualisation of the hierarchy can be augmented to display the relative frequency of points in individual clusters by scaling individual nodes in the hierarchy visualisation proportionately.

The dynamic color palette can be extended to two dimensions, that is both hue and luminance being assigned dynamically based on the situation. This would improve the utilization of the available color space and the discernability of the clusters as well.

A mentioned but unexplored method is to place labels in the visualization, which would make it easier to distinguish and identify individual clusters. The labeling of the clusters needs to consider the density and overlapping of the clusters, as naive label placement would produce ambiguous or confusing labeling. Čmolík and Bittner [12] propose a method that evaluates places of possible label anchoring based on local opacity salience, overlap salience, and the distance from the edge of the labeled object.

The technique of subsampling the data might also be helpful in achieving the visualization of the density of clusters while maintaining the distinctness of the clusters. A suitable subsampling technique is described by Chen et al. [4], aiming specifically at improving the visualization of overlapping clusters in a multi-class scatterplot.

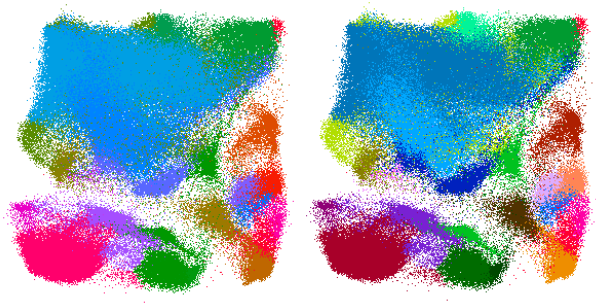

Figure 10: a) Coloring using hue only. b) Coloring using hue and brightness.

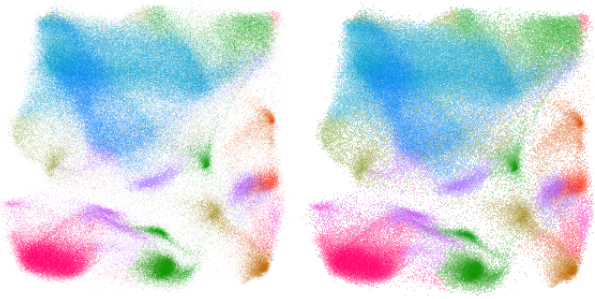

Figure 11: a) Single-pass rendering. b) Two-pass rendering.

References

[1] Hany Alashwal, Mohamed El Halaby, Jacob J Crouse, Areeg Abdalla, and Ahmed A Moustafa. The application of unsupervised clustering methods to alzheimer's disease. *Frontiers in Computational Neuroscience*, 13:31, 2019.

[2] Jason Altschuler. Github - jasonaltschuler: Improvements on classical kmeans clustering, 2013. https://github.com/JasonAltschuler/KMeansPlusPlus [online; accessed 2023-01-05].

[3] Diego Catalano, Robert Theis, and Yuri Pourre. Github - diego catalano/catalano-framework: Framework, 2013. https://github.com/DiegoCatalano [online; accessed 2023-01-05].

[4] Haidong Chen, Wei Chen, Honghui Mei, Zhiqi Liu, Kun Zhou, Weifeng Chen, Wentao Gu, and Kwan-Liu Ma. Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1683–1692, 2014.

[5] Florian Heimerl, Chih-Ching Chang, Alper Sarikaya, and Michael Gleicher. Visual designs for binned aggregation of multi-class scatterplots, 2018.

[6] Rolf G. Kuehni. Hue uniformity and the cielab space and color difference formula. *Color Research & Application*, 23(5):314–322, 1998.

[7] Kecheng Lu, Mi Feng, Xin Chen, Michael Sedlmair, Oliver Deussen, Dani Lischinski, Zhanglin Cheng,

and Yunhai Wang. Palettailor: Discriminable colorization for categorical data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):475–484, 2021.

[8] Nicholas Waldin, Mathieu Muzic, Manuela Waldner, Eduard Gröller, David Goodsell, Autin Ludovic, and Ivan Viola. Chameleon. *Eurographics Workshop on Visual Computing for Biomedicine*, 2016, 09 2016.

[9] Jianxin Wang, Jun Ren, Min Li, and Fang-Xiang Wu. Identification of hierarchical and overlapping functional modules in ppi networks. *IEEE Transactions on NanoBioscience*, 11(4):386–393, 2012.

[10] John Wenskovitch, Ian Crandell, Naren Ramakrishnan, Leanna House, Scotland Leman, and Chris North. Towards a systematic combination of dimension reduction and clustering in visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):131–141, 2018.

[11] Achim Zeileis, Jason C. Fisher, Kurt Hornik, Ross Ihaka, Claire D. McWhite, Paul Murrell, Reto Stauffer, and Claus O. Wilke. colorspace: A toolbox for manipulating and assessing colors and palettes. *Journal of Statistical Software*, 96(1):1–49, 2020.

[12] Ladislav Čmolík and Jiří Bittner. Real-time external labeling of ghosted views. *IEEE Transactions on Visualization and Computer Graphics*, 25(7):2458–2470, 2019.