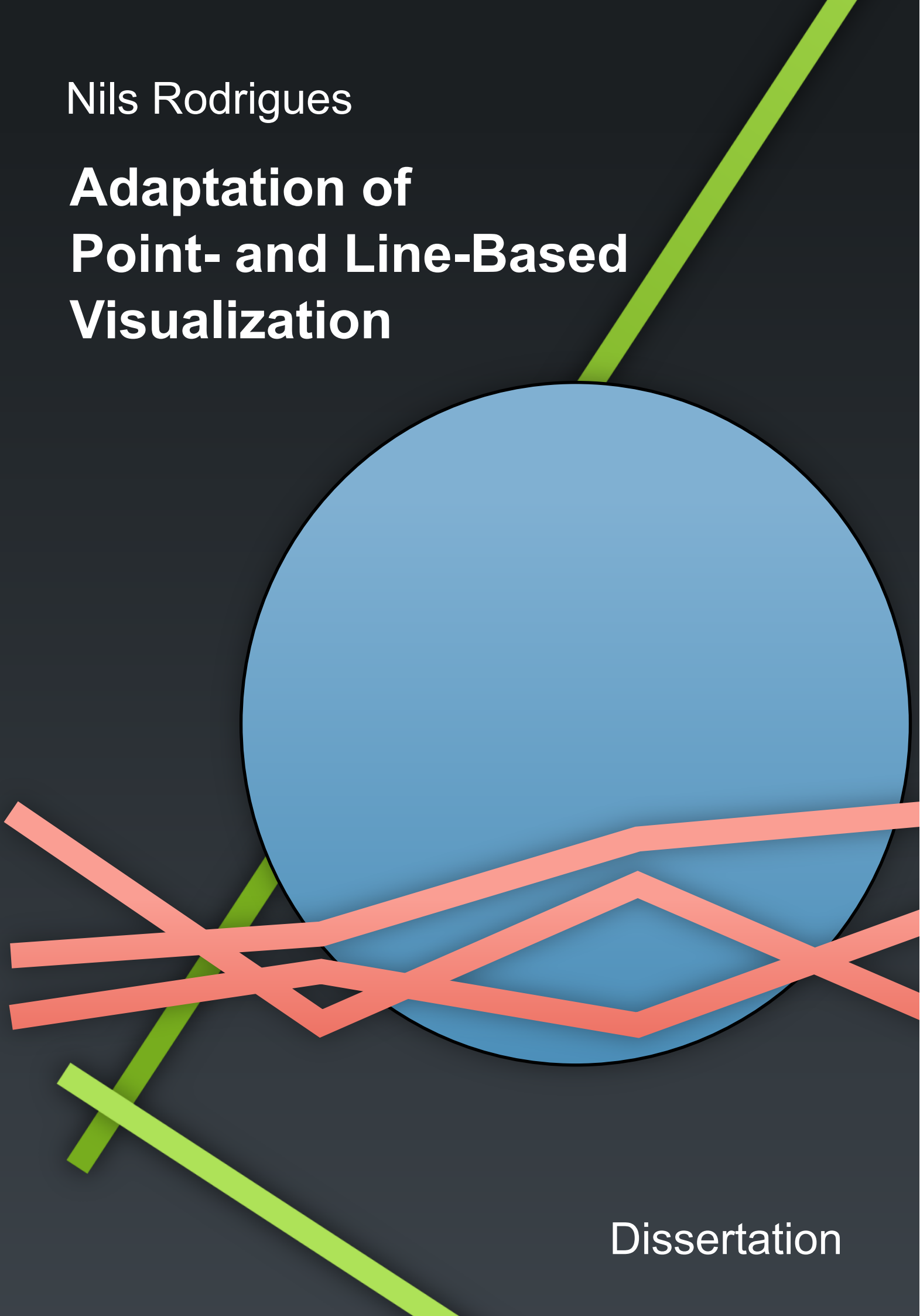
Nils Rodrigues

Adaptation of Point- and Line-Based Visualization



Dissertation

Adaptation of Point- and Line-Based Visualization

Von der Fakultät Informatik, Elektrotechnik und
Informationstechnik der Universität Stuttgart
zur Erlangung der Würde eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

Vorgelegt von

Nils Rodrigues

aus Stuttgart

Hauptberichter:	Prof. Dr. Daniel Weiskopf
Mitberichter:	Prof. Dr. Lars Linsen
Tag der mündlichen Prüfung:	13. Dezember 2023

Visualisierungsinstitut
der Universität Stuttgart

2024

Acknowledgments

First, I want to thank Daniel Weiskopf for giving me the opportunity to do research into visualization, advising me, providing guidance, feedback, and enough liberty to explore various ideas. Daniel, I am also grateful for you taking an active role and all the work you put in—even during the late hours—that helped to get the research published! Thank you also for your patience and motivation.

Thank you, Lars Linsen, for taking the time to review this dissertation and participating in its defense. Thank you, Tobias Schreck, for welcoming me to your institute for the research stay, for the continued collaboration, and for suggestions on what to do in Graz.

I thank all coauthors I met over the years and with whom I had the joy of working on various projects. In alphabetical order: Albrecht Schmidt, Alexander Schultz, Andreas Bulling, Andreas Henicke, Andrés Bruhn, Anja Haug, Antoine Lhuillier, Arman Mielke, Benedikt V. Ehinger, Bruno Burger, Christian Baumann, Christoph Schulz, Cristina Morariu, Daniel A. Keim, Daniel Baumgartner, Francesco Chiossi, Frederik L. Dennig, Guido Reina, Harald Reiterer, Jakob Karolus, Jia Jun Yan, Johannes Zagermann, Katrin Angerbauer, Kazi Riaz Ullah, Krishna Damarla, Kuno Kurzhals, Lewis L. Chuang, Lin Shao, Lorenzo Di Silvestro, Marc O. Ernst, Marco Amann, Martin Raubal, Maurice Koch, Michael Becher, Michael Burch, Michael Sedlmair, Michael Stoll, Nelson de Jesus Silvério da Silva, Nelusa Pathmanathan, Peter Schäfer, Priscilla Balestrucci, Radu Jianu, René Cutura, Rudolf Netzel, Sabine Storandt, Seyda Zerife Öney, Sven Mayer, Sören Döring, Tanja Blascheck, Thomas Ertl, Tiare M. Feuchtner, Tim Krake, Tobias Schreck, and Vincent Brandt. From the long list of coauthors, I want to single out Christoph Schulz for the pleasant and fruitful discussions and for always working hard in our collaborations.

I am also very grateful to Anabela, Christoph, Jessica, Kuno, Moataz, and Tycho (in alphabetical order) for reading this thesis and for their valuable feedback. Very heartfelt thanks to my family and friends who endured me and stood by my side. I would not be where I am now without your strong support! Thank you for everything! I am especially thankful to Jessica for helping me stay focused, supporting me over the years as well as during the write-up, and taking burdens off my back when she herself was swamped.

My work at the Visualization Research Center of the University of Stuttgart was mostly funded by the German Research Foundation DFG within project B01 of SFB-TRR 161.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations and Acronyms

2AFC	two-alternative forced-choice
ACM	Association for Computing Machinery
AR	augmented reality
BNP	blue noise plot
BSP	bee swarm plot
CF-PCP	cluster-flow parallel coordinates plot
CHCCS	Canadian Human-Computer Communications Society
CPU	central processing unit
DBSCAN	density-based spatial clustering of applications with noise
DDA	differential domain analysis
DR	dimensionality reduction
EBL	edge-bundling layout
FA	Fourier analysis
GPU	graphics processing unit
GUI	graphical user interface
HMD	head-mounted display
IEEE	Institute of Electrical and Electronics Engineers
IPCs	illustrative parallel coordinates
ISE	Fraunhofer Institute for Solar Energy Systems
KDE	kernel density estimation
KFE	kernel frequency estimation
MDM	mean dot movement
MOD	mean dot overlap in display diameters
MSE	mean squared error
NDP	nonlinear dot plot
PCA	principal component analysis
PCP	parallel coordinates plot
RDP	relaxed dot plot
RGB	red, green, and blue color model
SP	scatter plot

SPLOM	scatter plot matrix
SSE	sum of squared errors
TU Graz	Graz University of Technology
UMAP	uniform manifold approximation and projection
VAC	vergence-accommodation conflict
VISUS	Visualization Research Center of the University of Stuttgart
VR	virtual reality

Abstract

Visualization plays an important role in the lives of various heterogeneous parts of society: from a voter looking for the latest results of an election, to statisticians examining a distribution, to analysts trying to make sense of multidimensional data sets. This thesis adapts existing point- and line-based visualization methods to improve knowledge gain. The included contributions address three research questions: *How to scale unit visualization for 1D data? How to improve navigation between 2D visualizations of multivariate data? How to combine the advantages of multiple 2D views in a single static visualization for multivariate data?*

The first part of the thesis focuses on unit visualization of 1D data with dot plots. Compared to the previous state of the art, the developed visualizations fit a wider range of data and expand the number of potential users by requiring less prior knowledge for interpretation. They adapt the definition of dot plots to scale nonlinearly with sample count, accurately show value frequencies in high-dynamic-range data, reduce positional error in displayed data points, and enhance the perception of subtle nuances in the data while avoiding moiré effects. We provide evidence for claimed improvements through evaluation with computational metrics and a crowdsourced user study.

The second part of the dissertation focuses on visualizing multivariate data with scatter plots and scatter plot matrices. First, we evaluate six animated transitions between plots of different 2D subspaces with respect to task performance for tracking individual points and interactions between clusters. The results of a quantitative study with 170 participants show that orthographic rotation animation performs best and should be adopted more widely. Next, we develop a novel concept for recommending views in scatter plot matrices. It provides user- and task-specific suggestions by focusing on the data of interest to the viewer. Together, animation and recommendation adapt scatter plots to improve the user's ability to analyze more complex data effectively.

In the third part, we develop a new visualization technique that extends parallel coordinate plots to provide a static alternative to scatter plots with animated transitions. The approach does not require interaction to display data flow between 2D subspace clusters. A custom density-based rendering technique enables the visibility of individual lines and structures within highly overdrawn regions. Our technique can communicate fuzzy clustering results through binning and color mapping.

Finally, we discuss the presented contributions with respect to the original main questions and show possible directions for future research.

Zusammenfassung

Visualisierung spielt im Leben verschiedener Teile der Gesellschaft eine wichtige Rolle: Von einem Wähler, der nach den aktuellsten Wahlergebnissen sucht, über Statistiker, die eine Verteilung untersuchen, bis hin zu Analysten, die mehrdimensionale Datensätze verstehen wollen. Die Beiträge dieser Arbeit passen bestehende punkt- und linienbasierte Visualisierungsmethoden an, um den Wissenserwerb zu verbessern, und widmen sich dabei drei Forschungsfragen: *Wie kann man die Visualisierung von 1D-Daten mit einzelnen grafischen Elementen skalieren? Wie kann man die Navigation zwischen 2D-Visualisierungen multivariater Daten verbessern? Wie kann man die Vorteile mehrerer 2D-Ansichten in einer einzigen statischen Visualisierung für multivariate Daten kombinieren?*

Der erste Teil dieser Arbeit fokussiert sich auf die Visualisierung von 1D-Daten mit einzelnen grafischen Elementen in Punktdiagrammen. Im Vergleich zum bisherigen Stand der Technik sind die entwickelten Visualisierungen für ein breiteres Spektrum an Daten geeignet und erweitern den Kreis potenzieller Nutzer, da sie weniger Vorkenntnisse erfordern. Sie passen die Definition von Punktdiagrammen an, um nichtlinear mit der Punktezahl zu skalieren, Werteverteilungen mit hohem Dynamikumfang präzise darzustellen, Positionsfehler der angezeigten Daten zu reduzieren und die Wahrnehmung subtiler Nuancen zu verbessern, während Moiré-Effekte vermieden werden. Wir belegen die Verbesserungen mit rechnerisch ermittelten Metriken und einer Crowdsourcing-basierten Nutzerstudie.

Der zweite Teil der Dissertation behandelt die Visualisierung multivariater Daten mit Streudiagrammen und Streudiagrammmatrizen. Zunächst untersuchen wir sechs Animationen zwischen Plots unterschiedlicher 2D-Unterräume im Hinblick auf die Verfolgbarkeit einzelner Punkte und die Interaktion zwischen Clustern. Die Ergebnisse einer quantitativen Studie mit 170 Teilnehmern zeigen, dass orthografische Rotationsanimation am besten abschneidet und breiter angewendet werden sollte. Anschließend entwickeln wir ein neuartiges Konzept für die Empfehlung weiterer Ansichten in einer Streudiagrammmatrix. Es bietet nutzer- und aufgabenspezifische Vorschläge, indem es den Fokus auf die für den Betrachter interessanten Daten legt. Zusammen passen Animation und Empfehlung Streudiagramme an, um die Fähigkeit des Benutzers zu verbessern, komplexere Daten effektiv zu analysieren.

Im dritten Teil entwickeln wir eine neue Visualisierungstechnik, welche parallele Koordinaten erweitert und eine statische Alternative zu Streudiagrammen mit animierten Übergängen darstellt. Der Ansatz erfordert keine Interaktion, um den Datenfluss zwischen 2D-Subraum-Clustern anzuzeigen. Angepasstes, dichtebasiertes Rendering ermöglicht die Sichtbarkeit

einzelner Linien sowie Strukturen innerhalb stark überlagerter Bereiche. Die Ergebnisse von unscharfem Clustering werden durch Intervallbildung und Farbcodierung dargestellt.

Abschließend diskutieren wir die Beiträge im Hinblick auf die ursprünglichen Fragen und zeigen mögliche Richtungen für zukünftige Forschung auf.

1

Introduction

Visualization is ubiquitous. Young pupils use them in school when they first draw charts of mathematical functions to gain intuitive insight into the characteristics of equations. Viewers are confronted with visualization as part of the news when bar charts show a recent development. Infographics help to understand complex systems and how individual components are interconnected. Data analysts and visualization experts compose figures to gain insight into large data and use them to communicate information with other people. All these kinds of visualizations, simple or complex, can be created using an underlying pipeline. Haber and McNabb created a formal definition for such a pipeline in the context of scientific visualization [HM90]. We adapted Card et al.'s derived pipeline [CMS99] and will use it throughout this document to put proposed changes into context (see Figure 1.1). The first stage of the pipeline takes a potentially large input data set and extracts a subset of data that is relevant for visualization. The second stage maps the data to a description of individual visual primitives. The rendering stage creates a visible representation from the description without referencing the original data. Visualization serves human beings. Hence, the final stage is the perception by a viewer. In the case of interactive visualization, the viewer can adapt parts of the pipeline, creating a feedback loop for visual analytics [Kei+08].

Some visualizations are dynamic or interactive, giving their users, e.g., data analysts, the ability to adjust parameters to gain new insight from images. Other visualizations are created by a designer and remain static. They can then serve the same person—this time in the role of an analyst—or be distributed among others, who then use them for their own analysis or as confirmation of communicated results. The latter is often the case in print media, e.g., newspapers, where readers are purely "consumers" and cannot change the presented graphics.

The scientific field of visualization research has matured, and publi-

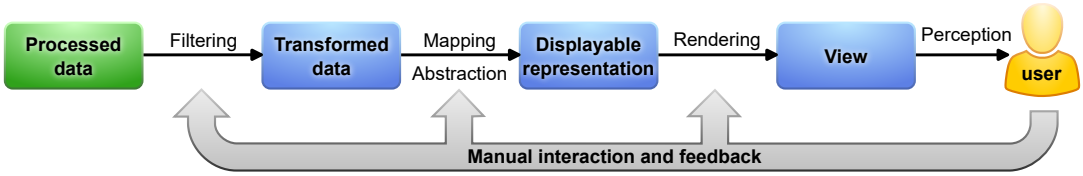


Figure 1.1: The visualization pipeline, adapted from [CMS99]. Humans can use existing knowledge to adjust parameters and interact at various steps to adapt the output and gain new insights.

cations include increasing numbers of visual representations [Rei+20; Che+21]. Researchers have creative freedom for the design of new visualization techniques. The range of used visual elements spans from text and simple dots, over shapes composed of lines and polygons, to continuous fields of color and further. The complexity of employed and developed software for data visualization has increased over time [Rei+20]. On the one hand, and if used correctly, this growing complexity of tools along the visualization pipeline might reduce the workload for human users of the resulting images by preprocessing data, using more elaborate algorithms to find information of interest, optimizing perceptual variables, etc. On the other hand, it might also lead to the creation of visualizations that encode more and more data, contain evermore visual primitives, and become too complex or overwhelming for humans to analyze and extract information.

1.1 Goals and Research Questions

Contrary to the underlying trend of growing possibilities, this thesis focuses on known visualization techniques that only use dots and lines. They are typically used for relatively small data sets with a limited number of dimensions. The overarching goal of this thesis is not the creation of radical new visualizations but the adaptation and extension of proven methods to improve the knowledge gain. The presented plots and techniques target data analysts with prior experience but also the general public. To achieve our overarching goal, we identify three central research questions:

RQ1 – How to scale unit visualization for 1D data?
Visualizing 1D data with a graphical element for each data point helps to make it relatively simple to understand. It is, therefore, well suited to target large audiences, including the general public. The most straightforward visualization techniques for 1D data might be limited to low sample counts in the orders of tens or hundreds. We develop these plots further and make them suitable

for high-dynamic-range data sets with several thousands of data points. As screen space is limited, we do not expect the resulting techniques to show each and every sample but to adapt to the underlying data and retain the advantages of unit visualization where possible. In the context of recent work on scalability in visualization [Ric+23], the result should be more scalable in the sense of *shape* because it requires nonlinear functions to improve the visibility of outliers in high-dynamic-range data. Additionally, the answer to this question should increase the suitable problem size for a given *threshold* to allow for larger sample counts in the same screen space.

RQ2 – How to improve navigation between 2D visualizations of multivariate data?
Most visualization is 2D and presented on digital screens. It seems logical to select only two attributes from a multivariate data set and to only display this subspace. It results in comprehensible visualizations that allow for multiple types of insight. To gain more knowledge from the underlying data set, it is then necessary to select and analyze multiple such subspaces. This question aims to help the analysts with their analysis tasks by facilitating the discovery of and switching between fitting 2D visualizations from multivariate data.

RQ3 – How to combine the advantages of multiple 2D views in a single static visualization for multivariate data?
Visualization is not always interactive, e.g., when printed on paper. This question targets the creation of a plotting technique that can provide insights from multiple 2D visualizations in a single static image. To this end, we extend the range of included approaches beyond simple dots and points to also encompass polylines, giving us more freedom to design a new visualization technique.

1.2 Outline and Research Contributions

This thesis contains chapters that are based on published or submitted papers and reused with consent from the coauthors. I am the main author in most cases. For publications where I only contributed to distinct parts, I will limit reuse to the sections that I wrote. Daniel Weiskopf is a coauthor of all papers, contributing his experience and valuable feedback, helping to polish the content for publication, and editing the manuscripts. He supervised me in all projects behind the papers.

After the introduction, **Chapter 2** provides a coarse background on data types and how they can be visualized. On this basis, it helps motivate the main research questions and explains the selection of techniques that follow. The main content of this document is structured around these main research questions, which themselves are arranged according to the dimensionality of numerical data and main drawing primitives for visualization. Each of the following major parts of this dissertation creates a link back to the research questions and provides an overview of how the presented content ties into the general visualization pipeline.

Part I (Dot Plots) answers  RQ1 and focuses on 1D data and its frequency. At the beginning, it gives a short overview of the concepts behind multiple visualizations that are called “dot plots” and clarifies which of these techniques we use as a basis for further development.

Chapter 3 – Nonlinear Dot Plots This chapter introduces nonlinear scaling to dot plots in order to allow the visualization of larger data sets with highly dynamic data frequencies. It is based on the results of the first project I started to work on at the Visualization Research Center of the University of Stuttgart (VISUS) [RW18].
Daniel Weiskopf had the initial idea of nonlinear scaling for dot plots. I implemented the adapted algorithm and extended it into a two-way-sweep layout with two scaling functions. I developed a software suite for testing the layouts and introduced a vertical blur in dense areas.
Chapter 4 – Relaxed Dot Plots In this chapter, we improve on the display of data frequency and increased positional correctness in dot plots while, at the same time, providing better aesthetics with little to no moiré effect. It is based on a project that led to the publication of dot plots without straight columns [Rod+23b; Rod+22b].
Christoph Schulz and I both identified parts of the Linde-Buzo-Gray stippling algorithm [DSZ17] that might be applicable to dot plots and introduced new constraints. As a result, we arrived at Lloyd relaxation as a technique that was useful for redistributing circles inside an envelope shape. I performed the transfer from dot-count-based nonlinear scaling from Chapter 3 to a frequency-based approach in order to compute the envelope shape from kernel frequency estimation of the underlying visualization data. Tim Krake proved mathematical properties and helped to create consistent equations.
Having a container shape and an algorithm for the distribution of circles, I designed the initial adapted dot layout that serves as a starting point for relaxation and avoids visual artifacts. I also developed an approach with linear complexity for *tunneling* swaps between dots that increases positional correctness and decreases runtime. Two student assistants—Sören

Döring and Daniel Baumgartner—implemented all algorithms. Christoph Schulz and I analyzed the code and provided feedback for improvements. I designed and conducted a crowdsourced study to evaluate the relaxed visualization against regular nonlinear dot plots. Sören Döring implemented a generator for study stimuli and set up a template for the online survey.

Chapter 5 – Excursion: Mathematical Transition from Columns to Frequencies The equations for the mathematical transformations from column-based to frequency-based scaling are contained in this chapter. The content is based on a document that is part of the supplemental material to relaxed dot plots [Rod+22b]. Tim Krake provided a first draft for a clean write-up of the math behind the transformations, which I then edited and extended with other material for the completed document. I present this work as an additional excursion to be able to better explain the ideas behind the transformations and to not interrupt the flow of text in Chapter 4 with too many equations.

Part II (Scatter Plots) answers RQ2 and focuses on exploring and navigating pairs of two dimensions from multivariate data. It starts with an introduction to the application of scatter plots—as individuals and in matrices of small multiples—for the visualization of multidimensional data.

Chapter 6 – Comparative Evaluation of Animated Scatter Plot Transitions This chapter compares multiple animation techniques for switching between scatter plots. It evaluates them with respect to traceability of individual points and interactions between data clusters. We published the work behind this chapter in IEEE Transactions on Visualization and Computer Graphics [Rod+24].

This project started with the desire to visualize flow between clusters in different scatter plots of the same underlying data set. Under my supervision, Vincent Brandt created a software library for animated transitions between scatter plots and made a preliminary small-scale comparison of a wide range of animations that sparked interest in a more thorough evaluation. To this end, I designed and conducted a crowdsourced study with 170 participants and two pilot studies. Frederik Dennig gave feedback on the design of the study and provided a first partial draft for the write-up of related work. Daniel Keim supervised Frederik Dennig and provided feedback on the ideas of the paper. I performed the statistical analysis of the study results and wrote the remainder of the publication.

Chapter 7 – Gaze-Based Recommendations for the Exploration of Scatter Plot Matrices This chapter follows logically from the previous one, and both together provide an answer to RQ2: since we now know what animation between scatter plots works best, we want to recommend specific dimension pairs that suit the analyst's task. The main content of

this chapter is based on a joint publication with researchers from Austria [Rod+22c].
The project started with a research stay at Graz University of Technology (TU Graz). Tobias Schreck, Lin Shao, and I jointly supervised the student Jia Jun Yan, who implemented a proof of concept for a visual analytics application. I proposed possible ways of using gaze to implicitly select data and helped by suggesting techniques to solve specific problems. I took the lead role in writing the paper. Lin Shao and Tobias Schreck gave feedback for the paper and helped with the related work based on their prior knowledge of multiple recommender systems.
Chapter 8 – Excursion: Eye Tracking for Immersive and Situated Analytics We based the implicit data selection from the previous chapter on gaze data. Within this excursion, we extend the display dimension and discuss the use of eye tracking in the context of immersive and situated analytics. First, the chapter presents hypothetical benefits and challenges of adding gaze support to an existing virtual reality application (FiberClay [Hur+19]). The work is based on my contributed sections about the analytics tool that appear within a publication with many authors from institutions in multiple countries [Sil+19].
The second part of the chapter contains an empirical evaluation of measured gaze depth on real and virtual objects in augmented reality. It is based on a publication from 2020 [Öne+20]. Seyda Öney implemented all necessary software as part of her bachelor thesis. Michael Becher advised her on topics related to HoloLens, while I focused on eye tracking and the comparative study. Seyda Öney conducted the study and analyzed the results. She summarized them in a first draft that I used as inspiration to write the publication. Michael Becher helped to edit the paper and was supervised by Guido Reina and Thomas Ertl. Michael Sedlmair was the examiner of the student work.

Part III (Parallel Coordinates Plots) answers RQ3 and presents a visualization technique that aims at combining advantages from animated scatter plot transitions and static parallel coordinates plots (PCPs). It starts with a short introduction to rendering techniques that tackle the problem of overdraw in PCPs.

Chapter 9 – Cluster-Flow Parallel Coordinates Plots This chapter is based on published work [Rod+20] and describes a visualization technique that we developed to answer RQ3. During conversations with Christoph Schulz about fuzzy clusters, I came up with the initial idea of replicating the axes of parallel coordinates to show cluster assignments. We involved Antoine Lhuillier in the discussions due to his experience in visualizations with bundled lines. All three of us regularly discussed ideas and came up

with the final design. Thus direct attribution is difficult. I implemented the web-based visualization, metrics, and an algorithm for ordering axes (horizontal) and clusters (vertical). Christoph Schulz contributed fuzzy clustering and his knowledge in the field of uncertainty. Antoine Lhuillier performed initial analyses of the NetPerf data set. While I was involved in writing and editing all sections of the paper, the other authors contributed text segments and provided proofreading over the entire document.

Chapter 10 concludes this thesis with a discussion of the results with respect to the research questions and an outline of future research directions.

Material from the publication [RW18] is copyrighted by IEEE and used with permission under the agreement for reuse in a thesis. The material from the publication [Rod+23b] is public domain and licensed to IEEE under the Creative Commons Attribution 4.0 (CCBY 4.0). Material from the publications [Rod+22c; Sil+19; Öne+20] is copyrighted by ACM and used with their permission under the agreement for reuse in a dissertation. The copyright for material from the publication [Rod+20] is retained by its authors, while CHCCS is granted non-exclusive rights of publication.

I had the joy of collaborating with many good, intelligent, and motivated researchers from VISUS and other fine institutions. To honor their contributions to the work presented in this dissertation, I use the editorial “we” for most of the content and only revert to “I” where appropriate. I do not explicitly cite extracts from coauthored publications to improve readability.

During my time at VISUS, I also coauthored other publications that do not fit the general theme of this thesis and from which I do not use excerpts. A collaboration with the Fraunhofer Institute for Solar Energy Systems (ISE) led to a paper that presented a map interface for data from power plants that was integrated with existing charts on a website (`https://www.energy-charts.info`, accessed 2024-01-27) [Rod+17b]. Collaboration with IBM resulted in a paper about the application of PCPs for data analysis at the same company [Sch+17].

Eye tracking was a topic in multiple publications. We created visualizations and tools for gaze analysis on static stimuli [Rod+18; Sch+22; Sch+23] and videos [Kur+20]. We used eye tracking as input for interactive applications in augmented reality [Pat+20] and showed how it could be used with adaptive user interfaces [Chi+22].

The human visual system and perception have also played a role in multiple publications. For example, we modeled the strength of simultaneous orientation contrast and compensated it through counter-rotation [Net+19]. We developed a simulator for various physiological disorders of the visual system [Sch+19] and analyzed the effects of color vision deficiency on the

accessibility of figures that were published over the years at the IEEE VIS conference [Ang+22].

Based on prior work from my diploma thesis and together with my former supervisor Michael Burch, I published a visual analytics approach for the exploration of texts from multiple corpora that won a 🏆 best paper award [Rod+17a]. I also generated figures for a previous publication about the visualization technique that I implemented during the work on my diploma thesis [Bur+14].

2

Data Types and Their Visualizations

All visualizations have one thing in common: they represent the underlying data that forms the starting point of the pipeline in Figure 1.1. However, not all visualization techniques are well-suited for all kinds of data. This chapter discusses different types of data and provides a background for corresponding visualization approaches. It shows how the detailed techniques proposed in the following parts of the thesis relate to the overarching goal and research questions.

2.1 Low-Dimensional Data

Simple 1D data and its visualizations are essential for humans, even in a world where *big data* and *data retention* are increasingly relevant to artificial intelligence [Jai+20] and further evolving quantum computing [Tow22; UK 21]. For example, large portions of populations are affected by the results of elections for public offices. The number of samples, i.e., votes, in such a case is typically high while the dimensionality is low because each vote is a single choice from a small range of values, i.e., parties. Visualizations of this low-dimensional data are sufficient to gain information and educate the public about the outcome of the election: the total number of votes, the ratio of abstentions, the ratio between parties, etc. Most people do not know the details of the volatile energy market, how much electricity is generated, where it comes from, or where it is currently needed. In the context of climate change, however, they might be interested in knowing simple, 1D data. For example, the percentage of renewable energy, the amount of carbon dioxide emissions, or the mix of primary energy sources of their consumed electricity. Single-dimensional data even arises when friends and family enjoy their free time and play games: the heights of the

bar in limbo, the number of goals in football, or the score of each round in a card game.

Dot plots visualize 1D data. They only use a simple number line and place dots along it. This works well for small data sets and is very accessible to a broad range of viewers. Previously, this kind of visualization did not scale well for larger high-dynamic-range data, i.e., containing outliers and regions with high sample frequency alike. Addressing RQ1, we will investigate methods of applying nonlinear scaling to dot plots. Although they require more experience to interpret correctly, nonlinear scales already work well with bar charts. For the nonlinear plots proposed in Part I, we will apply the scaling to the dots that represent individual data points. Such unit visualization provides an accessible layout where the data frequency is low because viewers are able to recognize and count the individual dots. On the one hand, the simplicity of dot plots provides visualization for a broad range of audiences. On the other hand, it is only suited to display a single dimension. We can enrich the visualization by encoding a second attribute as color or icons within the dots. Such an approach works best for few and quantized values.

When the data contains two numerical dimensions, it is intuitive to also use two display axes. Many viewers are used to this from school, where they plotted equations with x- and y-variables in Cartesian coordinate systems. But there is also a need to analyze data where there is no previously known equation that governs the relationship between the two dimensions. In this case, we can use the x-axis and y-axis of the display and draw unconnected individual data points. The resulting scatter plot allows viewers to gain deeper insight into possible connections between the data dimensions: It allows them to find correlations, to identify trends and clusters.

2.2 Multivariate Data

Plots and graphs up to 2D are helpful in cases where the data is of the same low dimensionality. Nevertheless, data sets can include more than two attributes for the same record. For example, a table with champions in motorsports might include the year of a race, name of the driver, name of the team, various measurements of the vehicle, length of the track, total race time, etc. A different data set might include additional attributes forming a network graph that shows how different racing teams interact. With the abundance and variety of recorded information, it is inevitable for some analysts to work with multivariate data. For example, when trying to optimize engine power output, it is necessary to analyze multiple variable settings, i.e., dimensions, and find their effect on the final output but also

interactions between multiple settings. Increasing the amount of injected fuel requires a different setting for the air intake valve and might influence the combustion temperature.

What visualizations can analysts use when the number of dimensions increases beyond two? There is a plethora of techniques that fit specific kinds of data. To keep the length of this thesis within reasonable limits, we will only discuss generic visualizations for numerical data on rational, ordinal, or nominal scales.

2.2.1 Point-Based Visualization

Scatter plots are a suitable approach for 2D data, and many people have used them, but their perpendicular axis layout does not scale with a higher number of dimensions. 3D scatter plots on 2D paper or computer screens are prone to overdraw, and the introduced virtual depth is not easily perceivable for the user. It is, however, possible to create a grid with scatter plots of all dimension pairs: the scatter plot matrix (SPLOM) [Eme+12]. As the number D of data dimensions increases, each individual cell shrinks to fit the increasing number of D^2 cells in the matrix. Therefore, this approach is often combined with a regular scatter plot. The matrix provides only coarse thumbnails and allows the user to select a dimension pair for the larger plot.

This technique allows analysts to explore the data in detail. But how can they trace individual points between the different plots? Previous work has suggested animating the dots when switching between data axes in the main scatter plot [EDF08]. In accordance with RQ2, this thesis presents a selection of animation techniques and compares them with respect to user performance for following individual data points and for identifying cluster interactions (see Chapter 6).

With animation, it is easier to maintain the mental model while switching between different views. But how should analysts know what cells of the SPLOM to select? Which cells are the most relevant and have the least redundancy? Previous work has resulted in recommender systems that use a manual selection of known relevant cells to suggest new dimension pairs [Beh+14]. In line with RQ2, this dissertation presents a technique that uses the analyst's gaze to identify data of interest. The proof of concept in Chapter 7 combines this implicit data selection with multiple metrics to provide information that can help the viewer select the next interesting cell in the SPLOM. The concept of using gaze to select data is reasonably well suited for 2D visualization. Beyond the original scope of RQ2, we also explored the use of 3D gaze with stereo eye-tracking hardware in Chapter 8.

2.2.2 Line-Based Visualization

A SPLOM scales with the square of the number of dimensions (D^2), requires much space, and the individual cells become increasingly smaller to fit on the screen. Additionally, it requires interaction to switch between different detailed views. What happens when the number of data dimensions increases too far, or there is no way of interacting with the plotting software? For this reason, we formulated RQ3 and investigated ways to combine advantages from multiple 2D views in a single static visualization. One could use dimensionality reduction techniques to arrive at a 2D scatter plot from an arbitrary number of data attributes. However, the resulting display dimensions are often not intuitive to grasp as the connection to the underlying data dimensions is obfuscated.

A possible answer to RQ3 is not based on visualization with points but rather with lines. A PCP shows all data dimensions with parallel—most often vertical—axes. Each data point becomes a polyline, and the crossings with the axes represent the data value in each dimension. The results of an expert review suggest that they are helpful for analysts that want to explore previously unknown multivariate data [Sch+17]. In Chapter 9, we propose a technique that extends on the original PCPs and allows reading the data values of individual samples but also shows soft clusters in 2D subspaces (advantage of scatter plots). Our proposed visualization shows how individual data points move between clusters and how the clusters of adjacent dimension pairs interact with each other (advantage of animation between scatter plots). The technique is, however, targeted at experienced analysts.

As with all PCPs in general, the axis sequence on the screen influences heavily what patterns of the data are visible. There is previous work on quantifying the patterns and optimizing the sequence [DK10]. As our work in the context of RQ3 focuses on transferring the advantages from multiple 2D visualizations, we also developed new metrics to optimize the dimension order for cluster interactions.

2.3 Spatial Data

Spatial data can be encoded with two and three dimensions. It is often well-suited for display with individual dots marking a position on a map and can also be useful to draw lines that show trajectories. We created a point-based map visualization of data on power plants for the general public [Rod+17b]. We also worked on the visualization of eye-tracking scanpaths. They are well suited for display on a stimulus by drawing lines for saccades and,

optionally, dots for fixations. We also worked with eye-tracking trajectories from multiple study participants on the same stimulus and found ways of reducing the visual clutter of scanpaths. We used one approach based on quad-trees [Rod+18] and one with group diagrams [Sch+22; Sch+23].

While work on spatial data is important, it is often targeted toward the visualization of data sets from specific contexts and does not fit the broader and more generic theme of this dissertation. Hence, we will not include these applications in further discussions.

Part I

Dot Plots

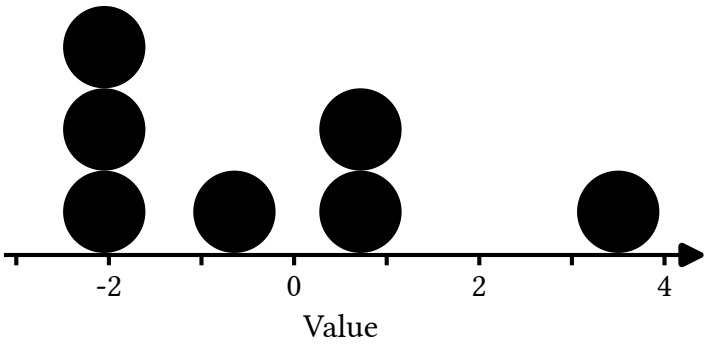


Figure I-1: A dot plot, according to Wilkinson [Wil99]. Each dot represents a value along the number line at the bottom. Dots are stacked into straight columns if they cannot be placed next to each other without overlap.

Visualization is not limited to complex interactive systems for the particular analysis of large data sets. It can be simple, show only a single data dimension with few samples, and yet be helpful. Dot plots are a case of such relatively simple visualization. Even school children without prior knowledge can use them to extract characteristics of the data distribution [I G15]. The basic algorithm behind dot plots does not require computing hardware and can be used for hand-drawn visualizations.

Dot plots are composed of a single number line and a circular dot for each data point. When neighboring dots would overlap, the layout algorithm stacks them into straight columns (see Figure I-1). As a result, while the input data only has a single attribute, dot plots use 2D output to convey information: One axis—typically horizontal—displays the value of the underlying items, and a perpendicular axis shows data frequency.

The attentive reader might now note how the goals of histograms and dot plots overlap. There are, however, decisive differences. First, dot plots are a kind of unit visualization that maps each data point to an individual graphical element. This allows for a clearly readable representation with countable items and tends to be more intuitive to understand [Bak04]. The second major difference concerns the bins in regular histograms. All points along the visible data axis belong to a bin. There are no unassigned ranges. Additionally, the bins themselves are uniform. Contrary to this, the dot plot positions the circles where there are data items. This can create empty spaces between areas of higher data frequency and outliers that are positioned accurately with regard to their underlying value.

Unfortunately, dot plots also have disadvantages with regard to other visualization techniques. Unit visualization is better for an intuitive understanding, but it does not scale well with increasing sample count and high-dynamic-range data. Stacking dots into straight columns moves the display points away from their underlying value. Finally, showing many

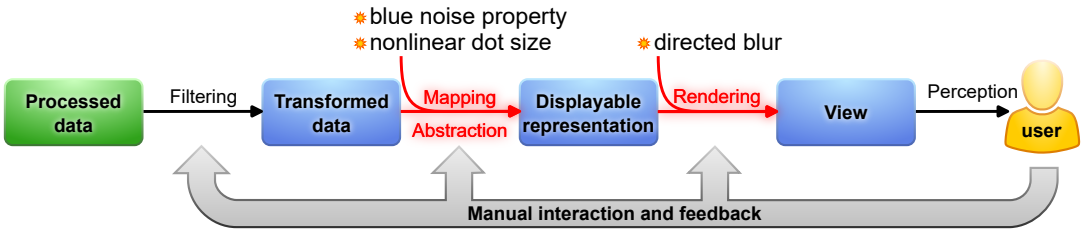


Figure I-2: The visualization pipeline (adapted from [CMS99]) with additions from the proposed dot plot techniques. Affected elements are highlighted in red. High-dynamic-range data is supported through dot size scaling. Directed blur and blue noise reduce unwanted perceptual effects.

dots in close proximity tends to create moiré effects. Following the theme of RQ1, we adapted and extended various parts of the visualization pipeline of dot plots to tackle the disadvantages and make this intuitive and accessible visualization suitable for a wider range of data (see Figure I-2). In Chapter 3, we change the mapping from data points to differently sized dots and include a data-agnostic low-pass filter to reduce moiré patterns. In Chapter 4, we adapt the location of dots to decrease positional error and ensure the blue noise property, which is better suited for visual perception in the later stage of the pipeline. With these latest changes, remaining visual patterns are not an unwanted side effect but actually convey information about the underlying data.

Definition of Dot Plots

Before continuing with details about the proposed techniques, we have to define what we mean by “dot plots.” There is ambiguity because, over time, there have been multiple visualizations that share the same name. Already in 1884, Jevons is supposed to have included a dot plot to show the distribution of coin weights [Wil99]. More than a century later, Cleveland also employs the name “dot plot” for a visualization that functions like a bar chart [Cle85]. The difference is that each bar is replaced by a single dot at the far end to avoid meaninglessly covering an entire area. He also proposed ordering the data items to enhance the perception of the distribution of values, as in the example in Figure I-3. The same author also composed the plots in matrices of small multiples to create “multiway dot plots” [Cle93]. In 1996, Sasieni and Royston [SR96] presented dot plots that use the regular binning of histograms but replace the bars with columns of same-sized dots. Depending on the bin width, the individual dots are hard to recognize.

So far, the mentioned techniques have a common theme: they are based

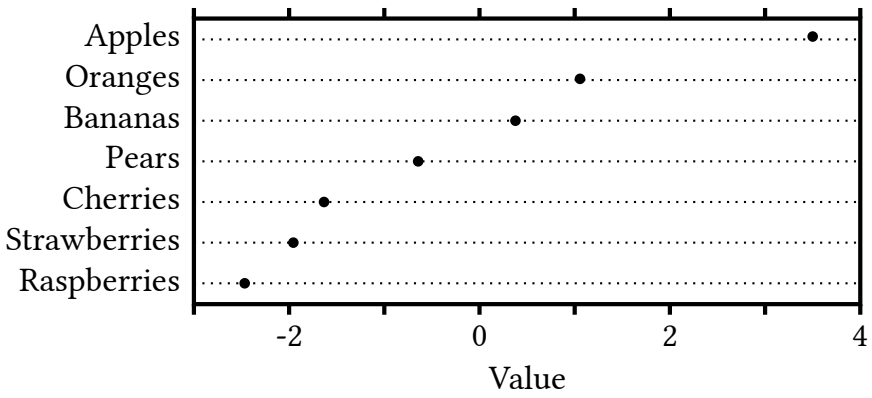


Figure I-3: A "dot plot," according to Cleveland [Cle85]. The dots in each row encode a value. Sorted rows facilitate the perception of distribution. The underlying data is the same as in Figure I-1

on and related to bar charts but use circles instead of rectangles. For our work, we will focus on the definition and algorithm introduced by Wilkinson [Wil99]. His technique yields Figure I-1 when run on the same source data as Figure I-3. We distilled four rules that apply to regular linear dot plots:

DP1 There must be a one-to-one mapping between each data value and each rendered dot.
DP2 All dots are of the same size.
DP3 Lone dots are placed at the exact position of their data value along the displayed scale.
DP4 Colliding dots are stacked into straight columns.

These rules result in properties that distinguish dot plots from histograms and other bar-chart-based visualizations.

More recently, Dang et al. [DWA10] generalized the concept of stacking and produced symmetrical dot plots. They further extended the previously existing technique to create 3D visualizations of 2D data. Their latest advances, however, go beyond the scope of this thesis. The 3D visualization is harder to read and more complex to create in hand-drawn plots, shifting the focus toward a different target audience.

In this part of the dissertation and the included chapters, we often talk about differences, advantages, and disadvantages. Note that we need to stress differences to distinguish between the techniques but do not intend to discredit or look down upon any of the mentioned plot types. They all fit their own specific use cases—often in statistical contexts. Even in other applications, such as landscape visualizations, dot-based visualization can be more memorable and outperform alternative approaches [TSD09].

3

Nonlinear Dot Plots

This chapter is based on previous work and will use extracts thereof without explicit quotation:
N. Rodrigues and D. Weiskopf. “Nonlinear dot plots.” In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 616–625. DOI: 10.1109/tvcg.2017.2744018.

3.1 Motivation

So far, we have established that regular dot plots provide a simple and intuitive visualization for the distribution of data items. They show clusters, gaps, and outliers and are most useful for smaller sample counts. However, they do not scale well for high dynamic frequency ranges in the distribution of underlying data items. The reasons lie within the rules and assumptions behind their layout algorithm. Rule **DP1** is responsible for the intuitive unit visualization. There are, however, multiple implications from rule **DP2** that have negative implications that we will now discuss.

Imagine increasing the number of items for visualization by replicating each point in the source data set. As a result, the sample count rises, but the range of values remains unchanged, just as in the real-world scenario of performing repeated measures to get a robust representation of a distribution. Assuming a constant circle diameter, the columns in the plot will grow linearly in height as the data set size increases. But the width is only affected by the range that the data points cover on the x-axis. The plot will need to be rescaled to fit the computer display or sheet of paper, and the aspect ratio will shift toward a narrow visualization. To address this issue, we need to reduce the diameter of the used circles. This will reduce overlap and produce a more fitting aspect ratio.

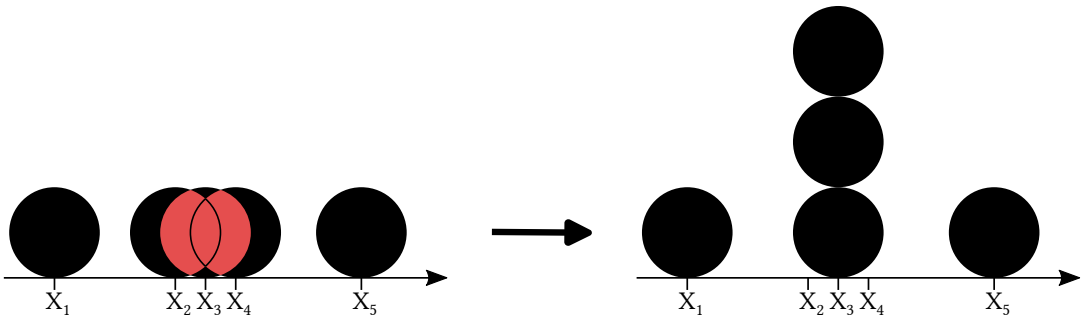


Figure 3.1: Illustration of Wilkinson's sweep layout algorithm [Wil99]. Adding dots for X_2, X_3, and X_4 would create overlap (marked red). Therefore, dots for these values are stacked. X_1 and X_5 can be plotted without any issues.

With a shrinking diameter, the overall shape of dense areas in the distribution remains visible, but outliers can become too small to perceive. Therefore, the selection of a suitable dot size is key to creating a usable visualization. Wilkinson observed an analogy between circle size in dot plots and the bin width in histograms. Similarly to data-dependent suggestions for the latter, he presented a dot size that depends on the number of samples in the source data and assumes a normal distribution with none to moderate skewness. His approach yields a steady aspect ratio for normal distribution, but outliers remain too small. The newer method from generalized stacking does not solve this issue either [DWA10]. However, depending on specific parameter settings and the variability in data density, it introduces more overlap or larger empty spaces.

Directly addressing RQ1, our goal within this chapter is to expand on the intuitive original dot plots and adapt them for higher dynamic ranges of data density while preserving the visibility of outliers. A useful aspect ratio requires small dots in dense areas of the plot, while outliers must remain large enough for perception. Therefore, our approach scales the circle diameters as the number of dots in a stack increases, yielding *nonlinear dot plots*. While the concept of nonlinear scaling has already been used with other visualizations [LA94], it is new for dot plots.

3.2 Technique

3.2.1 Original One-Way Sweep

We use a simplified version of Wilkinson's original algorithm for linear dot plots [Wil99] as a starting point for our extension to nonlinear dot plots. For

Algorithm 3.1: Wilkinson's original sweep algorithm

Input: Data set X. Dot diameter d.

```
1 Sort X ascending
2 while X ≠ ∅ do
3 |   X_i ← lowest non-placed value from X
4 |   c ← 1
5 |   while |X_i − X_{i+c} | ≤ d do
6 |   |   c ←c +1
7 |   Place c dots with diameter d in a column above (X_i − X_{i+c} )/2
8 |   Mark X_i to X_{i+c} as placed
```

a self-contained description, we briefly summarize the original algorithm without smoothing and lateral offsets in this section.

Algorithm 3.1 performs an upward sweep through the data to place the dots from left to right. It assumes that the data samples are given as data points with corresponding data value X_i, with index $i = 1, 2, \ldots, n$ for n samples in the entire data set. It depends on the distance d, which is the (constant) diameter of the dots for the creation of columns. The key point is that the algorithm keeps incrementing the number of dots, c, for the current column as long as further dots still overlap with the column (lines 5 and 6).

To accommodate large, high dynamic range data sets and preserve the advantages of dot plots, the diameter has to be varied in the same plot: the higher the dot column, the smaller the rendered symbols. This leads to a nonlinear modification in column height and width, which is not very intuitive to interpret. In a histogram with nonlinear scaling, we would only have to measure the height and know the transformation function in order to calculate the represented value. With nonlinear dot plots, we also have to factor in varying column widths, making the computation of displayed value densities more difficult (see Section 3.2.5). However, the total size of the plot can be reduced while retaining large dots for outliers. The decrease in height also provides enough flexibility to bring the output closer to the optimal aspect ratio.

We first describe an extended two-way sweep algorithm that allows us to work with varying dot sizes (Section 3.2.2). Then, we discuss the merging of intermediate results of the two-way sweep (Section 3.2.3), models for the data-driven adaptation of the dot diameter (Section 3.2.4), resulting envelopes (Section 3.2.5), and aliasing from rendering dots (Section 3.2.6). The section closes with extensions and variants of the visualization, including color coding and overlays with other diagram components (Section 3.2.7).

Algorithm 3.2: Adaptation for symmetric nonlinear dot plots.

Input: Data set X. Scaling function $d_c(c)$ for stacked dots.

1 Sort X ascending
2 **while** $X \neq \varnothing$ **do**
3 $\quad X_i \leftarrow$ lowest / highest non-placed value from X
4 $\quad c \leftarrow 1$
5 $\quad$ **while** $|X_i - X_{i \pm c}| \leq d_c(c)$ **do**
6 $\quad\quad c \leftarrow c + 1$
7 $\quad$ Place c dots with diameter d_1 in a column above $(X_i - X_{i \pm c})/2$
8 $\quad$ Mark X_i to $X_{i \pm c}$ as placed
9 Do one upward and one downward pass
10 Use average of upward and downward pass

3.2.2 Two-Way Sweep Algorithm

We present a summary of our new two-way sweep in Algorithm 3.2. The modifications with respect to the original Algorithm 3.1 are highlighted.

To adapt the dot size, we have to replace the constant diameter from Wilkinson's original algorithm (line 5) with a data-dependent variant:

$$d_c(c) = d_1 \cdot g_c(c) \ , \tag{3.1}$$

where the *index* c denotes a function that performs the mapping of a dot count within a column. The variable c (not indexed!) represents the current number of data points in a column, and d_1 is a given start diameter that facilitates an overall scaling of all dots and remains constant during the entire layout pass. The function $g_c(c)$ should be weakly monotonically decreasing, i.e., dots should become smaller for columns with more data points. Also, we should have $g_c(1) = 1$ so that we obtain the diameter d_1 if just a single dot is placed. Section 3.2.4 discusses concrete examples of $g_c(c)$. With our approach, the original linear dot plots are included as a special case when $g_c(c) = 1$ for all c. Therefore, all improvements from the new algorithm that are not related to scaling carry over to the linear variant.

In general, scaling compresses the height range but also increases the cognitive load for reading exact values [Hla+13; Tuf83; CM85; BDJ14; Arb+17]. Please note that—as per Equation (3.1)—we keep diameters within each stack constant. As a result, our technique provides a partially linear representation that preserves rule **DP1** locally to aid comprehension.

As indicated in Algorithm 3.2, we can use the data-dependent dot diameter (Equation (3.1)) within Wilkinson's original single-sweep algorithm. However, while this single-sweep approach works very well for a constant dot size, it exhibits some issues when the data value density decreases along the sweep direction. The underlying reason is that shrinking diameters affect column width and, in consequence, they change which dots are to be stacked together.

To illustrate this problem, we will assume that the data consists of a dense, sorted group of values X. The first and last values are within the initial dot diameter, which would lead to rendering a single column for linear plots: $X_n - X_1 \lessapprox d_1$. During the upward sweep from the smallest to the largest value X_i, the dot count c_1 in the first stack increases, and the column becomes narrower: $d_c(c_1) \ll d_1$. Only a few values from X remain and create their own column with a small number of c_2 dots with a size close to the initial diameter: $d_c(c_2) \lessapprox d_1$. Since all values have less than d_1 distance, there will be overlap between the two columns' dots.

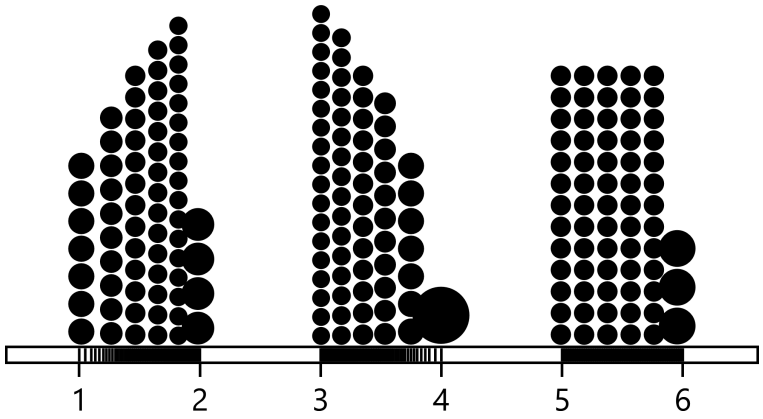
The inverted case, however, presents no problems: When the density increases along the sweep direction, the columns become narrower. Therefore, a second column will not be as wide as the first one, and there will not be any overlap between them.

This observation leads to a solution to the problem: we use two sweeps—one in each direction—and combine their results by averaging. Algorithm 3.2 indicates the two sweeps as *upward* and *downward* passes. Please note that c remains a positive number of dots in the current column, regardless of the sweep direction. Section 3.2.3 describes in more detail how the two sweeps are combined.

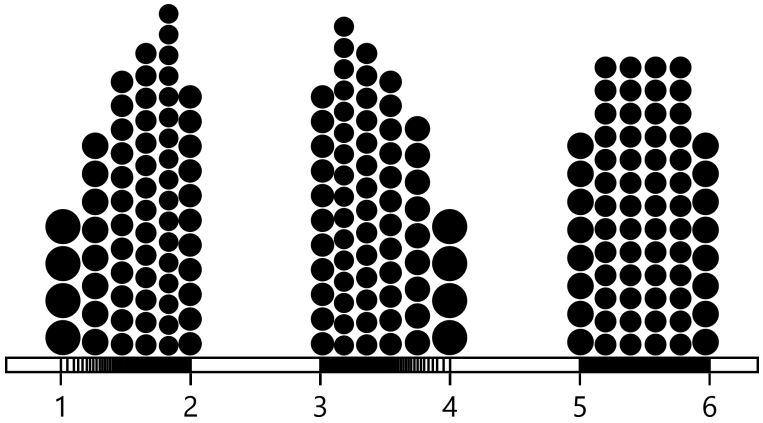
The original sweep algorithm only needs one pass over the source data. Since none of our alterations are of higher complexity than $O(n)$, our proposed algorithm still has the same linear runtime complexity (with a factor of roughly 2 for the number of computations).

3.2.3 Combining Sweeps

Dang et al.'s greedy version of a dot plot algorithm [DWA10] was designed to achieve symmetrical diagrams. While it is preferable to create symmetrical visualizations when using symmetrical data, the algorithm either introduces severe overlaps or gaps between dot stacks (depending on the chosen factor for h in step 1 of their algorithm). As already indicated in Section 3.2.2, a single sweep direction creates overlap problems for nonlinear dot plots, too. In addition, it also causes asymmetry under certain conditions.



(a) Single left-to-right pass



(b) Two opposed passes combined

Figure 3.2: Two dot plots of the same data with varying value density: rising from 1–2, falling from 3–4, and constant between 5–6. Vertical lines on the bottom axis show the individual data value positions. Merging two passes results in a plot with less overlap and more symmetry (b).

To address these problems, we decrease the sweep direction's influence on the final layout by combining both directions, as described in Algorithm 3.2. The open question is: how are the results of two opposing sweeps adequately combined?

As we will explain later in this section, the rules we imposed on the dot diameters (Section 3.2.2) guarantee that both sweeps return the same number of stacks. We can, therefore, define a one-to-one mapping of columns from one layout pass to the other. Columns resulting from a single sweep are defined by three values: a position in the data dimension as well as the number and diameter of the dots. For the merging of stacks from opposite sweep directions, their positions can be readily averaged by using their arithmetic mean. However, it is not straightforward to merge the actual dots if the two columns contain a different number of them: the arithmetic

mean of the number of included dots might contain “half” points, but we can only draw entire dots.

We choose to only draw entire dots by rounding off the number of dots in the current column and then carry the remainder to the following column. In total, this will result in the same number of dots as from a single pass, and the remainder is only distributed within a neighborhood of columns with less than d_1 distance. If the neighboring column—that received the remainder—were farther apart than d_1, it would belong to a different cluster of columns. A distance between data points greater than d_1 would not allow for overlap, thus, would not require stacking of dots that, consequently, cannot interfere with each other in the sweep algorithm. In a nutshell, dots that are farther than d_1 apart can never appear in the same column!

The arithmetic could be applied to the column’s positions directly and their height (number of dots) with the slight modification above. However, the arithmetic mean is not suitable for the diameter of rendered dots. This is an implication of the nonlinear dependency between c and $d_c(c)$ in Equation (3.1). Instead, we calculate the correct diameter from the actual number of dots in each column by applying Equation (3.1) to c after the number of dots is averaged.

There remains one prerequisite for the two-pass method to work: both passes must return the same number of columns. This is important because the first column of the upward pass has to be averaged with the last column of the downward pass. If they do not match, the data values of different neighborhoods will be merged. Now we analyze the scenarios encountered during the layout process to check whether they meet the mentioned prerequisite. Our first observation is that data values that are further apart than the start diameter d_1 divide the data samples into clusters that can be treated separately. Therefore, we can restrict the discussion to a single cluster of data samples.

Clusters that produce a single column in one direction will also produce a single column in the opposite direction, i.e., this is a trivial case. Let us assume the upward pass A resulted in two columns and look at the following two scenarios.

Scenario 1: If the downward pass B returned a single column, it would mean that the total number of values in the cluster was so small that $d_c(c)$ was still larger than the distance between the first and last data value. This would, in turn, also mean that pass A would have returned a single column containing the complete cluster, which is in direct contradiction to our premise.

Scenario 2: Can pass B return three columns? If it started at the same data point, where pass A finished, then its first column would cover at least

the same amount of data values as the last column of A. That would leave the same data that pass A used for its first column, which in turn would also only result in a single column of pass B. Therefore, three columns could only be generated if pass B encountered additional data values outside the cluster. However, we only look at the data inside the cluster because different clusters are so far apart that they do not interfere with each other.

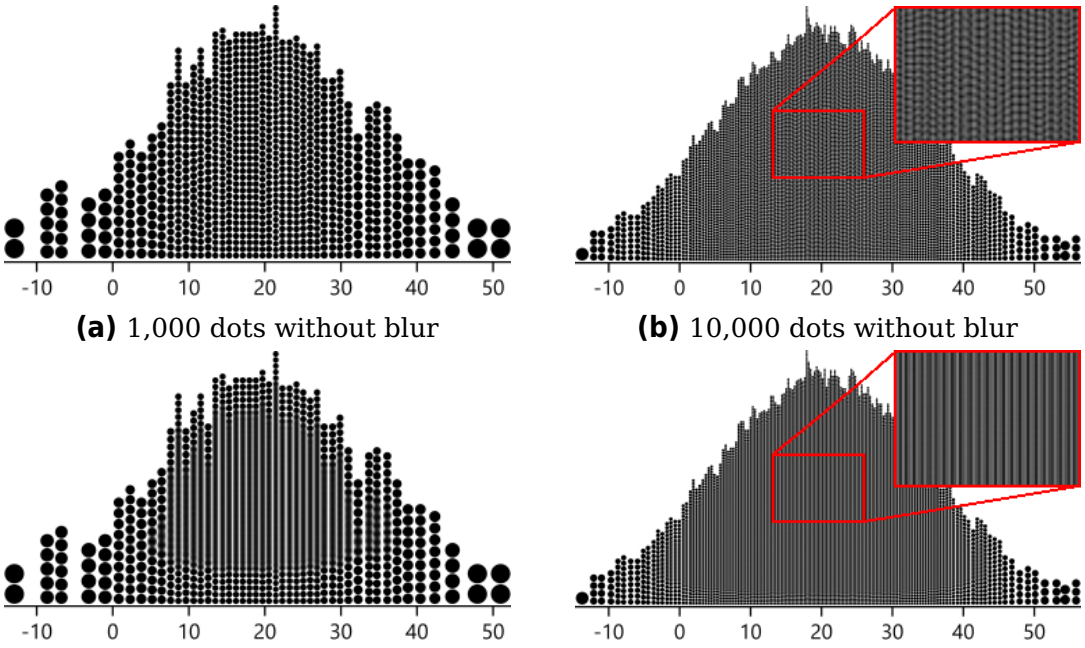
According to this logic (similarly to mathematical induction), the two opposed passes cannot result in a different dot stack count, which makes our averaging method applicable to arbitrary source data.

As shown in Figure 3.2b, the two-way layout leads to columns that are better centered on contained data and have less severe overlaps than the single-pass algorithm (see Figure 3.2a). This makes our algorithm well suited not only for nonlinear dot plots but also improves the layout of traditional, linear plots. Figure 3.2b also shows that a constant sample density does not necessarily lead to a constant dot stack height, which makes the visualization inconsistent with the underlying data. This is due to rendering individual dots, which leads to quantized column heights. In order to get a constant height among all columns of the third cluster, we have to fit the initial dot diameter and the nonlinear transformation to the specific samples. While these adapted parameters will fit the targeted cluster, they might be wrong for the clusters at 1 and 3. Therefore, it is not trivial to find a globally optimal solution.

It is not even clear how to completely define the optimality of a solution. We might define an optimal solution as one in which the distance of rendered dots from their input values (layout error) is minimal. If we then selected a sufficiently small dot diameter, we would render each dot at the exact data value position and have an optimal solution according to that metric. However, the resulting plot would not be readable because the dots would be too small to perceive. Therefore, an objective function for an optimal solution would have to consider the layout error but also the dot sizes, display resolution, viewing distance, and aspect ratio. We leave the definition of such an objective function as an open question for future research.

3.2.4 Dot Diameter

The above two-way sweep algorithm makes heavy use of the dynamic adaption of dot size according to Equation (3.1). We will now discuss useful choices for the adaptation model, as formalized by $g(c)$. For this, we assume that we have a rather dense packing of dots so that the dot plot resembles the corresponding histogram.



(a) 1,000 dots without blur

(b) 10,000 dots without blur

(c) 1,000 dots with blur

(d) 10,000 dots with blur

Figure 3.3: Low-resolution plots of normal distributions. Column-oriented anti-aliasing dampens moiré patterns in visualizations (c) and (d). When used with large dots, this blurring introduces unwanted optical effects (c).

Although $g_c(c)$ controls the nonlinearity of the dot plot, it is not identical to the nonlinear mapping known from histograms or other function plots. It is important to note that $g_c(c)$ does not map the original height of a stack to the nonlinear modification. For example, we cannot simply use $g_c(c) = \log(c)$ to obtain the analog of a log-scale dot plot. In fact, the height of a column with c dots is:

$$h_c(c) = c \cdot d_c(c) = c \cdot d_1 \cdot g_c(c) \ . \tag{3.2}$$

As noted earlier, we require that $g_c(1) = 1$ and that $g_c(c)$ decreases with increasing c. In addition, we want to guarantee weak monotonicity of the plot: a column with more data points should never be smaller than a column with fewer points, i.e.,

$$h_c(c_1) \leq h_c(c_2) \quad \text{if} \ \ c_1 < c_2 \ . \tag{3.3}$$

The extreme case of constant height is obtained for:

$$g_c(c) = \frac{1}{c} \ , \tag{3.4}$$

i.e., this choice leads to $h_c(c_1) = h_c(c_2) = d_1$. As shown later in Figure 3.4c, there are some applications for this extreme model. The corresponding

visualization resembles a combination of jittered strip charts and stripe charts [Cha+83] (see Figure 3.4c). Typical nonlinear mappings target strong monotonicity. To this end, the dot size needs to shrink less quickly than in Equation (3.4). A corresponding mapping is achieved by:

$$g_{c,r}(c) = \frac{1}{c^s} \ , \tag{3.5}$$

with an additional parameter s that controls the shrink rate. The useful range for the shrink rate s is between 0 and 1. Selecting a shrink rate of $s = 0$ will create traditional linear dot plots; $s = 1$ yields the earlier model of Equation (3.4). For in-between shrink rates, the column height is proportional to c^{1-s}, with the fractional exponent $1 - s$ corresponding to a root. We mark functions related to this root scaling with index r. For images within this chapter, we select a default value of 0.4, which leads to a column height of $c^{0.6}$. This choice leads to good results for the upcoming example plots but could certainly be replaced with other values. We added a table with varying d_1 and s as supplemental material to the original publication [RW18] to illustrate the effect of these parameters on the final layout.

The root mapping in Equation (3.5) might be good for many data sets, but it will not fit most processes in nature that exhibit exponential growth, where the growth rate is proportional to the current function value. Examples of such can be found in nuclear decay, the Weber-Fechner law [Fec60], and population growth. The existence of exponential processes is also reflected in the way we represent floating point numbers: mantissa and exponent. To get a log-scale mapping, we naively try to approximate

$$h_c(c) = \log(c) \ . \tag{3.6}$$

Based on Equation (3.2), we would calculate

$$g_c(c) = \frac{\log_b(c)}{d_1 \cdot c} \tag{3.7}$$

for any desired base b, but this would violate the requirements for single dot columns, as $\log(1) = 0$. To get $g_c(1) = 1$, we could change the numerator in Equation (3.7) to

$$\log_b(c) + d_1 \ .$$

This, in turn, would lead to other problems. For instance, with $b = 2$ and $d_1 = 0.5$, the diameters in columns with two dots would be larger than a single dot ($g_c(2) = 1.5$). Instead, for logarithmic dot plots (index l), we propose

$$g_{c,l}(c) = \frac{\log_b(c + b - 1)}{c} \ , \tag{3.8}$$

as it fits all requirements when used with $b \geq \frac{\sqrt{5}+1}{2}$ (the golden ratio). The supplemental material of the underlying publication [RW18] includes a table with varying d_1 and b to illustrate the effect of these parameters on the layout.

The final issue is the choice of d_1. This parameter should be chosen according to the number of data samples, their distribution, the size of the plot, and—very importantly—the targeted aspect ratio. We iteratively optimize for d_1 with nested intervals: first, we compute a plot with the current value of d_1; then increase it in the next iteration if the current aspect ratio is wider than desired and vice versa until we reach a ratio that is sufficiently close to the target.

3.2.5 Envelope

The above discussion holds for the mapping of height, i.e., a single dimension. Now, the dot plots intrinsically link column height and width because both dimensions are determined by the dot diameter. If the diameter is scaled by a factor a, the covered area (i.e., the area of the column) is scaled by a factor a^2. Therefore, a square root computation needs to be included in all mappings if the adjustment is meant for areas, not just height.

Based on these observations, one can consider the limit case of a very large number of dots and the envelope of the nonlinear dot plot. For the case of root dot plots according to Equation (3.5), the height of the envelope scales with $\alpha^{(1-s)/(1+s)}$ if the number of dots is multiplied by α. Thus, the exponent s from the diameter scaling matches an exponent of $(1-s)/(1+s)$ in the corresponding nonlinear histogram.

3.2.6 Anti-Aliasing

In general, dot plots can come with high demands regarding rendering quality because they consist of clearly defined dots with sharp boundaries. Such boundaries, in combination with some rather regular placement of dots, can lead to aliasing and moiré artifacts [Gla69; SW82]; see Figure 3.3. Small differences in dot sizes are the main cause of moiré effects. These accumulate along the height of neighboring columns and create virtual tilted lines.

Typical anti-aliasing approaches from computer graphics work with supersampling on the image plane, followed by low-pass filtering and downsampling. Our solution adopts the same strategy but exploits the special characteristics of dot plots. In the rendering stage of the visualization pipeline, low-pass filtering blurs the image, i.e., the individual dots would eventually disappear, and only a solid colored area would appear. Other

than generic anti-aliasing techniques, we limit blurring to the vertical direction because individual columns should still be distinguishable. Furthermore, a few dots at the top and bottom are left unchanged, as they play a key role in estimating the size of individual dots and in comparing column heights.

We also decided on not blurring those columns that are not surrounded by others because they do not add to the moiré effect. Finally, we only start blurring after the dot count inside a column exceeds a certain lower threshold (in our examples: 12) because otherwise, the rendering does not create an area that is big enough for the effect to be perceivable. Figures 3.3c and 3.3d show examples of anti-aliasing. Dots in a blurred line are not countable anymore. However, as the height of columns increases, and the dot diameter decreases, it becomes more and more difficult to make out individual dots for counting, anyway. We use our anti-aliasing method with moderation in order to balance advantages and side effects. The vertical lines in Figure 3.3c are an example of overly aggressive blurring for the low dot density that trades the moiré pattern for even worse optical effects, especially when rotating the image. Therefore, we recommend anti-aliasing only for plots with very small dots, as in Figure 3.3d.

3.2.7 Variants, Extensions, and Hybrid Visualizations

Just like conventional dot plots, our nonlinear generalization can be widely applied to depict any kind of data distribution. Similarly, it can be combined with other visual mappings to include further information or emphasize certain aspects of the data.

One example is additional color mapping. In general, color plays an important role in visualization because it can show additional data attributes on top of the positional variables of the diagram. We argue that color mapping is especially useful in the context of dot plots because each single data sample generates exactly one dot, i.e., we can have a direct mapping between sample and color. To make use of perceptual grouping by color (hue), similar colors should be spatially grouped in the dot plot. We cannot change the layout between the columns in the dot plot because they are driven by the distribution of data values. However, we may modify the order of dots within a column. Therefore, the dots in each column should be ordered vertically according to the additional data attribute mapped to color. A typical example is a chronological data distribution, i.e., a data set with samples that not only carry some data value but also a timestamp. Such time-series distributions are best ordered chronologically in each dot column. Another possible application of color is the comparative visualiza-

tion of several data sets integrated into one dot plot: the color indicates the data sources. The figures in Section 3.3 use such colored dot plots.

We mostly render our dot plots with the stacks aligned to the x-axis. It is possible, however, to center the columns vertically and create diagrams with a horizontal symmetry axis (like Wilkinson's symmetrical dot plots [Wil99]). Further variations can align dot stacks to the y-axis. This is especially useful when dealing with nominal data as it improves the layout of labels and creates an output similar to Cleveland's multiway dot plots [Cle93]. In addition to layout and rendering variations, we can combine different visualization methods. For instance, Figure 3.6b shows a symmetrical dot plot overlaid by a box plot, whereas Figure 3.2 adds strip charts to the x-axis. Such hybrid diagrams can help users classify and identify data in meaningful ways, providing more insights. Tukey's suspended rootograms plot data in relation to a known density distribution [Tuk72]. The same technique could be applied to the vertical positioning of the dot stacks to show deviations but would require special care to adjust for areal distortions (see Section 3.2.5).

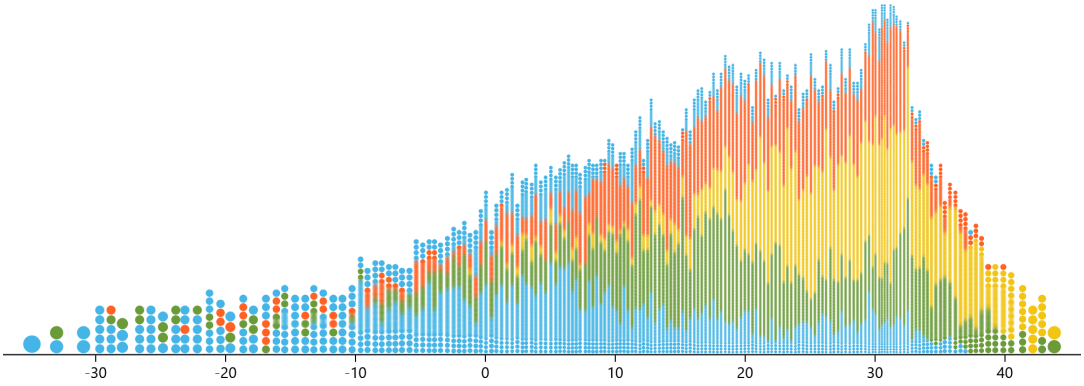
3.3 Examples

We demonstrate nonlinear dot plots for typical examples of real-world data. These contain frequencies of varying ranges. We include comparisons to histograms and linear dot plots to examine different characteristics.

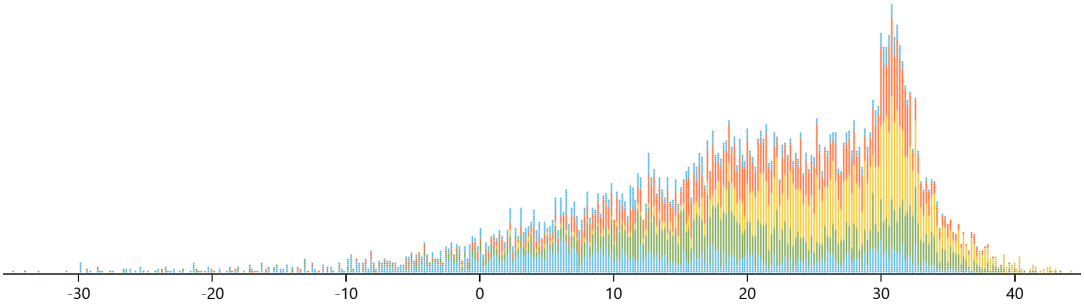
3.3.1 Distribution of Air Temperature

Air temperature has a direct impact on our daily life, but it is also related to issues of global climate change. Therefore, we pick this application as our first example. In 2017, we downloaded a data set from the "Climate Data Centre" of the German meteorological service (Deutscher Wetterdienst). It contains monthly averages of maximum daily air temperatures by weather station for the years 1961 to 1990. The stations are labeled using the identifiers supplied by the World Meteorological Organization (WMO). This data set provides a total of 9685 data samples from 875 weather stations worldwide; temperatures are in degrees Celsius. The data is still available online, albeit in a different format and with more samples [Deu].

Visualizing this data with the plots in Figure 3.4, one can immediately see that the distribution is unimodal and has its maximum between 31 to 32 degrees. While mean temperatures down to around -30 degrees are still quite common, only single instances of temperatures at or below -31 degrees can be observed. Temperatures between 16 and 29 degrees

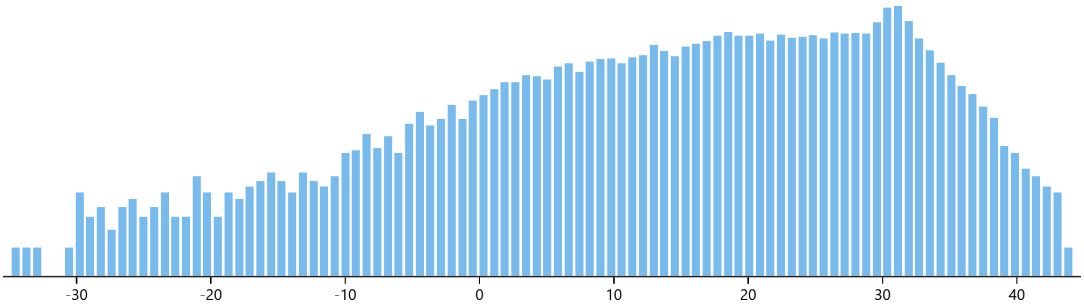


(a) Root dot plot



(b) Linear dot plot

(c) Strip dot plot (root plot with $s = 1$)



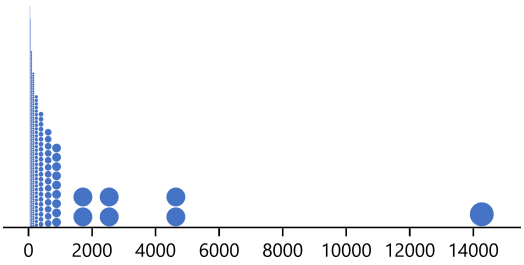
(d) Logarithmic histogram

Figure 3.4: More than 9,000 air temperature values (in degrees Celsius). The data set shows the monthly average of daily maxima measured by weather stations all around the world. Individual dots in (a) and (b) are colored according to the month: J F M A M J J A S O N D. We use black dots in plot (c) to maximize the perceivable range of brightness.

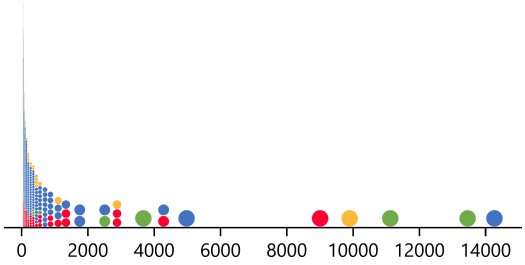
are almost evenly distributed, forming a plateau in the linear dot plot. This plateau is also visible in the logarithmic histogram and the root dot plot, albeit less noticeable. The nonlinear plots outperform their linear counterparts when we focus on less dense areas of the plots. Figures 3.4a and 3.4d both show minimum and maximum values (outliers) clearly, but only the dot plot allows counting them. The histogram would need a fine-grained vertical axis with ticks at intervals of 1, which would lead to overplotting. The extreme, strip chart-like visualization in Figure 3.4c only shows the outliers clearly but turns the dense regions into light-gray areas. Cross-referencing the temperature data with the WMO database yields additional geographic information. Picking data points in nonlinear dot plots for cross-referencing is very simple because of the one-to-one relationship between dots and data points and since the low-frequency points are rendered as quite large dots.

While the two nonlinear visualizations look similar at first glance, the dot plot provides more detail in dense areas. The histogram cannot provide such a view, as all its bins are of equal width. Figures 3.4a and 3.4b also exhibit some gaps in the stacks (for instance, near −4 °C), which contrast the tightly packed neighboring areas. This is due to the characteristics of the data source: the temperatures are rounded to a single decimal place, and there are many data points with the same value. The value density in this area is too low to create high and narrow columns, but at the same time, it is too high to place a wider column at each decimal of a degree. Therefore, the layout algorithm cannot create a tightly packed field of uniform columns, which leads to visible gaps.

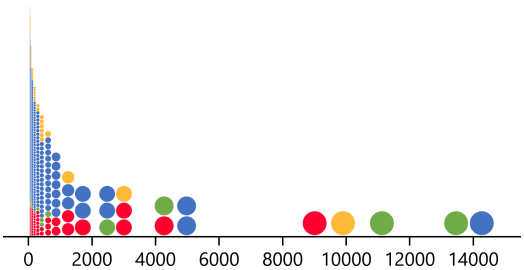
To encode more information, we colored the individual dots in Figures 3.4a and 3.4b according to the four seasons. We compensated the phase shift between the Earth's northern and southern hemispheres by adding six months to data from weather stations below the equator. Using this colorization with the nonlinear dot plot, it is immediately noticeable that higher temperatures tend to occur in summer, while the lower ones are predominantly measured in winter. That is no surprise; however, the tendency is not so obvious when looking at the linear dot plot. This effect can be explained by the unimodal sample distribution that has a lower value frequency at the outer edges and the bias toward outliers in Figure 3.4. It would be possible to color the histograms analogously to the dot plots by subdividing the bars. However, there is no appropriate color coding for the logarithmic histogram because there is a conflict between the linear (proportional) splitting of individual bars versus the overall logarithmic scaling. There is no such ambiguity with the colored dot plots, as the individual stacks are always linear, giving the users an impression of the distribution of sample categories within the stacks.



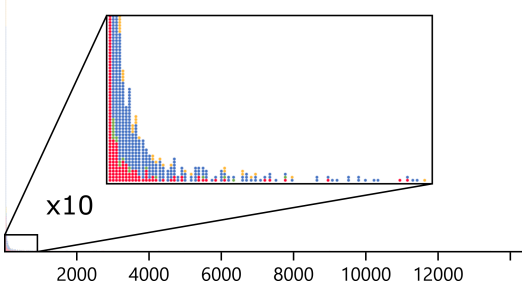
(a) Log_2 dot plot using $d_1 = 795$ (Shneiderman only)



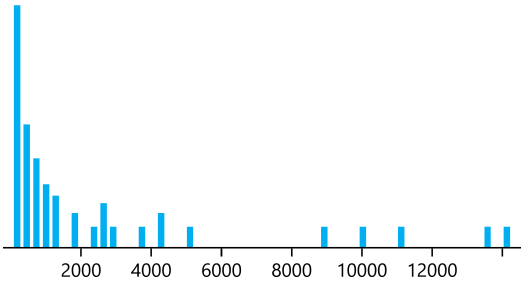
(b) Root dot plot using $s = 0.6$ and $d_1 = 526$



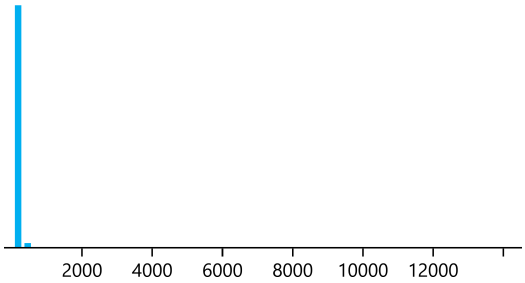
(c) Log_2 dot plot using $d_1 = 795$



(d) Linear dot plot using $d_1 \approx 8.4$



(e) Logarithmic histogram



(f) Linear histogram

Figure 3.5: Number of citations for papers by Ben Shneiderman, William S. Cleveland, Leland Wilkinson, and William E. Lorensen. Each dot represents one of 1,463 publications from these authors.

3.3.2 Citation Statistics

Our next example shows bibliometric data in the form of citation statistics. The h index [Hir05] is a popular indicator of publication impact by an author, heavily aggregating data about all papers into a single number. In contrast, visualizing the complete citation data presents a challenge, as the number of citations may vary extremely per author and paper: there tend to be few publications with a big impact (i.e., many references) and many papers that hardly anyone notices (i.e., little to no references). Therefore, we obtain a high concentration in the frequency plot near zero and some outliers with many citations, forming a long tail. Since these outliers are the most relevant publications, the visualization should represent them

accordingly. However, even the bulk of low-citation papers is interesting because it indicates publication productivity.

For illustration purposes, we use citation data of four well-known researchers (Shneiderman, Cleveland, Wilkinson, and Lorensen) obtained from *Google Scholar*[1] through the *Publish or Perish* software [Har16] on June 22, 2017 (without any data cleansing). Figure 3.5 shows the results. In 3.5a, we plotted 1,045 publications by Ben Shneiderman. His most cited work is "Designing the user interface: strategies for effective human-computer interaction" (14,309). There are six additional publications that are clearly distinguishable, but most of the other papers seem to have one thousand or fewer citations.

To compare his citation data to that of the other three authors, we add their data and use color-mapped dots. By the dominant dark blue color in the nonlinear dot plots in (b) or (c), we can immediately see that Shneiderman has the largest number of publications. Dots below about 1,000 citations become too small to distinguish individually, but their nonlinearly scaled column heights can still be perceived; therefore, we can obtain the approximate frequencies and compare them between authors. The much fewer papers with high citation counts are large and clearly visible. From these dot plots, we can see that all four authors have publications with 9,000 or more references. Bill Lorensen even has two papers with very high citation counts: "Marching cubes: A high resolution 3D surface construction algorithm" (13,495) and "Object-oriented modeling and design" (11,147). Leland Wilkinson's most cited work is "SYSTAT for Windows: statistics, graphics, data, getting started, version 5" (9,914). The red dot at 9,017 represents William Cleveland's "Robust locally weighted regression and smoothing scatter plots."

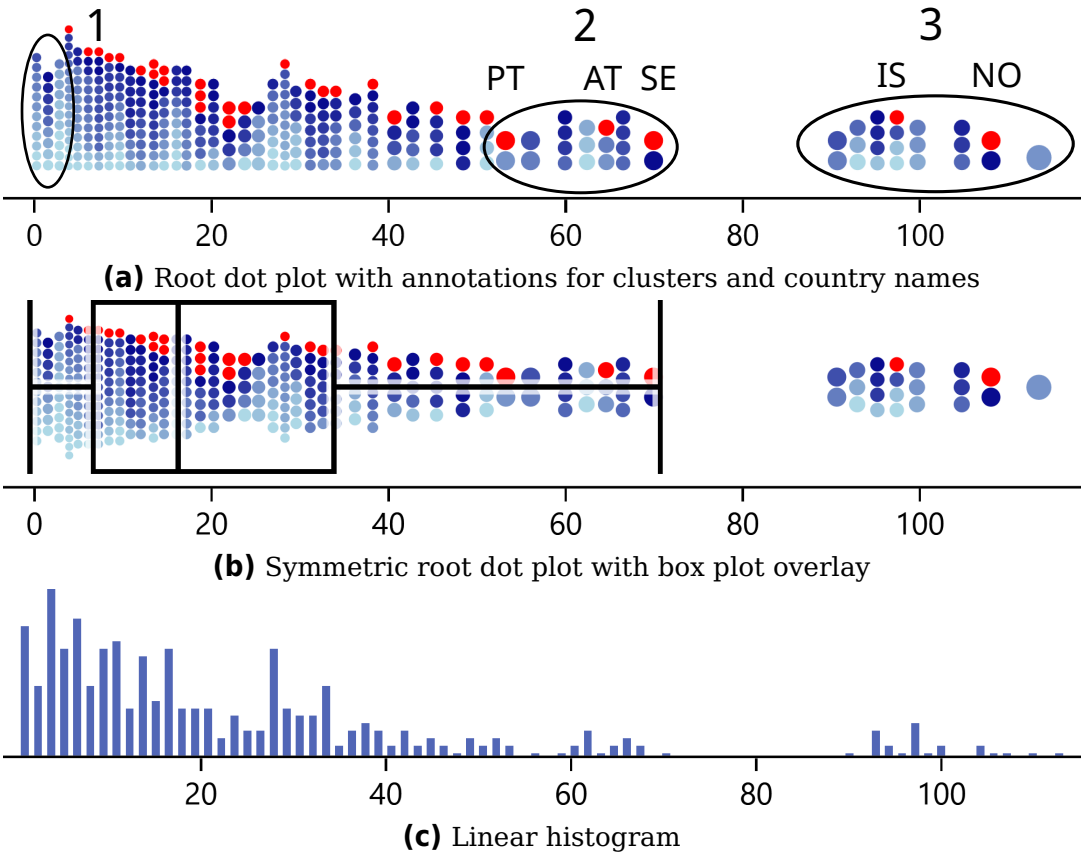
In contrast, linear plots (Figures 3.5d and 3.5f) are not well suited for such high-dynamic-range data, as they cannot show any useful information about the important long tail. The logarithmic histogram in (e) renders the highly referenced publications as bars. This allows for a relatively accurate estimation of the citation counts but does not provide any method of showing and comparing the authors.

3.3.3 Renewable Energy

Our third example combines dot plots with box plots to address the way in which electricity is produced. The European Commission provides the general public with access to statistical data through its website[2]. It includes

[1] `https://scholar.google.com`, accessed 2024-01-27.

[2] `https://ec.europa.eu/eurostat/web/main/home`, accessed 2024-01-27.



(a) Root dot plot with annotations for clusters and country names

(b) Symmetric root dot plot with box plot overlay

(c) Linear histogram

Figure 3.6: Percentage of electricity produced from renewable energy sources versus total consumed electricity of 30 European countries from 2004 to 2014. There is a value for each country and year. Dot plots use a color scale to represent the value's year: '04 '05 '06 '07 '08 '09 '10 '11 '12 '13 '14.

information on the amount and type of energy produced in each country of the European Union (EU), as well as Norway and Iceland. The data set "tsdcc330"[3] compares the generated renewable electricity with the total consumed amount on a yearly basis from 2004 to 2014.

Visualizing this data set with dot plots and histograms, three groups of values stick out that can be interpreted by combining the use of plots with the tabular representation of the source data. As Figure 3.6 and, more specifically, cluster 1 in Figure 3.6a show, there is a relatively high data density near 0 %. Temporal information from the dot plots shows that it must have been from countries that have only recently started producing electrical power from renewable sources in relevant amounts. This conclusion is evident because there are stacks around zero percent, but red dots (indicating the data set's last year) are no longer present.

[3] `http://ec.europa.eu/eurostat/tgm/download.do?tab=table&plugin=1&language=de&pcode=tsdcc330`, accessed 2016-08-01.

A second cluster from Figure 3.6a is less visible in the histogram (c). The red dots indicate that it is composed of values of approximately three countries. Assuming a general trend of increasing the amount of renewable energy (in order to meet climate protection goals), it seems probable that the country with the leftmost red dot has the lowest share of the cluster. A look at the tabular data confirms this hypothesis: a single dot is contributed by Portugal, while Austria and Sweden make up the remainder of the cluster.

Cluster 3 from Figure 3.6a is well separated from most of the other values and is well discernible in all three plots. The overlaid box plot in (b) explicitly classifies the entire cluster as consisting of outliers. At first glance, there are two countries that mainly use renewable energy. A closer comparison and the dot count show that each color is present exactly twice, further indicating that all values are only from those two countries. Cross-referencing with tabular data identifies them as Norway and Iceland.

Due to the lack of countability, the regular histogram was not as useful in arriving at these conclusions. The dot plots show individual data values that prompt further research into details.

3.4 Expert Review

To evaluate nonlinear dot plots, we conducted an expert review [TM05]. We recruited four experts from the field of visualization and computer graphics from our university. All of them were Ph.D. students with several years of research experience. Our evaluation focused on the visualization itself, not the interactive tool used to generate the plots. Therefore, we presented each of the experts with static printouts on paper. These included four different kinds of visualizations: linear dot plots and histograms, nonlinear dot plots, and logarithmic histograms. Each review took approximately 30 minutes, during which an operator engaged the experts in a dialog (structured by a previously prepared sequence of questions) and took notes.

At the beginning of the review, the expert was introduced to the concept of dot plots and specifically to the nonlinear dot plots and stack sizes. Then, the expert was asked to work with all four plot types; each applied to three different examples of time-series data sets: (1) “FOL_acc_1_1” from the MobiFall[4] [Vav+13; Vav+17] data set, which contains axis-dependent accelerations of people falling, measured with smartphone devices. The visualizations showed 812 samples. All plots were rendered in black, i.e., no additional color mapping was applied. (2) The energy data set discussed

[4] `http://www.bmi.teicrete.gr`, accessed 2016-11-23.

in Section 3.3.3. The dot plots were colored as in Figures 3.6a and 3.6b, whereas the histograms remained black. (3) Temperature data showing minimum daily temperatures as in Figure 3.4. In this case, the dot plots were also colored, whereas histograms remained black. The review session concluded with collecting the expert's general feedback and remarks on the different data representation methods. Although we compare colored and black visualizations, we argue that this is a suitable study setup because it makes use of the most important advantage of dot plots (countability), and we would not want to compare techniques without their main features.

We first summarize our main observations made while the experts worked with the visualizations. For data set (1), all experts recognized a sharp peak near the number 10, but no one made a connection to the gravitational acceleration of approximately $9.81\,\mathrm{m\,s^{-2}}$. An expert with a pronounced background in statistics noticed the connection between dot plots and histograms and found the analogy between dot stacks and narrow bins in high sample frequency areas. In low-density areas, however, the same visualizations diverged as the number of stacks and bars did not match. This was due to the bin limits that separated samples into a bar each, while nonlinear dot plots joined multiple values into low stacks. Everyone noticed the lack of perceivable bars in low-frequency areas in linear histograms (due to rasterization), while some were able to discern individual circles in linear dot plots.

Experts were first introduced to a color scale for the dots when presented with the energy data. The first objective was to find out how many countries were represented in the plot. This was to check whether they understood the meaning of a single dot and the color scale. One expert with much experience in statistics made the connection himself: the number of countries is equal to the number of red dots. The others needed additional explanations but eventually, everyone got a grasp of the concept and answered the next question correctly: There were no more countries without electricity from renewable energy sources in the last year. When asked whether the data presented outliers, all experts mentioned the cluster around 100 % and that, at least in the last year, there were only two countries. The small difference between printed colors and the plot's small size did not allow them to compare individual dots and notice that the cluster always contained two dots per year.

The third group of visualizations came with estimation tasks. The experts should determine the 25, 50, and 75 percentiles in each plot separately. Everyone reported that it was much easier with linear representations than with nonlinear ones, as they had to compensate for representational distortions. This is to be expected and in line with previous findings [Hla+13; Tuf83; CM85; BDJ14; Arb+17]. The estimation results

reflected the experts' statements, although the number of participants was too small to achieve statistical relevance. Two experts with a background in statistics were able to compensate for the distortions more effectively and arrived at virtually the same results for each visualization.

After having gone through three data sets, we asked the experts which would be the first visualization (out of the four available) they would use on completely unknown data. Three preferred the linear histogram, mostly because they already knew it from previous work and were quite familiar with it. One preferred the nonlinear dot plot because outliers were better visible, with and without color mapping.

Finally, we asked for general feedback and remarks on the different data representation methods. The consensus was that dot plots are aesthetically pleasing, especially when combined with color mapping. The nonlinear representation allows for identifying and counting outliers. However, the scaled height introduces distortions for which the observer has to compensate. The experts were intrigued by the possibilities of color mapping and counting individual data and mentioned that it allowed for the extraction of more details from the visualizations.

3.5 Discussion

All dot plots have a wide range of possible uses, similar to other frequency plots. They can be combined with other visualization elements, including color coding and hybrid overlays. Nonlinear dot plots combine the advantages of conventional dot plots and nonlinear scaling known from log-scale histograms. They are suitable as unit visualization for countable samples and data sets with a large range of frequencies alike. With our new type of plot, we are able to visualize data distributions with well above a thousand individual values, as required for  RQ1. Furthermore, nonlinear dot plots allow the viewer to discover outliers because the single data points are drawn as large dots in areas with few samples. In our experience, linear dot plots work just as well if the data sets are small (less than 1,000 values) and the value frequency stays within a small range.

The two-way sweep algorithm has linear complexity. Wilkinson carefully analyzed the properties of dot plots to derive a suitable dot size from the dot count. But his approach did not consider on what scale the data is distributed: two data sets can have the same number of samples but spread over intervals that are orders of magnitude apart with respect to width. In contrast, modern computing hardware can easily perform multiple bidirectional layout runs to find the optimal dot size for any targeted aspect ratio.

It should be noted that the idea of nonlinearly scaling dots is more fundamental than the two-way sweep algorithm we presented. For example, iterative approaches like simulated annealing or genetic algorithms might also prove to give satisfactory layout results. Just as node-link diagrams of graphs can be laid out by force-directed methods [KK89; FR91; Kob12] with randomization elements, the creation of columns could be implemented by shifting dots, merging, and splitting stacks. In fact, we will discuss another iterative approach in Chapter 4.

Our presented algorithm retains the traditional layout of dot columns. On the one hand, this is an advantage as it keeps the complexity low and does not require users to learn new concepts to understand the visualization. On the other hand, our technique introduced moiré effects of tilted lines that require a specific low-pass filter for controlled vertical blurring along the inner parts of the dot columns. In the context of  RQ1, we see nonlinear dot plots as a step in the right direction but require further work to consider the research question sufficiently answered.

In the interest of open science, a sample implementation of the core of our algorithm is freely available [Rod17]. We also supervised a project that resulted in an easily usable software package for the statistical programming language R [DKF23].

4

Relaxed Dot Plots

This chapter is based on previous work and will use extracts thereof without explicit quotation:
N. Rodrigues, C. Schulz, S. Döring, D. Baumgartner, T. Krake, and D. Weiskopf. “Relaxed dot plots: faithful visualization of samples and their distribution.” In: *IEEE Transactions on Visualization and Computer Graphics* 29.1 (2023), pp. 278–287. DOI: 10.1109/TVCG.2022.3209429, and
N. Rodrigues, C. Schulz, S. Döring, D. Baumgartner, T. Krake, and D. Weiskopf. *Supplemental material for relaxed dot plots*. 2022. DOI: 10.18419/darus-3055.

4.1 Motivation

We have advocated dot plots as a form of unit visualization over histograms. We base this on the argument that there is no aggregation into bins. A histogram shows the number of values within a range, but it does not show the local distribution inside the range itself. A data point can be in the center or the edge of a bin, and the histogram will not show a difference. Dot plots can place a point at the exact location of its underlying value. But this advantage only applies to areas of low data density. When we need to stack dots, they fall into straight columns, once again hiding the distribution of values within the axis range that the stacked circles cover. In the previous chapter, we disregarded the original dot plot rule **DP2** (defined on page 19) and applied nonlinear scaling to the diameters. This allowed us to scale the visualization for larger data sets while preserving the visibility of outliers (see left vs. center in Figure 4.1). But the layout is still reliant on columns.

Circle diameters in dot plots are similar to bin widths in histograms and bandwidth in density estimation. Nonlinear scaling decreases the dot sizes

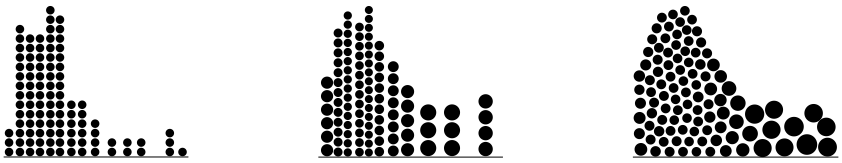


Figure 4.1: A column-based linear layout (left), a nonlinear layout (middle), and our relaxed layout, constrained to a more faithful envelope with locally equidistant dot spacing (right).

in dense areas. As a result, the displayed distribution of density becomes very ragged and contains many spikes.

Nonlinear scaling also worsened the already existing tendency for moiré effects in areas with small dots. A local directional low-pass filter ameliorated the issue but did not provide a fully satisfactory solution because it does not allow for the display of individual dots (rule **DP1**). The problem of moiré effects is systemic for regular layouts in the size range of a few pixels. Increasing the resolution or using vector displays can lessen the issue but will not provide a definitive solution. Ultimately, visualization is produced for human viewers with receptors in the retina. Therefore, the image of any plot is effectively rasterized with a high but limited resolution during perception.

We want to continue the work on dot plots to answer RQ1. In this chapter, we introduce *relaxed (nonlinear) dot plots*. The basic idea is to allow for deviation from rule **DP4** and not enforce straight columns (see center vs. right in Figure 4.1). This gives us more flexibility to place dots more faithfully to their underlying value. *Bee swarm* plots by Eklund and Trimble [ET21] are somewhat similar to what we will present in this chapter. They are a kind of dot plot that places the circles at the exact location of the underlying value without adhering to columns. However, they do not scale the dot size, leading to the same lack of scalability for larger data sets as the original dot plots by Wilkinson.

We want an envelope shape that provides a better display of the value density and prevents the ragged spikes from changing bandwidth. But what is the correct value density? Wilkinson [Wil99] already made the connection between dot plots and kernel density estimation (KDE). However, KDE is usually applied to compute a *probability density function* with a total area of one [Ros56; Par62]. We will adapt this approach for data density ρ in an interval h as corresponding to a data frequency f that may include values >1. As a result, we will find a smooth representation of the underlying data distribution with a constant bandwidth but no ragged spikes. In contrast, bee swarm plots only focus on the correct placement of individual dots without regard to the representation of overall value

density. Depending on the setting of their algorithm, the outer shape of the plot can change between the look of an upward-growing coral or branches hanging down from a tree.

Allowing for deviation from straight columns and enforcing a smooth envelope shape results in an optimized layout with regard to the faithful representation of underlying data. But the visualization will still be seen by humans and might suffer from unwanted perceptual effects. Our work employs constrained Lloyd relaxation [Llo82] to achieve anisotropic blue noise property. The resulting locally equidistant dot spacing will avoid unwanted patterns in the perception of tightly packed circles without the need for directional blur. We do not use the more general Linde-Buzo-Gray algorithm [LBG80] because adding or removing dots would interfere with the concept of unit visualization. Blue noise plots by Onzenoodt et al. [Onz+21] also use blue noise for an aesthetic distribution of dots perpendicular to the data axis. However, the publicly available implementation of their technique allowed for arbitrarily high degrees of overlap and did not show an accurate depiction of data density. In contrast, the dot placement in bee swarm plots is correct but allows for elongated gaps in the layout when there is insufficient space to fit dots without overlap. As a result, there is potential for diagonal moiré effects. An advantage of our relaxed dot plots is an adjustable balance for the trade-off between aesthetically pleasing dot positions and faithful data representation.

To help answer  RQ1, we want to go beyond merely suggesting improvements; we want to quantify the resulting changes. Therefore, we will measure the accuracy of dot positions in relaxed dot plots, present empirical evidence of improved interpretability through a user study, and compare our results with bee swarm and blue noise plots. In the interest of open science, the study data and our algorithm are both freely available [Rod+22b; DBR22].

4.2 Goals and Overview

We aim at more faithful and aesthetic dot plots, by which we understand the following goals: The dots should have **less positional error**, i.e., the difference between the actual display position on the x-axis and the true data value. The shape of the plot should **well-represent (nonlinear) data frequencies**, i.e., peaks, valleys, smooth and frayed features of the outline should match the true frequency distribution. The dot plot should be **space-filling and reasonably well-packed** with little overlap between dots and avoid patterns that are not backed up by data (e.g., moiré patterns).

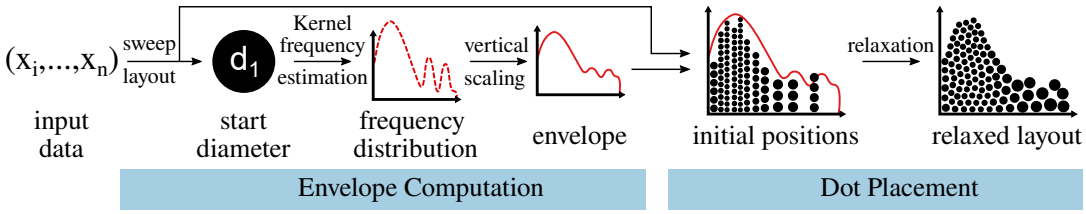


Figure 4.2: Overview of the relaxed dot plot method: first, we compute the (nonlinear) dot sizes and an accurate envelope shape. Then, we compute dot positions, optimizing for proximity to the data value and aesthetics.

To achieve these goals, our method consists of two main stages, as shown in Figure 4.2. Section 4.3 corresponds to the first stage, where we describe how to determine the nonlinear size of dots and the necessary plotting space and shape. Similar to KDE, we use a kernel function to obtain an accurate and smooth estimate of the data distribution because dot plots show *frequency distributions*. To obtain the *envelope*, we vertically scale the distribution to meet the space requirements of the nonlinearly sized dots. The dot scaling for the relaxed layout builds upon that of nonlinear dot plots from Chapter 3 and lifts the binning-like constraints of a column-based layout.

Section 4.4 corresponds to the second stage, where we present the relaxation of dots within the envelope. Here, we elaborate on *placing* the dots closer to their actual data value. The new technique uses the previous two-way sweep from Algorithm 3.2 to obtain initial positions efficiently. Then, we perform a modified Lloyd's algorithm [Llo82] to optimize for accurate placement and to establish the blue noise property.

4.3 Plotting Space, Shape, and Dot Size

To position dots accurately and aesthetically, we first need to determine the *envelope* in which we can move dots. We also have to compute dot sizes for nonlinear scaling.

4.3.1 Kernel Frequency Estimation

We require a distribution shape that faithfully represents the underlying data frequencies. There are multiple possible definitions of the true data frequency—a common problem in statistics—and even more methods. Our approach is based on KDE [Ros56; Par62]. Building upon the same statistical principles helps keep the visualization faithful to the underlying

data—KDE has well-known properties for approximating statistical distributions.

In a nutshell, if $(x_1, \ldots, x_n)$ are the values from the data set, KDE allows us to estimate the probability of a data value being close to x with the use of a kernel K and smoothing bandwidth h. Since we require data frequency instead of probability, we do not normalize the area between the function and the x-axis. Independently from the smoothing bandwidth h, each kernel can have a varying width greater than zero. To avoid having another user-defined parameter, we mandate using an adapted kernel K_h to have the same width as the bandwidth. Thereby, every kernel function produces a non-zero value within $[-h/2, +h/2]$. Outside this interval, it should be zero. The resulting frequency estimation is:

$$f(x) = \sum_{i=1}^{n} K_h(x - x_i) \tag{4.1}$$

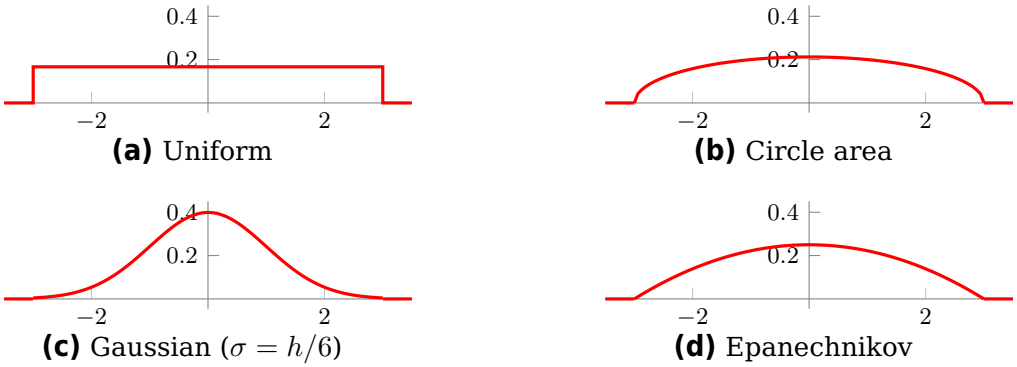
As mentioned previously, the choice of a dot diameter is similar to the bin width in histograms [Wil99]. It is also related to the smoothing bandwidth in KDE, for which there are rules of thumb and complex fitting approaches. Moreover, for frequency estimation, each kernel can have multiple additional parameters. The previous approach to nonlinear dot plots from Chapter 3 allowed for an intuitive selection of an initial dot diameter d_1 by inferring it from the user-desired aspect ratio of the final plot.

Building upon that definition of diameter d_1, we determine the bandwidth and calculate the necessary kernel parameters so that kernel and bandwidth share the same size. Then, we can assume the kernel to be zero at distances larger than the bandwidth. This approach limits computational effort. In our experience, a bandwidth of $h = 2d_1$ is a good compromise between the compactness of the dots and showing the shape of the data distribution. We will discuss the implications of bandwidth for relaxation later in Section 4.4.2.

Our method supports any kernel with an integral of ≈ 1 since we use the frequency to estimate how many data values appear close to a point x in the data domain. Accordingly, each value in the underlying data should contribute approximately a value of 1 to the frequency. The public implementation includes four kernels that are detailed in Equations (4.2a) to (4.2d) and depicted in Figure 4.3.

$$Uniform(x) = 1/h \tag{4.2a}$$

$$CircleArea(x) = \frac{2\sqrt{(h/2)^2 - x^2}}{\pi(h/2)^2} = \frac{8\sqrt{h^2/4 - x^2}}{\pi h^2} \tag{4.2b}$$



(a) Uniform **(b)** Circle area

(c) Gaussian ($\sigma = h/6$) **(d)** Epanechnikov

Figure 4.3: Kernels for frequency estimation, adjusted to have the same width as a single dot (here, $h = 2d_1 = 6$).

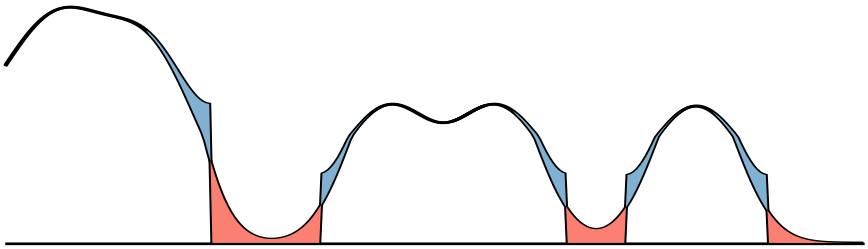


Figure 4.4: Unbounded kernel frequency estimation has unused space (red) from the tails of the kernel. Bounds mirror this space toward the data, making it available for dot placement (blue).

$$\begin{aligned} Epanechnikov(x) &= \frac{1}{2h/2} \cdot \left(1 + \frac{1}{2}\right) \cdot \left(1 - \left(\frac{x}{h/2}\right)^2\right) \\ &= \frac{3}{2h} \cdot \left(1 - \left(\frac{2x}{h}\right)^2\right) = \frac{3 - 12x^2/h^2}{2h} \end{aligned} \tag{4.2c}$$

$$Gaussian(x) = \frac{1}{\sqrt{2\pi\left(\frac{h}{6}\right)^2}} \cdot e^{-\frac{x^2}{2(h/6)^2}} = \frac{6}{h\sqrt{2\pi}} \cdot e^{-18x^2/h^2} \tag{4.2d}$$

The public implementation includes an Epanechnikov kernel, which minimizes the mean squared error with regard to density estimation [Epa69] and can be adjusted to end at exactly the desired width. We truncate the Gaussian kernel as it would otherwise never reach zero: we calculate the parameter for standard deviation with $\sigma = h/6$, as is common practice. Doing so results in an interval of $\pm 3\sigma$ within the bandwidth and an included area of approximately 0.97. The small difference to the required area of 1 is not problematic. Additionally, we include a uniform box kernel and an exact representation of the area covered by circular dots.

When a high dot frequency falls off abruptly, we would still get many dots

with a diameter much smaller than the bandwidth $h = 2d_1$. The envelope shape would force these dots to spread out into the empty area covered by the kernel, leading to large errors in dot placement. To address this issue, we employ a concept from bounded KDE called *reflection* [KA05]. We mirror the frequency values back into the denser area when at the outer limits or in gaps that are at least d_1 wide (see Figure 4.4). Also, we determine the radius of the dot closest to the gap or outer limit and place the reflective boundary at the position of this dot's edge (data value $\pm$ radius).

4.3.2 Nonlinear Frequency Scaling

Kernel frequency estimation (KFE) gives us an approximation of how many data values appear close to a position on the x-axis. While this approach is sufficient for linear relaxed dot plots, the envelope is too large for the smaller dots from nonlinear plots. Previous column-based approaches used the number of dots c in each column to compute the height $h_c(c)$. To scale the height of the envelope, we need to map the previous column height $h_c(c)$ to a new height function $h_f(f(x))$ that depends on the dot frequency f at any position x along the x-axis. With index c, we refer to existing column-based layouts, while index f refers to the new frequency-based functions.

The (nonlinear) sweep layout creates columns that contain as many dots as there are data values in the covered range of the x-axis. We can derive a mean local value frequency $f_c(c)$ through the width (equal to the dot diameter $d_c(c)$) and dot count in each column: $f_c(c) = c/d_c(c)$. The mean frequency $f_c(c)$ should be equal to the estimated frequency $f(x)$ when the distribution of the underlying data is uniform, the bandwidth is sufficiently large, and the integral of the estimation kernel is 1—as required in Section 4.3.1. Therefore, the following equalities must hold:

$$f_c(c) \stackrel{!}{=} f(x) \tag{4.3a}$$

$$h_c(c) \stackrel{!}{=} h_f(f(x)). \tag{4.3b}$$

Using these relations, we can calculate the frequency-based height function of the column-based **root** dot plots by

$$h_{f,r}(f(x)) = d_1 \cdot (f(x) \cdot d_1)^{(1-s)/(1+s)}, \tag{4.4}$$

where d_1 is the user-specified start-diameter of dots, and s is a parameter that characterizes the root. In **logarithmic** plots, the user-selected base b is used for column height, and the diameter is calculated through division

by dot count. Since solving for frequency-based height is not straightforward, we get the inverse equation for frequency from height and solve it numerically (nested intervals):

$$f(x) = \left(b^{(h_{f,l}(f(x))/d_1)} - b + 1\right)^2 / h_{f,l}(f(x)). \tag{4.5}$$

In Chapter 5, we explain in detail how we rearranged the equations to arrive at the results in this section. There, we also show plots of the envelope scaling functions.

When we draw dot plots, there are only integer numbers of dots. This conflicts with continuous, smooth value frequencies. When the estimated frequencies are lower than $1/d_1$, the height functions h_f will not provide enough room to place full-sized dots (see also the special point P in Sections 5.6 and 5.7 on page 75 and the following pages). In practice, when the frequency is so low, there are only a few data values to display, and there is no need for stacking. Therefore, we extend the height functions to handle this special case. The piecewise functions $\hat{h}_f(f(x))$ and $\hat{d}_f(f(x))$ return 0 at frequency $f(x) = 0$ and return d_1 when the frequency $f(x)$ is between 0 and $1/d_1$. For higher frequencies, they return the same value as the original functions $h_f(f(x))$ and $d_f(f(x))$. In Chapter 5, we analyze the new functions with respect to limits for the user-defined parameters.

4.3.3 Individual Dot Diameter

While we could reuse the circle diameters from the sweep layout in the relaxed variant, having circles of the same diameter in spatial proximity could work against our efforts to find ideal dot locations and avoid moiré effects. Also, we cannot know the number of dots c in nonexistent "relaxed columns." However, we know the height of the envelope at any point along the x-axis. Accordingly, we can use generic observations about the height of columns and the local frequency within them to get a generic function

$$d_f(f(x)) = \sqrt{h_f(f(x))/f(x)} \tag{4.6}$$

for the individual frequency-based diameter of each dot. Again, we refer the interested reader to Chapter 5.

4.4 Dot Placement

After having determined plotting space and dot size requirements, we optimize the actual dot positions in a two-stage process: First, we initialize the layout using the two-way sweep from Algorithm 3.2 (page 21) for a good

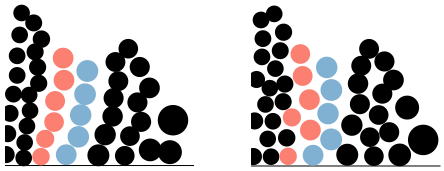


Figure 4.5: Relaxation of dots in ascending order can create leaning columns (left). We avoid this behavior with an alternating vertical order before applying relaxation (right).

Figure 4.6: Narrow bandwidths lead to rough shapes that can split Voronoi cells (left). Higher bandwidths create smooth envelopes and keep the cells intact (right).

approximation. Then, we relax and swap the dot positions iteratively with the goal of more correct placement while achieving blue noise property. Algorithm 4.1 shows the high-level procedure.

4.4.1 Alternating Vertical Order

Column-based nonlinear dot plots sort all underlying data points by their value along the x-axis as a prerequisite of the sweep algorithm. As a consequence, when no second data attribute is shown as color, there is also a predetermined vertical order within each column: small values appear as dots on the bottom and larger ones at the top. The relaxation algorithm will move dots closer to their underlying data value later on (see Section 4.4.3). Similarly to bee swarm plots [ET21], this would create the “leaning towers of dots” shown in Figure 4.5.

As part of the initialization with the column layout, we sort dots vertically in alternating order. First, the dot representing the smallest value, then the largest one, next the second smallest, second largest, etc. To maintain the coherence of colored patches, we only reorder dots of the same color. As Figure 4.5 shows, this approach avoids the leaning columns. We will revisit further implications of this technique when discussing faithfulness in Section 4.8.1.

Algorithm 4.1: Layout

Input: Envelope shape s from $h_{f,r/l}$ in Section 4.3.
List c of columns from two-way sweep with central positions, included dots, and underlying data.
Weight v of the underlying value for x-position of dots ($\in [0, 1]$).
Output: List d of relaxed dot positions, diameters, and data values.

▷ `Initialize from sweep algorithm`
$c' \leftarrow$ alternate vertical order of dots in c
$c'' \leftarrow$ squeeze dots in c' vertically to fit underneath s
$d \leftarrow$ extract dot positions and underlying values from c''

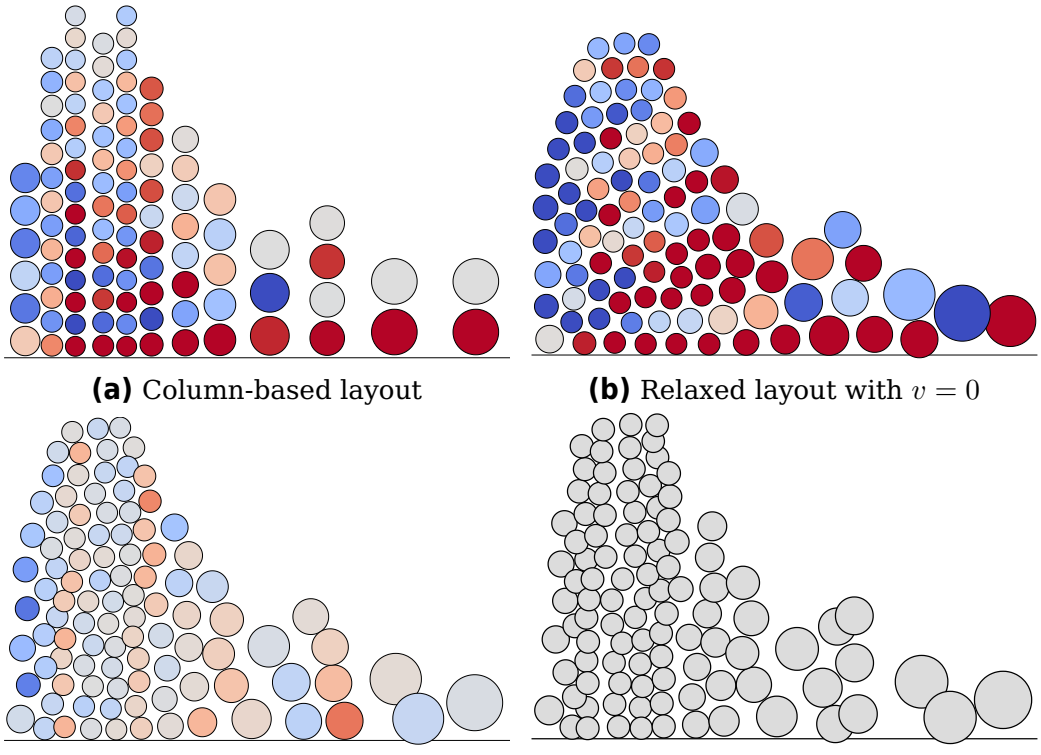
▷ `Dot diameters from estimated frequency`
diameters of $d \leftarrow d_f(f(datavalues(d))$

▷ `Relax dot positions`
repeat
  $d \leftarrow$ `TunnelSwaps(`d`)`
  $d' \leftarrow d$
  ▷ `New positions as in Lloyd's algorithm`
  $position(d) \leftarrow$ Voronoi centroids of d clipped to s
  ▷ `Relaxation toward data value`
  $position_x(d) \leftarrow v \cdot datavalue(d) + (1 - v) \cdot position_x(d)$
until *mean dot movement from* d' *to* $d \leq \epsilon$

4.4.2 Centroidal Voronoi Tessellation (CVT)

Our relaxation is based on Lloyd's algorithm [Llo82] to establish the blue noise property. In each iteration, the algorithm calculates a Voronoi diagram of circles from the given dot positions and diameters. We use signed distance functions and GPU-friendly jump-flooding [Yua+11] in our implementation. However, there are also other algorithms to compute the Voronoi diagram of circles [Aur91]. The result is one cell per dot. To stay within the desired plot bounds, we clip these cells to the envelope before computing the centroids. When moving the dots toward their respective centroids, one obtains a *Centroidal Voronoi Tesselation* (CVT) where the distances between dots are locally equalized, and the blue noise property is observable [SD11; WW11] (see Figure 4.9).

Since we clip the Voronoi cells to the envelope, outer cells might split into disjunct regions, as in Figure 4.6. The centroids of such split cells could escape the envelope, disrupting the visualization of the distribution of data frequency. To avoid split cells, we use a bandwidth of twice the initial dot diameter. The remainder of the algorithm has no issues with roughly shaped cells and spiky envelopes.



(a) Column-based layout

(b) Relaxed layout with $v = 0$

(c) Relaxed layout with $v = 0.3$

(d) Relaxed layout with $v = 1$

Figure 4.7: Analysis of positional error with varying settings for relaxation. The color indicates a dot's horizontal offset from its underlying value: -1 +1. The offset is given in units of the individual dot's radius and clamped to ± 1.

4.4.3 Placement Correction

One of our goals was to create a more faithful visualization in which dots are closer to their underlying value than in previous sweep layouts. Since a dot plot shows 1D data, the error is the difference between the actual display position on the x-axis and the true data value. Think of two dots that are both offset by 5 pixels. One of them has a diameter of 20 pixels, the other 3 pixels. The latter has much more error, as it represents a value that lies outside the area of the x-axis covered by the dot. The error is smaller for the large diameter because it covers the correct data position. The center of a large circle is more difficult to determine, making a slight offset less significant for the user's perception of the visualization. Therefore, we quantize the positional error of a dot as the offset in units of its radius.

A plot with many small errors is more trustworthy than one with a few large ones: If a dot should be on the far right but appears on the far left, it introduces fabricated information, and the user cannot map it to the underlying data. In contrast, small errors are barely perceptible and can fall

within the inaccuracy of reading the dot positions on the x-axis. Therefore, we use the squared error to favor layouts with smaller errors and heavily penalize large offsets. The overall error of a plot is then calculated as the sum of squared errors (SSE). Shifting the dots toward the position of their underlying data reduces the SSE and leads to a more faithful layout (see Figure 4.7). Dividing the SSE by the number of data points allows us to compare errors between plots of different data sets.

Implementation-wise, we move dots along the x-axis through weighted positions from the CVT and the actual data value, as shown in line 9 of Algorithm 4.1. Too much weight on the correct position leads to increased overlap between dots (see Figure 4.7). Too little weight for correction, and dots will not move toward the correct data value. Users of the visualization have to decide on a trade-off between aesthetics and correctness. As Figure 4.7 shows, a default value of $v = 0.3$ provides satisfactory results. However, users can still adjust the weight to fit their needs.

4.4.4 Swaps for Tunneling

One major limitation of any partition-based relaxation is that partitions can block each other, i.e., dots cannot move closer to their ideal position at high dot density. This problem is also known as the conflict of Voronoi diagrams [Qin+17] in multi-class sampling. Inspired by quantum tunneling [Row91], we let dots cross an obstacle by tunneling through it instead of displacing it.

In Algorithm 4.2, we check whether the overall error decreases when two dots swap positions. Note that dots only exchange their positions but retain their diameter and underlying data. This way, we can achieve a layout with less overall error. The dots can be stored in a sorted manner with minimal overhead in line 1 of Algorithm 4.2 because the input data is already sorted at the beginning of the conventional sweep algorithm. This order enables us to restrict the search for swap partners to only a small neighborhood of each dot (line 3).

If the dots are indistinguishable, e.g., by color, the influence of swapping is hardly noticeable. But swaps can change the vertical order of dots, which interferes with the display of a second data dimension, e.g., categorical data. Without vertical coherence of categories, the dot plot would quickly look variegated and cluttered. Therefore, we only swap dots whose colors are sufficiently similar.

Algorithm 4.2: TunnelSwaps

Input: List d of **current** relaxed dot positions, diameters, and data values.

Output: List d' of **swapped** dot positions, diameters, and data values.

1 $d' \leftarrow$ sorted copy of d
2 **foreach** $A \in d'$ **do**
▷ `Abbreviations:` $d_A := diameter(A)$, $val_A := datavalue(A)$, $posX_A := position_x(A)$
3 $p \leftarrow$ dots from d' that (are closer to val_A than A) and (have similar color to A)
4 $B \leftarrow$ dot from p with minimal $E' \leftarrow \left(\frac{val_A - posX_B}{d_A}\right)^2 + \left(\frac{val_B - posX_A}{d_B}\right)^2$
5 $E \leftarrow \left(\frac{val_A - posX_A}{d_A}\right)^2 + \left(\frac{val_B - posX_B}{d_B}\right)^2$
6 **if** $E' < E$ **then**
7 swap $position_{x,y}(A)$ with $position_{x,y}(B)$

4.5 Termination Criterion

The loop in Algorithm 4.1 could run indefinitely, but it would not be computationally efficient. The iterative approach requires a condition for termination in line 10, which we will now investigate.

A main aspect of the quality of visualization is the absence of errors. A plot where dots are close to their intended position along the x-axis is better than one with large errors/offsets. Hence, we analyze the use of mean squared error (MSE) as a criterion for the termination of our algorithm. Measuring the error in pixels would not work well because the vector-based plot layout can be shown at an arbitrary resolution. We measure the error of each dot in units of its radius to keep values comparable between different ranges of the visible x-axis. The metric for the entire plot is then the MSE of all dot positions. Our tests show that the range of possible values of the metric is not sufficiently confined for practical use with data sets of different sizes. For example, in a small data set, the low number of dots leads to relatively large radii and low MSE. A large data set with many small dots has a much higher MSE. Therefore, we need to explore another metric.

We argue that there is no need to continue complex calculations that will not result in visible changes to the plot. To determine when there are no significant changes, we need to quantify dot movement first. To this end, we use the mean dot movement (MDM) in units of radii. Figure 4.8 shows how the MDM changes with each iteration. Overall, the values

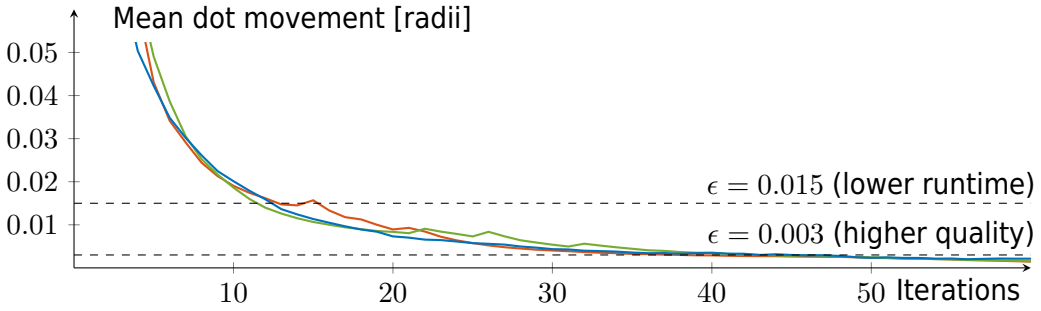


Figure 4.8: Mean dot movement between relaxation iterations with three data sets. Number units are in diameters to get comparable numbers across varying dot sizes. Dashed lines show the suggested dot movement threshold ϵ for algorithm termination.

are very comparable over a variety of data sets, although there are some spikes when dots move through tunneling swaps. Our MDM metric behaves similarly to the quality metric in previous work: there were large changes with the first iterations, but the values stagnated shortly after [DSZ17].

Based on the experimental results shown in Figure 4.8, we choose a threshold of $\epsilon = 0.015$ for MDM. This value is sufficient for a fast approximation of the layout because it includes most movement. A less time-constrained and more highly refined layout is achievable with $\epsilon = 0.003$.

4.6 Approaches Without KFE and Scaled Envelope

During the development of our algorithm, we explored other paths and rejected them in favor of the approach described in the preceding pages. In this section, we discuss some of these “paths not taken.”

We tried to avoid the explicit computation of KFE and the envelope shape. The first issue was that the Voronoi cell of the uppermost dots would extend to the top of the plotting area. Relaxation would then move dots toward the center of mass of each cell and slowly spread all dots over the entire vertical space. The result looked similar to *blue noise plots* without centralization [Onz+21] and was not in accordance with our initial goals from Section 4.2.

We adapted the previous attempt by adding virtual boundary dots on top of the columns from the initial sweep layout. They were supposed to restrict the vertical dot movement. However, they did not succeed in containing the actual data dots. When a column protruded far above its neighbors, it would open a gap and let dots escape toward the upper region. We also tried filling the entirety of the remaining space with virtual boundary dots. The result, again, was not satisfactory because the boundary dots enforced

the same rough and spiked shape of the original column-based layout that we were trying to avoid.

We also experimented with pure Lloyd relaxation without correction toward the underlying x-value. As could be expected, this led to a uniform distribution of dots in the entire available plotting space. This was an issue with both the rectangular plotting window and the envelope shape. Such a layout would not serve the purpose of dot plots because the positions were arbitrary and not suited for meaningful visual analysis.

4.7 Examples

In this section, we showcase the relaxation technique using two real-world data sets that touch upon the topic of global warming in a broader sense and have a high dynamic range, i.e., nonlinearity is worthwhile. Figure 4.9 shows a derivation of the same global air temperature measurements that we already used for the plots in Figure 3.4 on page 21 [Deu]. The original data was rounded to one decimal place, which led to various regular patterns in non-relaxed plots. Since we want to compare the visualizations themselves without stray effects from the granularity of the data, we jittered the temperatures in the current example to get data points from a continuous spectrum.

The column-based nonlinear dot plot in Figure 4.9 cannot represent the data frequency well. The sharp spikes between the temperature of 20 and 30° C are not present in the data. Notably, there are no spikes for similar frequency fluctuation between 0 and 10° C, which is inconsistent. The intermittent gaps from -20 to -5° C show that the quantized widths of columns are not a good fit for the data frequency. The relaxed dot plot shows a consistent and appropriately smooth contour. Moreover, it shows a lower mean squared positional error of the dots. It is most noticeable at the edges of the x-axis that points with different values are not placed at the same x-coordinate. The plot not only improves correctness but also appears more aesthetically pleasing as dots are more evenly distributed, and the blue noise property avoids strong moiré effects.

We also provide visualizations of the data in two plots from related work. Note that both related implementations are not designed to separate differently colored dots within a single layout, making it difficult to extract information from the dot colors.

Blue noise plots by Onzenoodt et al. [Onz+21] have the option to centralize dots and limit their positions to an area that represents the underlying value frequency (similar to violin plots). As Figure 4.9 shows, the visualization we created with the publicly available implementation places

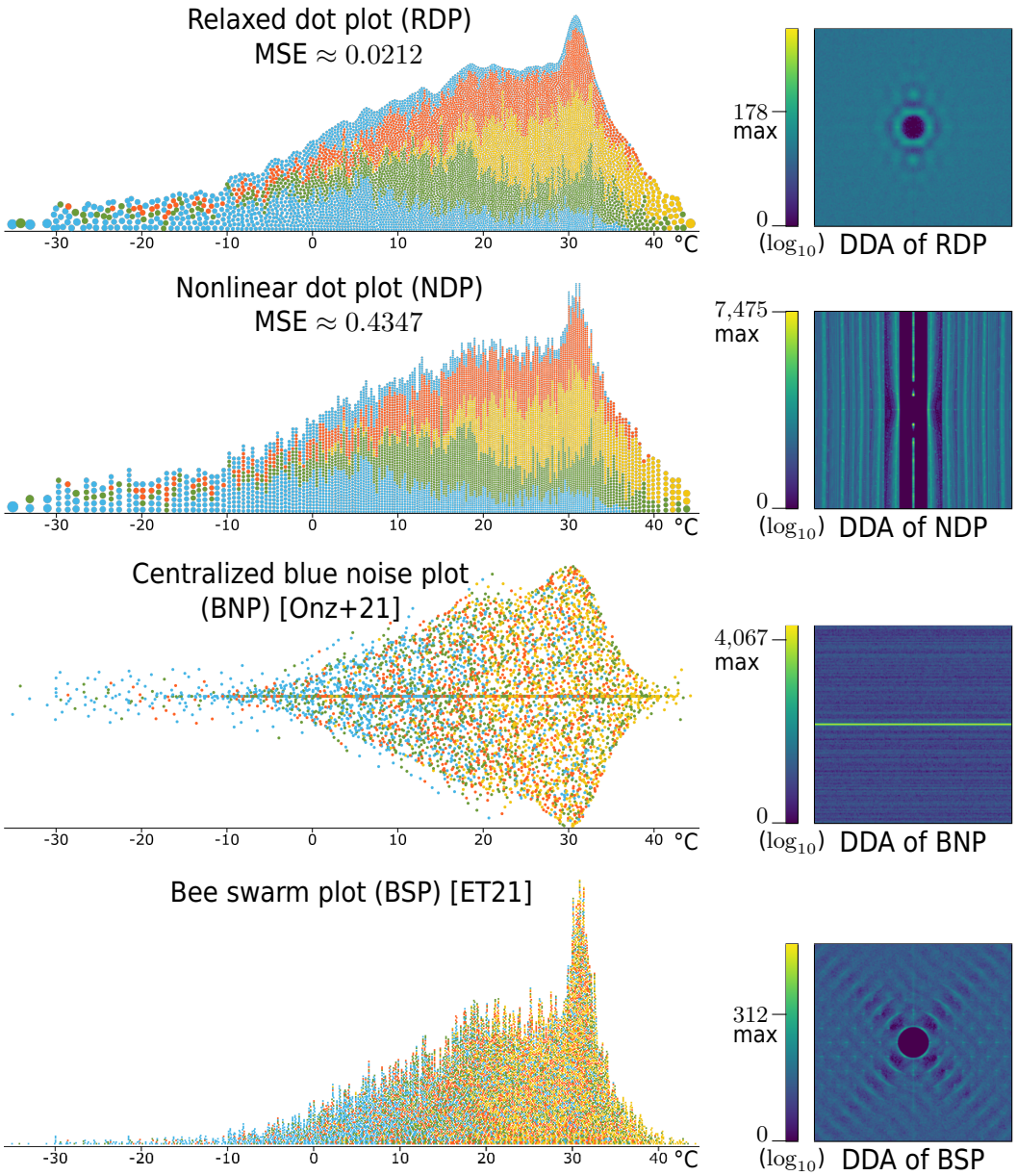


Figure 4.9: Different dot-based visualizations of temperature data. Dot color is mapped to month: J F M A M J J A S O N D. In comparison to the column-based plot (NDP), the relaxed variant (RDP) has far fewer moiré artifacts, dots are closer to their data value (lower MSE), and outliers at the edges of the x-axis do not require stacking. The $\log_{10}$ scaled results of DDA in the right column are for comparison of blue noise quality and show the maximum energy of each plot as a number next to the scale. The visualized DDA only includes distances up to 2.5 % of the image width to better visualize the blue noise property. Ideal blue noise has an eclipse-like pattern in the center (i.e., minimal low-frequency components) and low peak values (i.e., absence of spikes in energy). Regular patterns should be minimal if they are not founded in the underlying data.

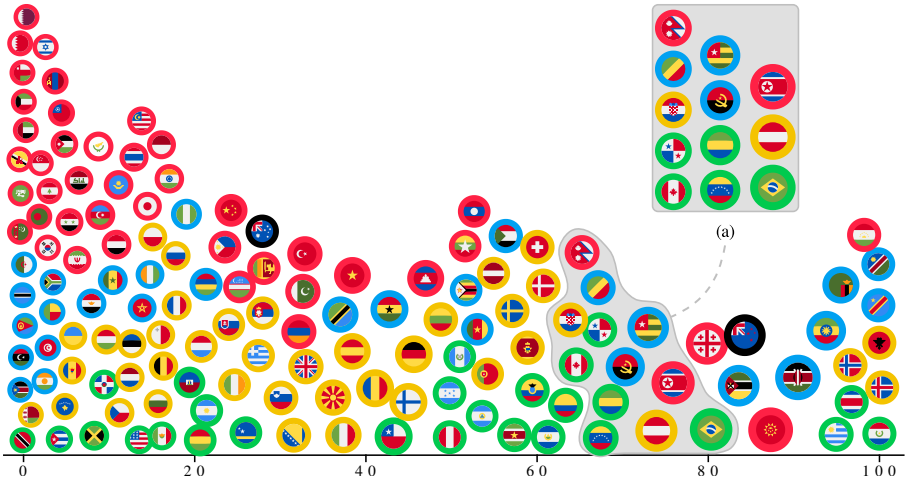


Figure 4.10: Relaxed dot plot of the percentage of renewables in electricity production. Dots represent countries (flags designed by Freepik from Flaticon.com). Outlines show the corresponding continent: Africa, America, Asia, Europe, and Oceania. Data retrieved from [Wik21]. Relaxation represents subtle differences in values more faithfully than column-based dot plots (a).

many overdrawn dots on a horizontal line. We did not expect the line-like structures at the outer bounds of the shape that is supposed to indicate value frequency. We do not know whether these two artifacts are caused by conceptual faults in the algorithm or errors in the public implementation. Plots of smaller data sets are consistent with the expected outcome of the technique and the figures in the original publication. The obtained layout was calculated in over 7 minutes with the regular CPU implementation as the graphics hardware was running out of memory (Intel Core i5-6600 3.3 GHz, NVIDIA GeForce 1060 GTX 3 GB). All other layouts were computed in less than 10 seconds.

Eklund and Trimble's bee swarm plot [ET21] has no overdraw but moiré effects. Figure 4.9 shows how the visualization appears inside the plotting pane of RStudio when using the current public R implementation of the algorithm. Dots that are close to the x-axis form straight stacks before branching out. In the supplemental material of the original publication [Rod+22b], we provide monochrome plots in which the spaces between dots merge to form worm-like shapes in mostly diagonal directions. The color noise in Figure 4.9 helps to mask such effects.

Both visualizations from related work do not adapt dot sizes, providing only linear plots. The small dot sizes required to visualize the entire temperature data set are problematic when trying to analyze outliers. This is where nonlinear dot plots are most suited.

(a) Column-based dot plot **(b)** Relaxed dot plot

Figure 4.11: Linear dot plots of two narrow normal distributions. The distribution on the left has a larger variance. The relaxation-based algorithm propagates the variance faithfully into the layout.

Figure 4.10 shows the percentage of renewables in electricity production of various countries. Here, we show each country's flag and encode the continent in a colored outer ring to emphasize the possibilities of unit visualization. Most countries are between 0 and 10 % renewable energy. There are two clusters, around 60 and 100 %, but only a few countries in between. Through color, one can estimate each continent's distribution in the context of the overall distribution.

This example is also well suited for *broad audiences*: While only a few individuals know all countries and their banner, most people know their own and neighboring countries' flag. Therefore, such visualization in a public place could be used for nudging and increasing awareness of the need for renewable energy in times of global warming. Passers-by might try to find their country's flag, read off the data value, and compare it to other known countries. While it has been shown that unit visualizations with column-based dot plots are simple enough for children to grasp and interpret quickly [I G15], we argue that relaxed dot plots lower cognitive barriers: There is no need to understand the binning into columns and the values are represented with less hidden uncertainty. Viewers are only required to grasp how a number line works to extract basic and more correct information than with column-based layouts.

4.8 Evaluation

In this section, we report on our evaluation of relaxed dot plots. We start by comparing metrics from computational experiments and continue with the results of a crowdsourced user study.

4.8.1 Correctness

In comparison to column-oriented layouts, the weighted placement correction in the relaxation algorithm strongly decreases horizontal error. This can be seen in Figure 4.9 and in the synthetic case of Figure 4.7. The relaxed plot in Figure 4.9 has over 95 % less error in dot positions.

The envelope shape uses KFE to determine the underlying data distribution and is completely filled in the relaxation step. Therefore, the overall shape of the relaxed layout shows a better approximation of the frequency distribution than the column-based dot plots. Additionally, the new method also provides a more faithful display of local variance. Previous techniques for dot plots align dots in perfectly straight stacks, depriving the user of a notion of variance within each column. If such a vertical column of dots appears in a relaxed dot plot, it strongly indicates a low variance of the underlying data values. Figure 4.11 illustrates such a case in which the layout's alternating order spreads the dots in the column with higher variance. This is especially useful when analyzing uncertain and noisy data: a vertical line pattern will call for attention toward an anomaly.

4.8.2 Crowdsourced User Study

We conducted a crowdsourced user study to check whether users can actually benefit from the reduced positional error, the depiction of local variance (or lack thereof in columns), and the smooth approximation of global data distribution (KFE). Thus, our hypotheses are centered around the comparison to nonlinear dot plots with straight stacks. As mentioned in Section 4.9, column-based layouts exhibit auxiliary lines that might help viewers to read the display position of a dot. These lines are missing in the relaxed dot layout, and therefore, we hypothesize that the accuracy for reading the dot positions in the relaxed layout is lower (**H1**). As the new layout indicates local variance, we hypothesize that comparisons between close underlying data values are more accurate (**H2**). The original column-based layout is ragged and contains high and low peaks close to each other. This might cause issues with the perception of the density in the underlying data distribution. When focusing on local minima of value frequency, it is our hypothesis that the apparent depth in the relaxed layout is closer to the KFE of the underlying data distribution than with the column layout (**H3**). Each hypothesis relates to a specific aspect of both dot plots and is tested through one experiment. All these experiments are designed as two-alternative forced-choice (2AFC) [SP09] to avoid bias and provide easy input methods for participants. The 2AFC design is sufficient to provide evidence for our hypotheses without going into overly complex

studies to extract just noticeable differences. Additionally, there are no missing answers with "no choice" because participants have to decide between the two alternatives. The study has a within-subject design.

Procedure We started with a short introduction to explain how dot plots are read. At the beginning of each task, we provided additional explanations and a test question with immediate feedback (more details in supplemental material [Rod+22b]). Further training does not seem to have been necessary, confirming the low cognitive threshold to access information in dot plots [I G15]. In the experiment for **H1**, we showed a plot and highlighted one dot, as in Figure 4.12. Participants were asked to read off the displayed dot position on the x-axis. They were presented with a correct choice that represented the exact display position of a dot and an incorrect value that was offset by the independent variable δ_1 (Figure 4.13). This task was designed to compare the influence of vertically aligned points (column-based) and evenly distributed points (relaxed) on value-reading performance. Participants were asked not to use additional tools. In addition to the reading strategy, the resolution of tick marks could also affect reading accuracy. Hence, the wrong choice was offset in units of distance between tick marks on the x-axis. The range of $\delta_1 \in \{\pm 0.005, \pm 0.25, \pm 0.5\}$ ensured easier and very difficult tasks (determined in pilot studies).

For **H2**, we highlighted two dots and asked which one represented the higher value (see Figure 4.12). The distance δ_2 between the underlying values of these dots was the independent variable (see Figure 4.13). With this setup, we examined which layout was better suited to decide which of the two data values was higher. Differences in lateral position between small dots are easier to notice than with large ones (same reasoning as with error metric in Section 4.4.3). In consequence, the values of the two dots that represented the choices in this experiment were spaced apart in units of displayed radii with $\delta_2 \in \{0, \pm 2/3, \pm 4/3\}$.

Regarding **H3**, we showed a plot in the center and two smooth silhouette shapes on the sides (see Figure 4.12). Participants were tasked with selecting the shape that best matched the plot. The experiment was designed to compare the perception of minima in both plots' silhouettes. We used the envelope of the underlying data distribution and generated Hermite splines to connect the two local maxima that surrounded a local minimum. Figure 4.13 shows how the independent variable δ_3 controls attraction between the Hermite spline and the envelope when positive, *filling the valley* (■), or repulsion when negative, creating a "deeper valley" (■). The range of $\delta_3 \in \{\pm 0.2, \pm 0.4, \pm 0.6\}$ was set in units of distance between the envelope and spline.

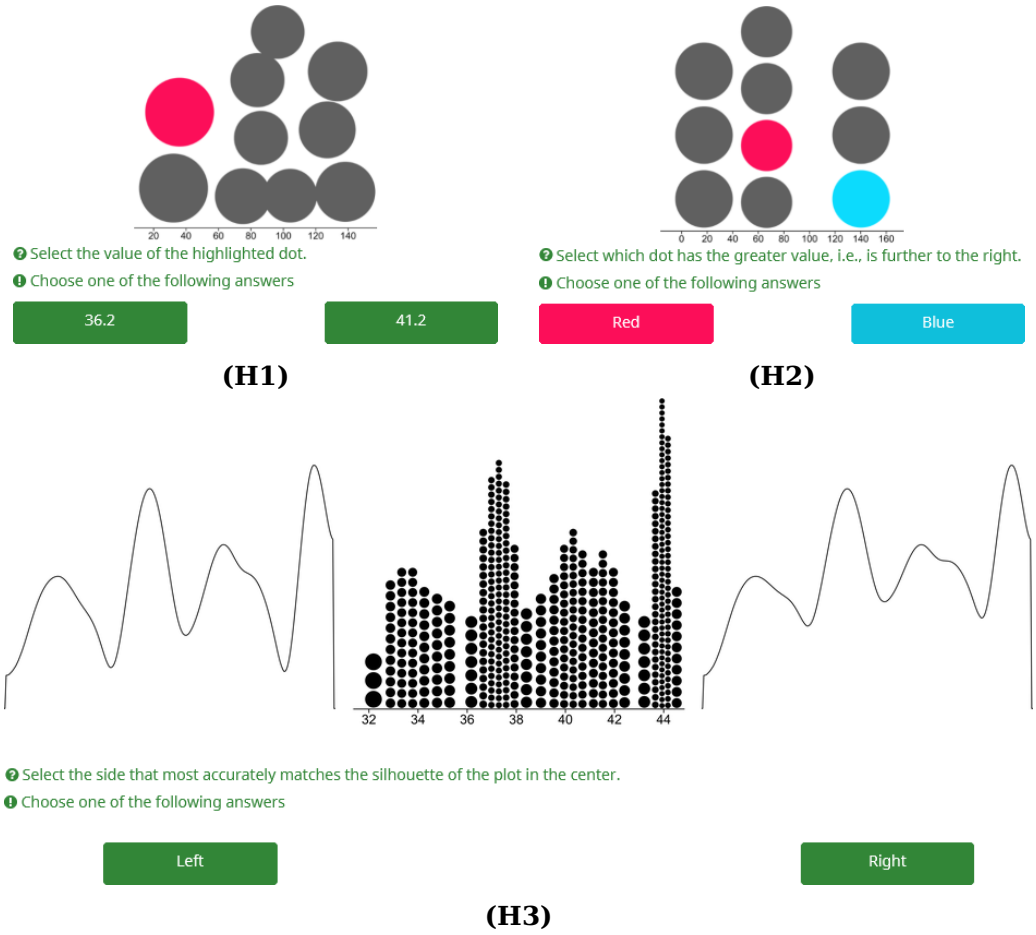


Figure 4.12: Examples of study tasks for each hypothesis. We generated the stimuli randomly by sampling from the same underlying data set.

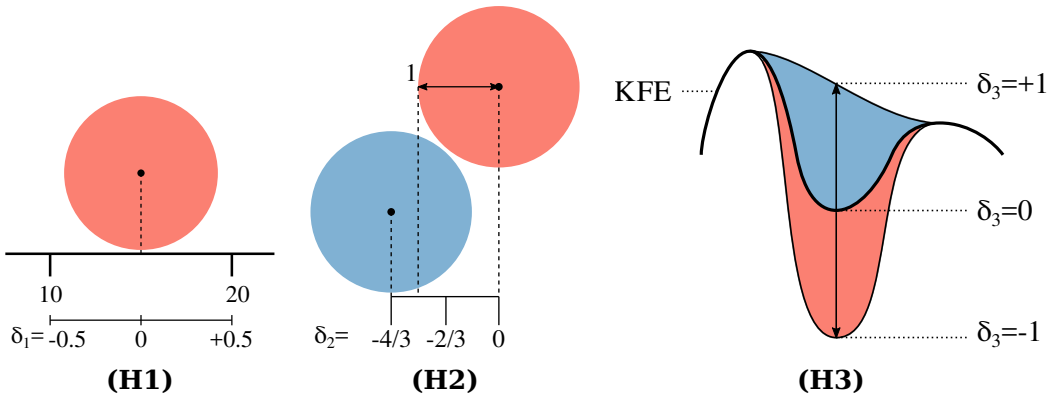


Figure 4.13: The independent variables δ of each experiment.

For crowdsourcing, we created 360 2AFC questions in batches (3 tasks $\times$ 3 data set sizes $\times$ 5 values for δ $\times$ 2 plot types $\times$ 4 batches). Additionally, we asked test questions with stimuli adjusted to have very obvious answers. Participants with an accuracy rate below 80 % on these questions were removed from the analysis.

Analysis In total, we collected 90 responses from each of the 100 participants. Figure 4.14 shows the distribution of correct responses for each task and δ. In the value-reading task, we checked the distribution of heights of selected dots because it could be a confounding variable. The distributions did not differ significantly between both plot types (two-sample Kolmogorov-Smirnov test, $D = 0.2$, $p = 0.182$). The selected data values for the task were drawn randomly and are the same in both variants. This means that possible influences of horizontal anchoring effects are rooted in the layout algorithm and are actually part of the dependent variable. The distributions of correct answers do not differ considerably. The percentage of correct answers for columnar dot plots ($Mdn = 0.78$) is not significantly different from the relaxed ones ($Mdn = 0.75$), which we confirm with a Mann-Whitney test, $U(N_{column} = 100, N_{relaxed} = 100) = 5415.5$, $p = 0.309$. Therefore, we cannot confirm our hypothesis **H1** and assume that users can read the dot positions with the same accuracy in both visualization types. Note, however, that relaxed plots place the dots more correctly in the layout, and consequently, the user-read values should be closer to the underlying data.

For the comparison task, there is a large difference in variance and median ($Mdn_{column} = 0.60$, $Mdn_{relaxed} = 0.86$) of correctness. When there is no difference in data value, the column-based dot plots produce less variance—without being more accurate. In all other cases, responses based on relaxed dot plots were consistently much more correct. We can confirm **H2** with a Mann-Whitney test, $U = 771.0$, $p \ll 0.001$.

Results for the perceived shape of the plot are consistently correct when estimating a higher limit—there is no confusion when we *fill in the valleys*. In contrast, participants are very erratic when choosing a lower limit for minima in dot distributions of column-based plots (overall $Mdn = 0.72$). The relaxed variant shows the drop in frequency more clearly, allowing participants to distinguish between the real depth and the artificial lower offset (overall $Mdn = 0.92$). We can confirm **H3** with a Mann-Whitney test, $U = 1876.5$, $p \ll 0.001$.

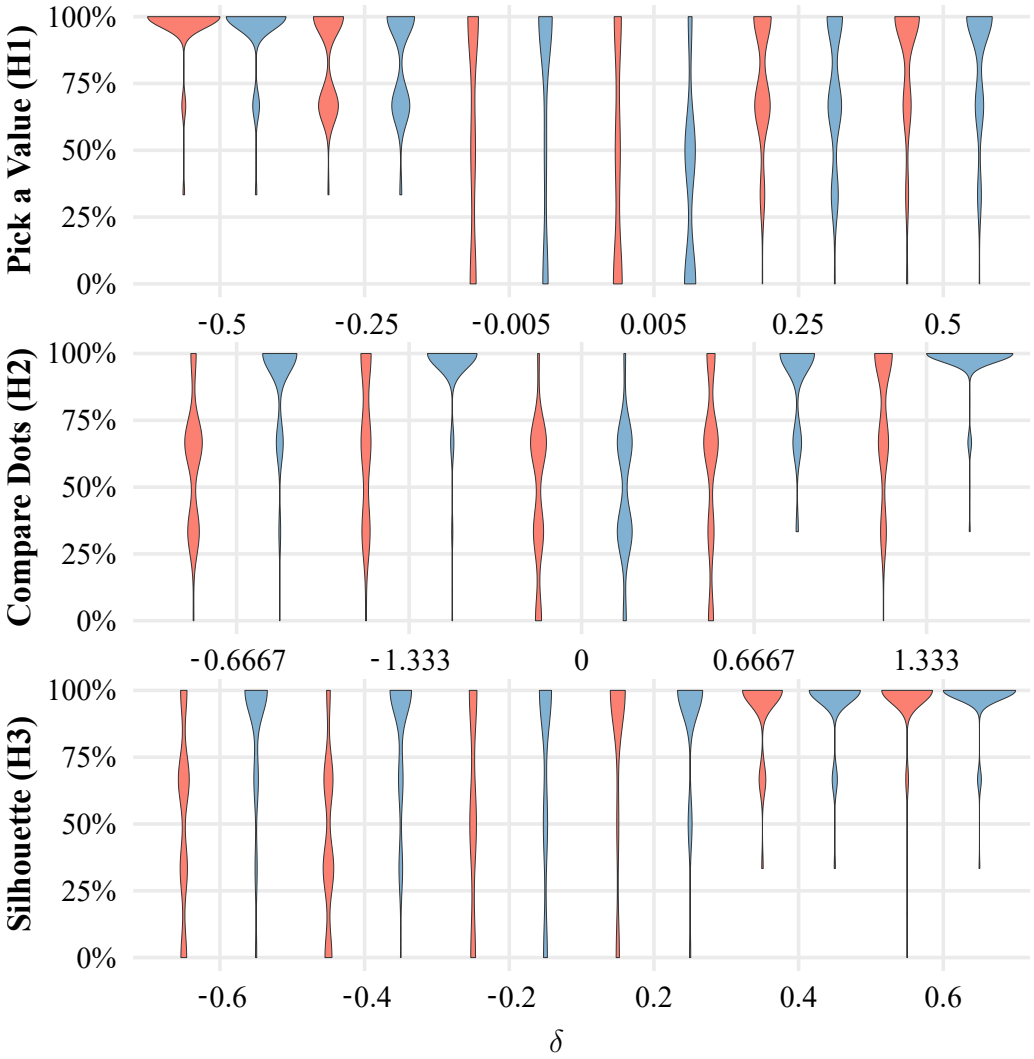


Figure 4.14: Correct responses (%), grouped by participant, task, independent variable (δ), and plot (column-based ■, relaxed ■). Each participant contributed one data point to the density displayed in the violin plots.

4.8.3 Blue Noise Property and Moiré Effect

An important advantage of relaxed dot plots is that they observe the blue noise property. In the context of dot plots, blue noise requires that the outer edges of the dots should be locally equidistant and, ideally, isotropic. Generally speaking, blue noise only has little low frequencies, and the remainder of the spectrum has no intense spikes in energy. This is a desirable property for applications in the context of sampling, and there have been many methods to implement it [Yan+15]. It is community standard in computer graphics to show this quality using Fourier analysis (FA) [SD11] and differential domain analysis (DDA) [WW11].

Dot plots are inherently non-uniform distributions (except for artificial corner cases that are evenly dotted with 1:1 aspect ratio). Since FA is not applicable to non-uniform point distributions without introducing strong assumptions (e.g., through window functions), we use DDA to assess the quality of dot plots instead. The main mathematical difference between those methods is that FA uses a cosine kernel, whereas DDA uses a Gaussian kernel. For the interested reader, we refer to the original publication [WW11] for a detailed derivation of DDA from FA. Accordingly, DDA does not show direction-dependent energy of frequencies but the direction-dependent distribution of distances. Note that for our analysis, we impose a lower limit of 0 for the signed distance functions between dot surfaces before applying DDA. This is necessary to handle the case of overlapping dots without introducing negative values.

The DDA plots in Figure 4.9 quantitatively show the influence of relaxation regarding blue noise. The static column layout induces regular patterns, whereas the relaxed dot plot shows the typical eclipse-like pattern of blue noise. The peak energy drops from 7475 to 178, i.e., only 2.4 % peak energy remains. As expected, the relaxed dot plot does not establish isotropic blue noise, as indicated by minor patterns around the eclipse. However, there are almost no low-frequency components in the center, and the remaining energy is noise-like distributed.

Blue noise plots (BNPs) by Onzenoodt et al. [Onz+21] placed approximately half the dots on a horizontal center line. This leads to a strong line in the DDA and many weaker horizontal stripes. In the supplemental material [Rod+22b], we provide a DDA of the same BNP as in Figure 4.9 but exclude the dots on the center line. It shows an oval eclipse and noisy distribution of energy around it. As the bee swarm plot (BSP) creates a tight packing of same-sized circles, their DDA has mostly diagonal patterns. The sharp edges of the center eclipse are due to the layout, where dots are all of the same size and can touch but will not overlap. In contrast, the eclipse in the DDA of the relaxed layout is blurred because the dots are of different sizes.

In the context of dot plots, a moiré pattern can emerge when the sampling scheme of the viewer (rasterization) becomes *too similar* to the dot pattern. For column-based layouts, this effect usually occurs when reducing the plot size (e.g., using bilinear interpolation for downsampling) or increasing the viewing distance (which has a similar effect). In such a scenario, the columns of dots align with the columns of display pixels. As every column might have a slightly different width, it aligns more or less perfectly with the pixels and can create virtual tilted lines. Because relaxed dot plots show sufficiently good blue noise, there are no intense frequencies to “catch on,” making the occurrence of moiré effects less likely.

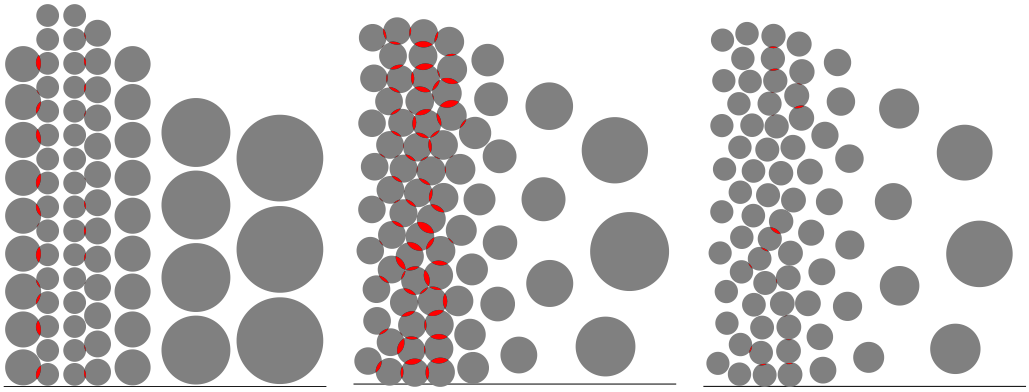


Figure 4.15: Dot plots with overlap marked in red. The columnar plot (left) draws dots 5 % smaller than the diameter during the layout to create separation. Relaxed variants with the same size reduction would have too much overdraw (center). Therefore, we shrink dots by 20 % to avoid most overlap (right).

4.9 Discussion

Overdraw and Dot Distinguishability When two dots overlap, the circle area becomes ambiguous and difficult to recognize (see Figure 4.15). Dots rarely overlap in column-oriented plots—usually only when the underlying value frequency changes rapidly. Lloyd relaxation with signed distance functions also tends to keep the dots apart, but the horizontal placement correction is not aware of circle boundaries and increases overlap. The previous nonlinear plot technique from Chapter 3 used circle padding. It reduced the display size of dots (typically by 5 %) to gain vertical separation and neutralize minute horizontal overlay. In comparison, the relaxed layout with $v = 0.3$ (Figure 4.15, center) has more overdraw. With the same padding, densely packed dots fuse and become difficult to distinguish. As a compromise between faithful data presentation and readability, we apply a padding of 20 % to decrease the mean dot overlap in display diameters (MOD) to a negligible amount.

The padding introduces inconsistency between user-selected dot diameters and the actually rendered plot. This can be a problem in specific use cases. We argue that, in practice, it is easier and probably more frequent for the user or software container to provide the desired size and aspect ratio of the entire plot instead. Automatic adjustment of d_1 and the default value for the padding will then produce plots that just “look right” without a user-selected d_1. An alternative would be to leave d_1 unchanged for display purposes and switch to a model with a margin to increase the dot sizes for internal computations.

Computational Complexity and Performance For this discussion, we assume that the size of the data set is $n \ll 100,000$ since the number of distinguishable dots is limited.

The practical runtime of sweep algorithms for dot plots is small. They run in linear time after sorting the input data, usually with a sorting algorithm that has an average computational complexity of $\mathcal{O}(n \log n)$. In the relaxed variant, the complexity for the envelope (see Section 4.3) depends on both n and the resolution r for sampling the KFE. However, it is linear in both variables.

The runtime of GPU-based jump-flooding for the calculation of the Voronoi diagram mainly depends on the texture size, which has a lower bound defined by the ratio between the largest and smallest diameter of the dots in the plot. Therefore, it scales with the square of the value frequency range, not the size of the data set. The complexity of determining the centroids—used for moving the dots during the relaxation step—also depends on the resolution of the texture. A CPU-based implementation would have $\mathcal{O}(n \log n)$ complexity to compute the Voronoi diagram, followed by linear complexity to compute the centroids.

The tunneling swaps algorithm makes use of the almost-sorted dot positions. First, it obtains a correct order in linear time, e.g., using the insertion sort algorithm. Then, it iterates the sorted list of dots and limits the search to a neighborhood because only dots that are closer to the ideal position are possible candidates for a swap. The result is a linear complexity $\mathcal{O}(n)$ with respect to the size of the underlying data set with a frequency-dependent factor for neighborhood search.

Our open-source implementation is a web-based *D3.js* [BOH11] plugin for maximum portability. This choice introduces considerable overhead through GPU-CPU communication in each iteration (WebGL `readPixels()`), which will be entirely avoidable as more compute-oriented technologies become available (WebGPU). Plots with 1,000 points achieve interactive responses ($\leq$1 s). Larger examples with high-dynamic-range require more time due to the above-mentioned technical constraints ($\approx$5 s for Figure 4.9).

Readability The relaxed layout works with both linear and nonlinear scaling. Column-based linear dot plots provide a regular grid that gives viewers implicit auxiliary lines to track the vertical and horizontal position of dots. The introduction of nonlinear scaling to traditional dot plots enables the visualization of larger and high-dynamic range data. At the same time, it sacrifices the ease with which the vertical position of dots can be read. Additionally, the small and regularly arranged graphics primitives lead to unwanted moiré effects. The advantages of having guides for the horizontal

position remain.

Are implicit vertical auxiliary lines in column-based dot plots actually a benefit? The ease with which users can read the value of a dot instills unjustified confidence because the display position itself is flawed: all dots in a column share the same x coordinate. Our relaxed layout does not align dots; neither vertically nor horizontally. This leads to more correct positioning and avoids unwanted moiré effects.

Scalability of Dot Plots Relaxed dot plots are suitable for large and small data sets with high-dynamic-range frequencies while avoiding unwanted perceptual moiré effects. We developed a formalism that allows us to predict the height of dot plots from data density. That is nothing special for linear layouts but is not trivial for nonlinear dot plots, as mentioned in Chapter 3. The final height has a nonlinear relation because we want a dense dot layout, and changing the sizes will affect not only the height but also the width of circular dots.

The examples from the data sets with temperature measurements (Figure 4.9) and energy percentages (Figure 4.10) show that we can visualize much larger data sets in a manner that serves for statistical visualization but can also be understandable and engaging for a broader audience. Hence, in the context of Richer et al.'s conceptual model for scalability, the new variants of dot plots allow for an increased problem size S. The nonlinear scaling might raise the necessary effort E for specific tasks, e.g., estimating integrals or reading exact value frequencies, but also lowers the effort when analyzing outliers. The smooth shape of relaxed dot plots reduces the viewer's effort in estimating value frequency because it avoids sharp spikes and ragged edges. In addition, the relaxed variant reduces the assumptions A in the scaling model with two approaches. First, users are no longer required to know that sharp spikes in dot plots of large data can be misleading due to overly narrow bandwidths. Second, viewers do not need to be informed about the misleading positions of dots within straight columns and can compare the values of individual dots more accurately (also affecting effort E).

We achieved the presented improvements with higher algorithmic complexity (modeled by Richer et al. as resources R). The computational requirements of relaxed dot plots would not have allowed for interactive real-time visualization on commodity hardware when Wilkinson introduced the original linear dot plots in 1999. At the time of writing, however, the iterative nature of our presented algorithm even presents opportunities for further application of dot plots for dynamically changing real-time data. When the underlying values change, the traditional sweep algorithms could

assign completely new positions to every data point, mainly in the vertical direction. For the viewer, tracking individual data points over discrete jumps would be difficult. The relaxation algorithm only uses the sweep for initialization and then moves dots more smoothly—even when adding or deleting data points. Tunneling, on the other hand, also creates discrete jumps. However, the swaps are an optional part of the algorithm that leads to lower runtime and higher quality. In any case, tunneling affects only a few pairs of dots, and it would be possible to overlay a dedicated animation for the swaps.

In conclusion, we accept relaxed dot plots as an answer to the question of scalability from  RQ1. They combine simple visualization principles and extend their applicability to larger data sets. The lack of strict columns might lower cognitive barriers, making our relaxed variant of dot plots better suitable for visualization for the masses. In addition to the goals from the research questions, the proposed visualization is more *faithful to the underlying data* than previous variants: the dot placement has less horizontal error; the lack of columns better shows local variance and allows interpretation of perfectly straight columns; the envelope shape represents the data distribution more accurately.

Future research might explore other scaling functions with no connection to the previous nonlinear dot plots. Instead of relying on circles, relaxed dot plots might be adapted to arbitrarily-shaped glyphs to make more efficient use of the plotting regions. Our public implementation could be ported to benefit from future technological developments to eliminate latency from GPU-CPU communication.

5

Excursion: Mathematical Transition from Columns to Frequencies

This chapter is based on previous work and will use extracts thereof without explicit quotation:
N. Rodrigues, C. Schulz, S. Döring, D. Baumgartner, T. Krake, and D. Weiskopf. *Supplemental material for relaxed dot plots*. 2022. DOI: 10.18419/darus-3055.

In Chapter 4, we moved away from the column-based layout of nonlinear dot plots and redistributed the dots. In this excursion, we provide the required groundwork in the form of mathematical details. We transform column height and circle diameter from dot-count-based nonlinear dot plots (Chapter 3) to frequency-based scaling for relaxed dot plots. The derived equations allow us to calculate envelope shapes for the plots and to replace the traditional role of columns with a more generic value frequency.

5.1 Recap, Definitions, and Observations

Table 5.1: Conventions for function and variable names in this chapter.

Name	Meaning
$\square_c$	Belonging to dot-count-based column layout
$\square_f$	Belonging to a continuous frequency-based layout
$\square_r$	Belonging to root-scaled dot plots
$\square_l$	Belonging to logarithmically scaled dot plots
$\hat{\square}$	Piecewise function definition for fringe cases

We start with information on the used notations. The additions to variable and function names presented in Table 5.1 provide information about their use in the plotting techniques. We continue by taking inventory of existing variables, recording observations, and noting definitions to get a self-contained excursion with the most relevant information from Chapters 3 and 4.

1. We use the variable c for the dot count within a traditional column.

2. We call a variable *user-defined* when it comes from any external source, which might be a user, algorithm, etc. A *user-defined* variable is not changed within the layout algorithm for dot plots and, therefore, serves as an externally provided constant within a layout pass.

3. User-defined variable d_1 corresponds to the diameter of a dot in a column that only contains one dot.

4. Function $d_c(c)$ represents the diameter of dots within a column (with c dots) in nonlinear dot plots. The two-way-sweep algorithm adds a dot to a column and updates the diameter before checking whether it needs to add another dot.

5. Function $h_c(c)$ represents the height of a column that contains c dots.

6. Variable x is a position in the underlying data dimension.

7. In our efforts to generalize the envelope shape, we move away from the notion of a dot column and toward a value frequency $f(x)$. It is defined as the number of data values within an interval of the data dimension centered at x. Let us take a hypothetical data set as an example. If it had five data values in the interval $[95, 105]$, then it would result in a mean value frequency of $f(100) = 5/(105 - 95) = 0.5$. We obtain a value for $f(x)$ through kernel frequency estimation on the underlying data to be plotted.

8. Function $d_f(f(x))$ represents the diameter of dots in a region with the value frequency $f(x)$.

9. Function $h_f(f(x))$ represents the height of an envelope shape with just enough space to accommodate columns of dots in a region with mean value frequency $f(x)$.

5.2 Goals

Our goal is to move from determining a column-based height h_c to a frequency-based height h_f that will give us the shape of the envelope when we provide it with the value frequency. We can determine this value frequency by applying kernel frequency estimation. In sweep layouts, the function h_c was only required to work for columns with at least one dot ($\mathbb{N}_{>0}$). However, the frequency-based approach has to work for all values in $\mathbb{R}_{>0}$. This will be of special importance for logarithmic dot plots. Since we do not draw discrete columns anymore, we also do not know the number of dots within a single column. Therefore, we also need to move from d_c to d_f, meaning that we can calculate the diameter directly from the value frequency.

5.3 Generic Observations

Traditional dot plots stack circular dots into columns. All dots within such a column have the same diameter. Therefore, we can calculate the height of the columns:

$$h_c(c) = c \cdot d_c(c). \tag{5.1}$$

This equation can be solved for c

$$c = \frac{h_c(c)}{d_c(c)} \tag{5.2}$$

or for $d_c(c)$:

$$d_c(c) = \frac{h_c(c)}{c}. \tag{5.3}$$

As all dots within a column have the same width, we can determine the mean value frequency for a traditional dot column that is centered on x:

$$f(x) = \frac{c}{d_c(c)}. \tag{5.4}$$

Alternatively, we can insert Equation (5.3) to avoid the explicit use of $d_c(c)$:

$$f(x) = \frac{c}{\frac{h_c(c)}{c}} = \frac{c^2}{h_c(c)}. \tag{5.5}$$

If the value frequency were constant along the x-axis and we used a wide kernel for frequency estimation, then both the estimated value frequency and the average frequency of dots in columns would be the same:

$$f_c(c) \overset{!}{=} f(x) \tag{5.6}$$

The goal is to find the height of the envelope shape at position x without the need for explicit dot columns. Both methods of calculating the height should provide the same result when a column with c dots is centered on x and the frequency of underlying values in the interval covered by the column is constant:

$$h_c(c) \stackrel{!}{=} h_f(f(x)). \tag{5.7}$$

If the height is the same and the number of dots is the same, then the diameter has to be equal, too:

$$d_c(c) \stackrel{!}{=} d_f(f(x)). \tag{5.8}$$

5.4 Linear Scaling

In column-based linear dot plots, all dots have a constant diameter, i.e., $d_c(c) = d_1$. Therefore, we can substitute $d_c(c)$ in Equation (5.2) and Equation (5.4):

$$c = \frac{h_c(c)}{d_1}, \tag{5.9}$$

$$f(x) = \frac{c}{d_1}. \tag{5.10}$$

By inserting Equation (5.9) into (5.10), we get

$$f(x) = \frac{\frac{h_c(c)}{d_1}}{d_1} = \frac{h_c(c)}{d_1^2}. \tag{5.11}$$

Now we can solve for $h_c(c)$, which needs to be equal to $h_f(f(x))$:

$$h_f(f(x)) = f(x) \cdot d_1^2. \tag{5.12}$$

5.5 Root Scaling

In column-based root scaling, a user-defined shrink factor $0 \leq s < 1$ (characterizing the degree of the root) is used to calculate the dot diameters:

$$d_{c,r}(c) = d_1 \cdot g_{c,r}(c) = d_1 \cdot c^{-s}. \tag{5.13}$$

We insert this Equation (5.13) into (5.1) to get

$$h_{c,r}(c) = c \cdot d_1 \cdot c^{-s} = d_1 \cdot c^{(1-s)} \tag{5.14}$$

and solve for c:

$$c = \left(\frac{h_{c,r}(c)}{d_1}\right)^{1/(1-s)}. \tag{5.15}$$

Now, we insert Equation (5.13) into (5.4) to obtain

$$f(x) = \frac{c}{d_{c,r}(c)} = \frac{c}{d_1 \cdot c^{-s}} = \frac{c^{1+s}}{d_1}. \tag{5.16}$$

Next, we solve for c:

$$c = \left(f(c) \cdot d_1\right)^{1/(1+s)}. \tag{5.17}$$

Then, we use Equation (5.17) and insert Equation (5.15) into it:

$$\begin{aligned}
\left(\frac{h_{c,r}(c)}{d_1}\right)^{1/(1-s)} &= \left(f(x) \cdot d_1\right)^{1/(1+s)} \\
\frac{h_{c,r}(c)}{d_1} &= \left(f(x) \cdot d_1\right)^{(1-s)/(1+s)} \\
h_{c,r}(c) &= \left(f(x) \cdot d_1\right)^{(1-s)/(1+s)} \cdot d_1.
\end{aligned} \tag{5.18}$$

Finally, using the equality of heights from Equation (5.7), we obtain root scaling from frequency with

$$h_{f,r}(f(x)) = \left(f(x) \cdot d_1\right)^{(1-s)/(1+s)} \cdot d_1. \tag{5.19}$$

If the shrink factor is $s = 0$, then the root scaling boils down to linear scaling. This can be either trivially observed in Equation (5.13) or confirmed by inserting $s = 0$ into Equation (5.19) and comparing the result against Equation (5.12):

$$\begin{aligned}
h_{f,r}(f(x)) &= \left(f(x) \cdot d_1\right)^{(1-0)/(1+0)} \cdot d_1 \\
&= \left(f(x) \cdot d_1\right)^{1} \cdot d_1 \\
&= f(x) \cdot d_1^2. \qquad \blacksquare
\end{aligned}$$

5.6 Logarithmic Scaling

In column-based nonlinear dot plots, there is a user-defined base b for logarithmic scaling. Its value is at least $(\sqrt{5}+1)/2 \approx 1.6180$ and is used for the calculation of the column height instead of dot diameters:

$$h_{c,l}(c) = d_1 \cdot \log_b\left(c + b - 1\right). \tag{5.20}$$

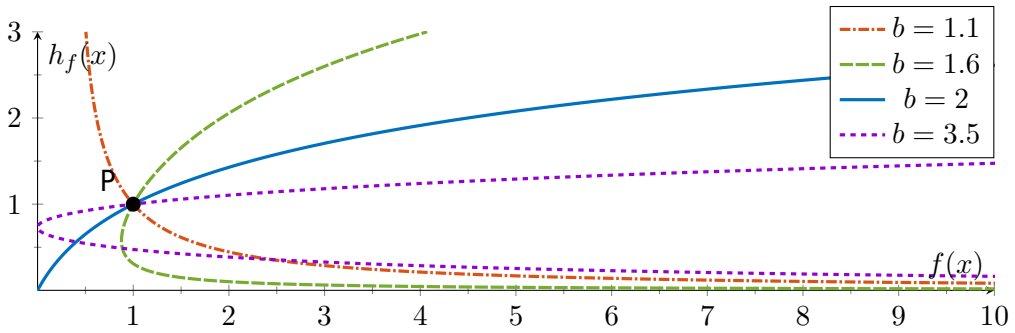


Figure 5.1: Plots of $h_{f,l}$ with varying values for the logarithm base b. The initial dot diameter is set to $d_1 = 1$.

Solving this equation for c gives:

$$\begin{aligned} h_{c,l}(c) &= d_1 \cdot \log_b(c + b - 1) \\ \frac{h_{c,l}(c)}{d_1} &= \log_b(c + b - 1) \\ b^{\left(h_{c,l}(c)/d_1\right)} &= c + b - 1 \\ c &= b^{\left(h_{c,l}(c)/d_1\right)} - b + 1. \end{aligned} \tag{5.21}$$

The next step is to insert this equation into Equation (5.5) to get a value frequency (without explicitly knowing the dot diameter) and to use Equation (5.7):

$$f(x) = \frac{\left(b^{\left(h_{f,l}(f(x))/d_1\right)} - b + 1\right)^2}{h_{f,l}(f(x))}. \tag{5.22}$$

We would like to solve this equation for $h_{f,l}$, but the height appears in both the main denominator and the exponent of b. Therefore, we use a numerical solution (nested intervals). This is an option because relaxed dot plots already know the value frequency $f(x)$ before calculating the envelope height. As Figure 5.1 shows, there is potential for multiple solutions, depending on b. Since dot plots should increase in height when more dots are stacked on top of each other, we can ignore solutions below d_1.

The column-based layout requires a lower limit for the base b. For the new frequency approach, we need to analyze Equation (5.22) to check for additional limits. Figure 5.1 gives concrete examples of $h_{f,l}$, allowing us to get an initial understanding of its behavior with different values of b. The function always goes through the special point $P = (1/d_1, d_1)$. This is the value frequency up to which regular dot plots could place a single dot without the need for stacking. We handle frequencies $f(x) \leq 1/d_1$ with a constant height of d_1, resulting in a piecewise definition for $\hat{h}_{f,l}$. For frequencies above this threshold, we only use solutions with $h_{f,l} \geq d_1$.

However, Figure 5.1 shows that the example of $b = 1.1$ forces the function to drop below the value d_1 (height of P).

The smallest possible value of b needs to provide a positive and finite slope at $h_{f,l}(1/d_1)$. We can write this as a constraint

$$0 < h'_{f,l}(1/d_1) < \infty \tag{5.23}$$

that is not simple to solve for b. In Equation (5.22), we actually defined how to get the frequency from height because it was not trivial to get the height directly. By switching the axes, we get a constraint for the derivative of $f(x)$:

$$0 < f'(1/d_1). \tag{5.24}$$

Now it is simpler to solve for b. First, we get the derivative of $f(x)$ and substitute $y = h_{f,l}(f(x))$ to improve readability:

$$0 < -\frac{\left(b^{y/d_1} - b + 1\right) \cdot \left(d_1(b^{y/d_1} - b + 1) - 2y \cdot \log_e(b) \cdot b^{y/d_1}\right)}{d_1 y^2}. \tag{5.25}$$

We already know the height $y = h_{f,l}(1/d_1) = d_1$ at our point of interest:

$$\begin{aligned}
0 &< -\frac{\left(b^{d_1/d_1} - b + 1\right) \cdot \left(d_1(b^{d_1/d_1} - b + 1) - 2d_1 \cdot \log_e(b) \cdot b^{d_1/d_1}\right)}{d_1 d_1^2} \\
0 &< -\frac{(b - b + 1) \cdot (d_1(b - b + 1) - 2d_1 \cdot \log_e(b) \cdot b)}{d_1^3} \\
0 &< -\frac{(1) \cdot (d_1(1) - 2d_1 \cdot \log_e(b) \cdot b)}{d_1^3} \\
0 &< -\frac{d_1 - 2d_1 \cdot \log_e(b) \cdot b}{d_1^3} \\
0 &< -d_1 + 2d_1 \cdot \log_e(b) \cdot b \\
0 &< -1 + 2 \cdot \log_e(b) \cdot b \\
1 &< 2 \cdot \log_e(b) \cdot b \\
\frac{1}{2} &< \log_e(b) \cdot b \\
e^{W(1/2)} &< b.
\end{aligned} \tag{5.26}$$

With the lower limit $e^{W(1/2)} \approx 1.42153$ of b, the height function will always have exactly one solution above d_1.

Another usual constraint for nonlinear dot plots is that the initial dot diameter is the largest, and dot sizes for higher value frequencies must not increase, i.e., for values $f(x) > 1/d_1$, the following inequality must hold:

$$h_{f,l}(f(x)) < d_1 \cdot (d_1 \cdot f(x)). \tag{5.27}$$

Using the inverse function $h_{f,l}^{-1}$ of the height, which is defined by Equation (5.22), we can rearrange the inequality:

$$\begin{aligned} h_{f,l}(f(x)) &< d_1 \cdot (d_1 \cdot f(x)) \\ h_{f,l}^{-1}(h_{f,l}(f(x))) &< h_{f,l}^{-1}(d_1^2 \cdot f(x)) \\ f(x) &< \frac{\left(b^{(d_1^2 \cdot f(x)/d_1)} - b + 1\right)^2}{d_1^2 \cdot f(x)} \\ (d_1 \cdot f(x))^2 &< \left(b^{(d_1 \cdot f(x))} - b + 1\right)^2 \\ d_1 \cdot f(x) &< b^{(d_1 \cdot f(x))} - b + 1 \end{aligned} \tag{5.28}$$

To show that this inequality holds for $f(x) > 1/d_1$, we define the function $\Phi(f(x)) = b^{(d_1 \cdot f(x))} - b + 1 - d_1 \cdot x$ and prove the equivalent assertion that $\Phi(\ldots)$ is strictly (monotonically) increasing for $f(x) > 1/d_1$. Using the derivative of $\Phi(\ldots)$, which is given by

$$\begin{aligned} \Phi'(f(x)) &= d_1 \cdot \log_e(b) \cdot b^{(d_1 \cdot f(x))} - d_1 \\ &= d_1 \cdot (\log_e(b) \cdot b^{(d_1 \cdot f(x))} - 1), \end{aligned} \tag{5.29}$$

we observe that $\Phi'(f(x)) > 0$ (which is equivalent to $\Phi(\ldots)$ increasing strictly) if and only if $\log_e(b) \cdot b^{(d_1 \cdot f(x))} > 1$. However, it suffices to show that this inequality holds for $f(x) = 1/d_1$ because, for values of $f(x)$ that are larger than $1/d_1$, it holds as well. Therefore, we can determine the limit by

$$\begin{aligned} \log_e(b) \cdot b^{(d_1 \cdot (1/d_1))} &= 1 \\ \log_e(b) \cdot b &= 1 \\ b &= e^{W(1)}. \end{aligned} \tag{5.30}$$

In summary, with a base $b > e^{W(1)} \approx 1.76322$, it is guaranteed that the dot sizes will decrease with higher value frequency.

5.7 Generic Dot Diameter

In relaxed dot plots, we do not know the height of inexistent columns. But we *do know* the height of a stack of dots: Equation (5.1). We also have the local frequency within a column: Equation (5.4). We can combine and rearrange both equations:

$$f_c(c) = h_c(c)/d_c^2(c). \qquad \text{(rearranged)} \tag{5.31}$$

Through the "magic" of KFE, we know the height of the envelope at any point along the x-axis. We can now derive the diameters by solving Equation (5.31) for $d_c(c)$:

$$d_c(c) = \sqrt{h_c(c)/f_c(c)}. \tag{5.32}$$

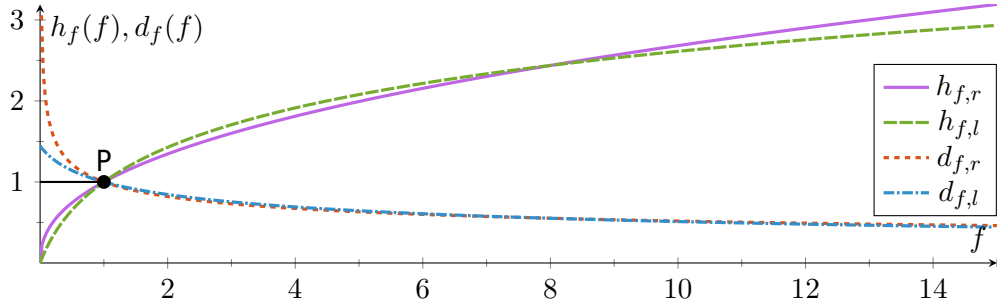


Figure 5.2: Plot of the functions for envelope height from frequency (h_f) and dot size from envelope and frequency (d_f). Parameters for root and logarithmic scaling are set to $d_1 = 1$, $s = 0.4$, and $b = 2$. All functions intersect in P. The black line toward P shows the piecewise function definition for very low data frequencies.

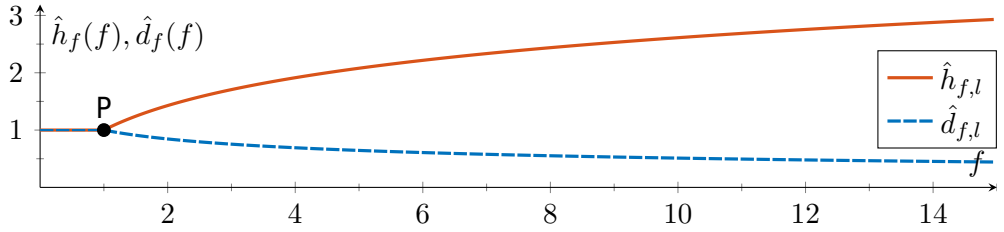


Figure 5.3: Plot of the piecewise functions for logarithmic dot plots ($\hat{h}_{f,l}, \hat{d}_{f,l}$). They ensure the use of the start diameter d_1 for low value frequencies (when dots are not stacked). Parameters are set to $d_1 = 1$ and $b = 2$.

In Equation (5.7) and Equation (5.8), we required frequency and height to be the same in both column-based and relaxed layouts. Therefore, we can use the generic function

$$d_f(f(x)) = \sqrt{h_f(f(x))/f(x)} \tag{5.33}$$

for the individual frequency-based diameter of each dot (see Figures 5.1 to 5.3).

5.8 Conslusion

In this excursion, we provided frequency-based functions for both original nonlinear scaling functions from Chapter 3. Other potential column-centric scaling functions would need to go through similar transformations to be usable with relaxed dot plots. However, because of the advantages of the relaxed layout and to avoid interference from the two-dimensional effect of shrinking dot sizes, we strongly suggest purely frequency-based scaling

functions for newer developments. The presented approach for individual dot sizes from Section 5.7 is generic and does not require any adaptation for future scaling functions.

Part II

Scatter Plots

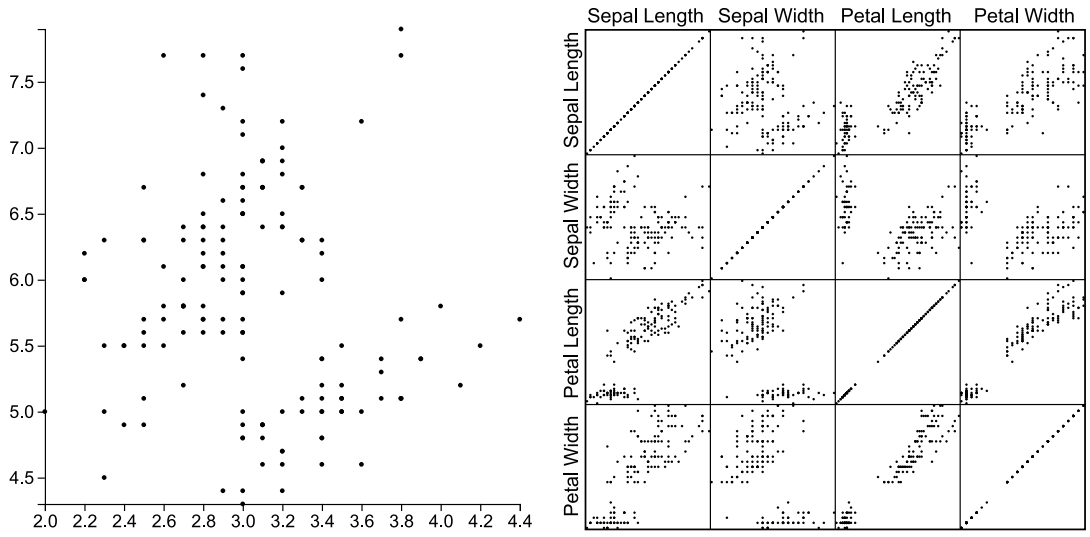


Figure II-1: A single scatter plot (left) and matrix (right) of the iris data set [Fis36]. The large scatter plot shows only a single 2D subspace: sepal width on the x-axis and sepal length on the y-axis.

So far, we have discussed mainly 1D data. We visualized a primary dimension and mapped a secondary attribute to dot color. But this is not always sufficient. When we want to analyze two equally important numerical data dimensions, scatter plots are often the visualization of choice. They use orthogonal display dimensions to represent data dimensions and position a dot for each data point [CM88; Utt05]. The example in Figure II-1 shows how we need the combination of both dimensions to separate flowers of the species *setosa* (the elongated cluster in the lower right) from the remaining samples. Similar to dot plots, we can extend the accurate positional display by mapping secondary attributes to the visual appearance of each dot, e.g., shape and color. Scatter plots are relatively easy to use when two dimensions are sufficient. An aspiring student can look at a plot of prestige vs. costs with dots representing multiple universities and decide where to study. Medical professionals can visualize blood pressure and heart rate to estimate the risk for cardiovascular diseases.

But what happens when there are even more interesting and potentially important data dimensions? Various dimensionality reduction techniques can project the data onto a 2D plane [Jol02; MHM18]. However, the resulting axes do not represent any original data dimension, making it difficult for viewers to link a display position back to the underlying values. One could switch between multiple plots with the same underlying data but with different selected dimensions for display. While this does not require additional skills over regular scatter plots, it is difficult to track individual points and clusters when the dots jump between positions in the visualizations.

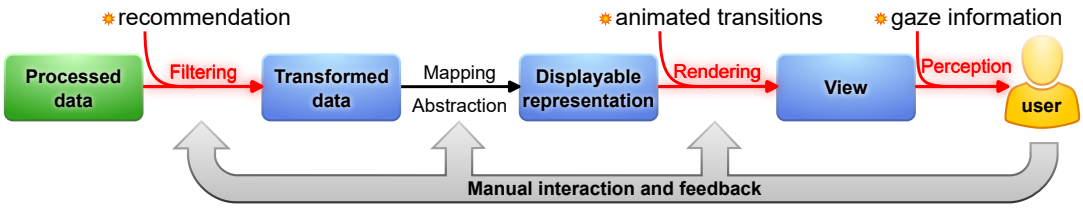


Figure II-2: The visualization pipeline (adapted from [CMS99]) with additions for the proposed techniques from this part. Affected elements are highlighted in red. Animations help users to trace points and clusters between plots. Gaze implicitly selects dots that are used to recommend 2D subsets of data.

This is where the reasoning behind RQ2 becomes relevant. We want to investigate ways to better navigate between different scatter plots of the same multivariate data set. There is previous work that suggests animating the dot positions between different views [EDF08]. We already discussed a similar concept in Section 4.9. There, we suggested using the iterative nature of the relaxation algorithm for dot plots to animate changing, appearing, or disappearing data points. Smooth movement would also be suited for viewers to find the corresponding dots of two scatter plots. In Chapter 6, we will implement a selection of new and previously suggested animations for scatter plots and evaluate how well they serve to trace individual points and clusters. Solving the issue of correspondence between two static scatter plots would also be of benefit to the explainability of dimensionality reduction (DR) techniques. Viewers could switch between the original data dimensions and the projections to enhance information gain.

Animation between only two views is not sufficient to answer RQ2. As mentioned in Chapter 2, the number of possible dimension pairs for scatter plots grows with the square of the number of underlying data dimensions (D^2). The many thumbnails of a SPLOM only scale to some degree before they become too small to give the viewer meaningful insights (see the example for $D = 4$ in Figure II-1). Chapter 7 will present a proof of concept for a novel approach to help navigate the space of possible dimension pairs. It uses eye-tracking hardware to log the user's gaze and infer data points of interest. The technique provides recommendations for the user to manually select and analyze other views by ranking the dimension pairs by (dis)similarity of various metrics for the relevant data points.

Our work on RQ2 shows how eye tracking can improve the workflow of 2D visual analytics software. The excursion in Chapter 8 presents potential benefits for immersive and situated analytics and analyzes the use of 3D eye tracking in augmented reality.

Overall, the techniques and experiments in this part of the thesis will affect various elements of the rendering pipeline shown in Figure II-2. Animation does not add nor remove data points and will not alter their visual mapping. It only introduces a step between two views in the rendering part. Recommendations help the user manually select a different 2D subspace from the multivariate data set. Therefore, they affect the interaction with data filters. Eye tracking reveals how the user looks at the plots, what data points might have been in the focus of sharp vision, and in what order. Eye tracking can also provide additional physiological measures that can be interpreted to gain insight into human cognition [Eck+17; ZP23]. Within this thesis, however, we only record the measured gaze positions and use them to infer the distribution of the viewer's interest in the input data.

6

Comparative Evaluation of Animated Scatter Plot Transitions

This chapter is based on preliminary work that was later extended and published. It uses extracts from the publication without explicit quotation: N. Rodrigues, F. L. Dennig, V. Brandt, D. A. Keim, and D. Weiskopf. “Comparative evaluation of animated scatter plot transitions.” In: *IEEE Transactions on Visualization and Computer Graphics* (2024). Early access, p. 11. DOI: 10.1109/TVCG.2024.3388558.

6.1 Motivation

Scatter plots are a useful visualization for a pair of numerical dimensions [CM88; Utt05]. But data often has more than two attributes. For efficient exploration and analysis in such a case, it is helpful to visualize the same data samples in multiple views that show different dimensions [Ngu+20].

Up to a relatively low number of dimensions, it is possible to visualize the data with combinations of full-sized scatter plots [Ngu+20] or with a SPLOM for a focus+context [BGS01] approach. However, as the number of data attributes increases, more and smaller plots are necessary. At the same time, viewers experience a higher cognitive load to link the separately shown data points from many plots in a single coherent mental map. Computational methods for DR [Jol02; MHM18] allow showing data sets with arbitrary values of D in a 2D scatter plot. The resulting *artificial* display dimensions can summarize a data set but are challenging to interpret and trace back to the original data attributes. Therefore, users of both approaches—SPLOM and DR—might benefit from switching between views with various pairs of original and artificial dimensions.

Nguyen et al. [Ngu+20] show that using a limited number of simultaneously visible scatter plots (called “multiple-scatterplots technique”) works better than having the plots appear sequentially. In their study, the visualizations are linked and share the same brushes. We argue that a setup with such a small selection of visible data dimensions is especially useful if the analysts search for answers to specific questions or already know the characteristics of the data set. For the exploration of a new data set, the users might not know how relevant each dimension is and what attribute to use for selecting and highlighting hundreds of data points across all plots.

Analysts need to be able to trace data points between the different views—presented sequentially or simultaneously—to work efficiently. There are multiple possible solutions to the issue of correspondence between dots in different plots. For instance, if the data set contains only a few samples, the plots could show them with varying shapes or colors without requiring explicit user input. However, typical humans are not able to remember all intricacies that would be necessary to distinguish dozens of samples [Mil56].

Animation is a well-known technique for users to solve correspondence tasks [CDF14] and supports data sets with hundreds of samples. As a benefit, other visual variables, e.g., color and shape, remain available to encode additional and better-fitting data attributes. Various animations for transitions between different scatter plot views have been suggested in prior work [Asi85; EDF08; Du+15; KCH19]. To answer  RQ2, we want to explore the design space of animations to identify fitting animation types for the tasks of tracing points and cluster interactions. We aim for ecological validity beyond tasks with only a few simultaneously traceable data points. Thus, this chapter compares multiple spline-based and 3D-rotation-based transitions in a crowdsourced study with hundreds of data points. We use a custom framework for animation, written as a plugin for *D3.js* [BOH11]. The recorded study data, the source code of the animation framework, and an interactive demo of the study are available as separate publications [Rod+23a; BR23; Rod23].

6.2 Related Work

The study behind this chapter of the dissertation relates to previous work on multiple aspects. First and foremost, we will discuss animation techniques for scatter plots. Then we will go into studies with animated visualizations as stimuli. Finally, we will touch on the topic of eye movement in the form of smooth pursuit because participants will be trying to follow moving points and clusters in this study.

6.2.1 Animated Visualization in Other Contexts

Animation is not a new concept, and there have been multiple publications on the topic. Moving visualizations are often used to show temporal changes or trends in the underlying data [TK07; Rob+08; KPS12]. Gapminder[1] is one such example, where the movement of dots in a scatter plot encodes the changes of country-level metrics over time. There are also static visualizations for changes over time that can be used effectively [Rob+08; Per+18]. However, RQ2 does not focus on mapping time to a dedicated third dimension or to the movement of data points. In our work, we want to evaluate the use of animation purely for correspondence across the transition between two scatter plots of different attributes from the same data set.

Storytelling is another field of research with multiple publications [DF15; TLS21; Cao+23] about the use of animation. RQ2 targets the interactive visual exploration and analysis of data sets. Hence, we will not evaluate the suitability of animation for storytelling but only for tracking individual data points and interactions between clusters. Animation has also been used to transition between different types of visualization [DF15; Tom+21] and to show how data is aggregated [KCH19]. Again, both use cases do not fit the goal of the research question behind this chapter of the dissertation.

Animated visualization is suited for the analysis of moving sources of data. This is especially noticeable in the context of sports, where situations from a game can be analyzed post hoc to learn about the strategy, strengths, and weaknesses [FOO23]. However, even in such a case, the animation is tightly linked to the dimension of time.

6.2.2 Animations for Scatter Plot Visualizations

Thompson et al. [TLS21] used animation for the correspondence of data points between different plots. In their work, however, all visible data remained constant, and only the plot type changed. Our motivation in the context of RQ2 is more related to work like *the grand tour* by Asimov [Asi85]. His work produces an animation through a sequence of projections onto 2D planes and is supposed to show the shape of data by taking into account all D dimensions. As Asimov himself stated, the "movies" with the projected data are difficult to understand, even with only 4 data dimensions, and require extensive experience with the visualization. Our approach evaluation targets the correspondence between pairs of the original or projected data dimensions. The tested animations only move

[1] `https://www.gapminder.org/tools/#\protect\T1\textdollarchart-type=bubbles&url=v1`, accessed 2024-01-27.

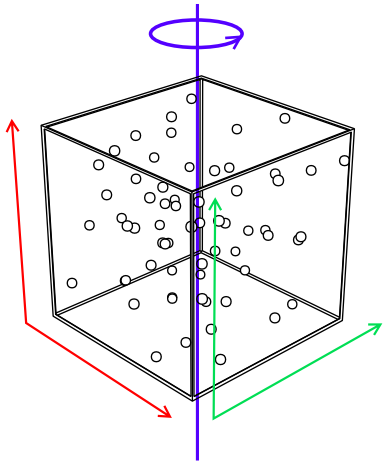


Figure 6.1: The generic concept of rotation for exchanging a single dimension in a scatter plot. Adding a temporary perpendicular depth dimension to a 2D scatter plot creates a cube. Rotation around an unchanged axis (e.g., height) reverses the roles between axes for depth and width.

the visible points between their positions in two scatter plots and are not aimed at generating a "movie" that spans all data dimensions. We argue that short transitions and longer periods for observation of a single scatter plot are closer to real-world applications of visual analytics.

The method by Elmqvist et al. [EDF08] is an example of what we want to evaluate and compare. It starts with a regular 2D scatter plot and temporarily extends it by adding a third dimension for depth. A rotation around an unchanged axis, e.g., height, results in role-reversal between the width and depth axes (see generic concept in Figure 6.1). Afterward, the new depth axis is removed and the result is simply the exchange of one data dimension for another. The proposed concept of "rolling the dice" goes even further and creates sequences of multiple rotations [EDF08]. It provides users with controls to navigate the space of all 2D scatter plots within a matrix and preserves brushed data points between views. Sanftmann and Weiskopf [SW12] also propose an analysis tool with a technique that is mostly 3D. It provides a 3D SPLOM for navigation as well as a 3D scatter plot for details and only reverts to 2D when showing thumbnails on the surface of the SPLOM. Furthermore, they present a technique for smooth animations while changing one or two dimensions simultaneously. The resulting projection provides consistent motion of dots. While we implemented 3D rotations of scatter plots, our focus is not on a full-fledged visual analytics tool.

In contrast to the previous approaches, Heer et al. [HR07] propose effective design guidelines for creating transitions between different types of statistical charts, e.g., scatter plots and bar charts. Kim et al. [KCH19]

present staged transitions between visualization types to depict aggregation. Both works tackle other problems related to animated plots. Our study compares and evaluates which scatter plot transition is most effective for the user. All approaches proposing transitions between scatter plots evaluated their techniques without comparisons to existing methods. We address this specific research gap in our quest for an answer to RQ2.

6.2.3 User Studies on Animated Visualizations

There is prior work on the evaluation of animated visualizations. Interaction with and cognitive effects of animated visualization are the focus of two qualitative studies by Nakakoji et al. [NTY01]. While they report animation to be beneficial, they also identify core challenges for designing interactive and animated visualization. Yao et al. [Yao+22] present a research agenda for visualization in motion and analyzed how well viewers could read data from moving charts. However, their focus lies on "entire visualizations moving, rather than data [points] [...] or individual parts of [the] visualization [having] different [...] directions, trajectories, or speeds."

Animation is often used for visualizing time-oriented data [KPS12]. Financial data, for example, typically appears as time-series and can be animated for analysis. Tekušová and Kohlhammer [TK07] evaluated their system with a usability study that included novices and experts from the financial industry. Their approach focused on user experience and system design. Work by Robertson et al. [Rob+08] contains a quantitative evaluation of different methods for visualization of trends in data: traces, small multiples, and animation. The results include qualitative feedback and a comparison between the techniques with respect to how accurately study participants can extract information from the underlying data. Animation is reported to perform worse but provide more enjoyment to the users. A newer quantitative study by Brehmer et al. [Bre+20] also compares animation against small multiples of scatter plots. Its results confirm that viewers of small multiples complete tasks objectively faster but with subjectively lower confidence. Our study also uses objective measures for task performance. We record subjective user ratings to adjust the perceived overall speed and task difficulty.

Our goal with respect to the main research question RQ2 is not to provide a new analysis method for information gain but to help humans make better use of the already existing and intuitive scatter plots. We analyze the use of animation for correspondence between data points in two static visualizations. Therefore, we restrict the evaluation to the traceability of visual elements and do not include tasks that require interpretation of the underlying data.

There are parallels between our work and previous studies with regard to visual primitives and animation techniques. The first study, by Chevalier et al. [CDF14], targets the viewer's ability to follow up to three of 30 animated dots. It compares the condition of simultaneous movement against various movements with temporal offset, i.e., *staggered* animation. The authors controlled for task difficulty by proxy of various metrics that correlated with task performance. The study results show that "staggering has a negligible, or even negative, impact on multiple object tracking performance." A second study by Du et al. [Du+15] compares straight paths against bundled paths for animated dots. The results indicate that bundling is of benefit when working with more data points or when there is a higher level of occlusion between dots. Our study is similar because it identifies groups of data points and uses time offset and trajectory bundling to animate dots between scatter plot views. However, we examine more types of animations and use a real-world data set with more samples to get results that are more representative of visual analytics tasks. We strive for ecological validity beyond relatively small laboratory setups.

Our study requires participants to follow moving points and clusters. As some visual targets have constant movement, we expect the onset of smooth pursuit eye movements [PW01] for parts of the transitions. Smooth pursuit maintains a static image of a moving target on the fovea. This allows the environment of the target to change, which might lead to motion blur. Previous eye-tracking studies show that the movement direction of targets has an impact on various metrics related to smooth pursuit [Rot+96; MFE19]. Our study is not primarily focused on the analysis of the human ability to control such smooth pursuit eye movements, and in Section 6.4, we describe how we try to avoid large influences from smooth pursuit. While we do not expect it, we acknowledge that there might be an effect of animation direction on the performance of tracking a visual target.

6.3 Design Space

The following section describes the design space for modeling animated transitions between different views in a scatter plot matrix. Here, we also describe what animations we implemented for inclusion in the study.

6.3.1 Dimensions and Views in a Scatter Plot Matrix

There are a total of D^2 2D views for a data set with $D \in \mathbb{N}$ dimensions. Scatter plots with two identical dimensions position the data points on a diagonal line, providing little additional insight. They are located on the

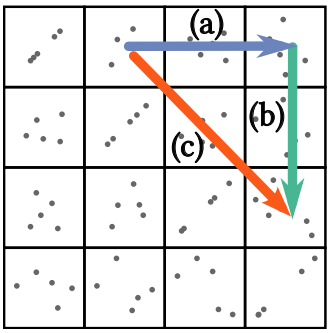


Figure 6.2: 1D and 2D transitions in a SPLOM. Each square represents a cell of the SPLOM. Arrow (a) depicts a horizontal 1D transition, (b) a vertical 1D transition, and (c) a 2D transition.

main diagonal of a scatter plot matrix. The upper and lower half (above or below the diagonal) show mirrored but identical dimension combinations. Thus, there are only $(D^2 - D)/2$ non-mirrored views of interest.

In this work, we define the combination of two dimensions x and y of a data set as $D_x \times D_y$. A view change from $D_1 \times D_2$ to $D_3 \times D_4$ does not introduce or remove data points. This change only alters their position in the scatter plot from P_s to P_e with $P \in$ display dimensions $D_x \times D_y$. Therefore, the data points can be animated on a path between P_s and P_e when transitioning to another combination of data dimensions in the scatter plot. This technique avoids sudden changes in position and helps preserve the viewer's mental map. Since scatter plots visualize two data dimensions, there are two different cases when analysts switch between plots (shown in the context of SPLOMs in Figure 6.2):

1D Transitions only exchange one of the two plot dimensions for a new data dimension.

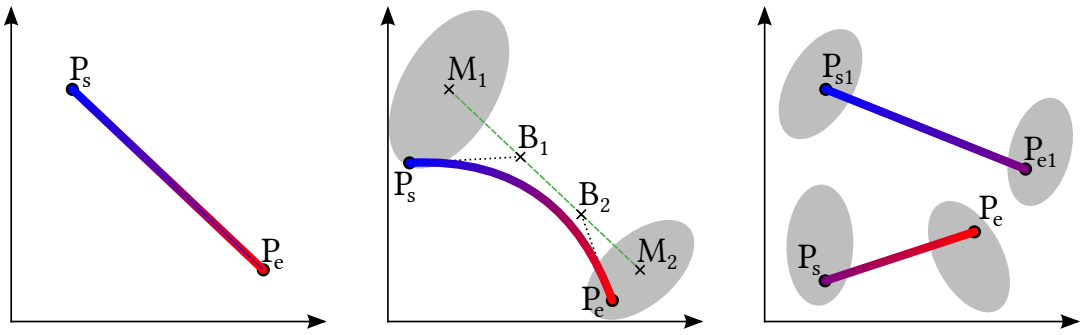
2D Transitions exchange both dimensions of the plot.

6.3.2 Transitions

This subsection describes the types of transitions we evaluated in our quantitative user study. We consider two types of point movements: (1) *spline-based transitions* and (2) *rotation-based transitions*.

Spline-Based Transitions

All spline-based transitions work independently of the number of exchanged dimensions, i.e., 1D or 2D transition, since they only require a start point P_s and end point P_e.



(a) straight **(b)** bundled **(c)** time offset

Figure 6.3: Examples of spline-based animation paths for point movement in transitions between scatter plots. Gray ellipses symbolize the area covered by clusters. Time is encoded as path color (start end).

Straight lines (STR) are a special case of spline-based movement. The path between P_s and P_e is determined by simple linear interpolation, resulting in a straight line (see Figure 6.3a). In the case of 1D transitions, the points move parallel to the changed dimension.

Bundled lines (BUN) use clustering and control points to create splines. In the first step, the DBSCAN algorithm [Est+96] determines a set of clusters C_s and C_e in the start and end pair of normalized data dimensions. Secondly, we group all data points that belong to the same pair of clusters $C_{s,i} \times C_{e,j} \in C_s \times C_e$. We calculate the mean position M_i and M_j of all points in $C_{s,i}$ and $C_{e,j}$. For each group, we select a set of control points B on the straight line between M_i and M_j. Finally, we compose the splines for each data point's path in the transition by combining the individual start and end positions with the common control points of each group.

Time offset (TIM) is a method that combines the technique from STR and information about cluster pairs ($C_{s,i} \times C_{e,j}$ from BUN). However, each pair of clusters is animated one after the other at a different time offset, such that each cluster is animated separately.

Rotation-Based Transitions

All rotation-based transitions use a 3D cube (see Figure 6.4 and Figure 6.5). To perform a 1D transition, we encode the new data dimension as depth and then rotate the cube around the unchanged dimension (see Figure 6.1). We realize 2D transitions through a sequence of 1D transitions: vertical first, then horizontal; or horizontal first, then vertical. In scatter plot matrices, this method corresponds to moving along columns and rows to arrive at the desired view (see Figure 6.2 (a) and (b)).

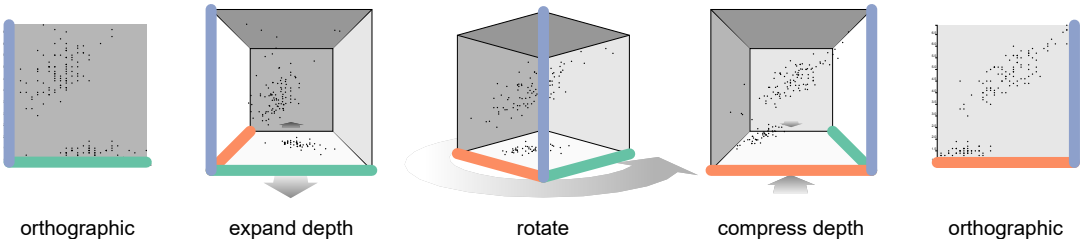


Figure 6.4: Staged rotation (STA) with a 3D cube happens in three sequential stages (left to right). The scatter plots at the beginning and end of the transition share one data dimension. The thick colored lines represent data dimensions during the transition

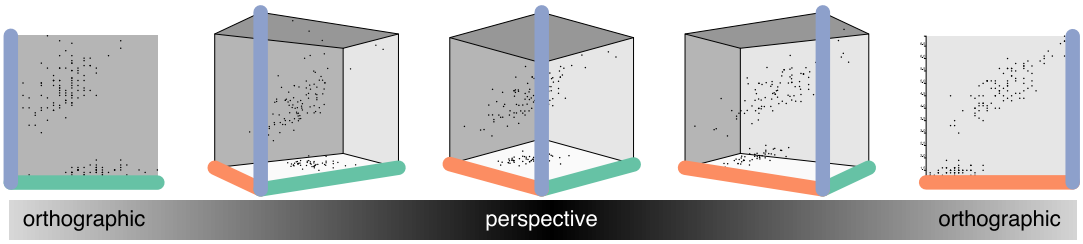


Figure 6.5: Perspective rotation (PER) has no dedicated stages (left to right). It gradually changes from an orthographic to a perspective camera and back again while simultaneously rotating the cube. The thick colored lines represent data dimensions during the transition.

Staged rotation (STA) is our implementation of the method by Elmqvist et al. [EDF08]. Figure 6.4 shows the individual steps of this transition. As preparation, we encode the new data dimension as depth (z-axis). However, this depth is not perceivable because we use an orthographic camera. In the first stage of animation, we move toward a perspective camera, turning the depth information visible and extending the new data dimension as the depth dimension (z-axis) of the scatter plot. Rotation of the cube is the second stage. A third and final stage collapses the z-axis and returns to an orthographic camera.

Perspective rotation (PER) works similarly to STA rotation but without stages. The extension and collapse of the depth axis take place while the cube rotates. Figure 6.5 shows the concept.

Orthographic rotation (ORT) does not use a perspective camera. Therefore, it lacks an explicit display of depth information, and like the PER rotation, this transition does not have any stages.

6.3.3 Non-Evaluated Alternatives

Our *D3.js* [BOH11] plugin provides a framework for further animations. Splines could theoretically animate points between more than two scatter plots simultaneously. Instead of using cluster-related control points, we could use the positions of data points in intermediate plots as control points. This would even allow for animations similar to the *grand tour* [Asi85]. However, the longer the animation, the more opportunities to lose track of data points or clusters of interest. The latter is especially true as the clusters change between the individual subspaces. We want to evaluate the animation techniques with a fair and practically relevant setup that allows analysts to get an overview of the data and extract finer details. Our selection of transitions matches the tasks of tracing individual points and clusters. While the work of Elmqvist et al. [EDF08] inspires the rotation animations, we specifically designed the BUN and TIM techniques to potentially help the viewer to trace clusters.

6.4 Hypotheses

Now that we have a list of six animated transitions, we want to evaluate them comparatively. Analysts extract information from data visualizations with various methods. While animation could encode a data dimension by itself, e.g., by showing changes over time, the selected transitions are only meant to help preserve the analyst's mental model of the data. We state the following four hypotheses to explore this notion more precisely.

Our main goal is to evaluate the chosen animations concerning how well a single data point can be traced from one plot to the next. Thus, we pose the following hypothesis to evaluate the tracing performance.

Hypothesis HP: *There is an animation-dependent difference in task performance for point tracing.*

It is relatively easy to find clusters in the 2D subspace of a single scatter plot. But it is nontrivial to trace how the clusters evolve between two scatter plots with different axes. We suspect that animation can show how the clusters interact between 2D subspaces. Therefore, our work aims to evaluate how well interactions between clusters of different scatter plots can be traced using the selected animations.

Hypothesis HC: *There is an animation-dependent difference in cluster task performance.*

Previous work suggests a difference in the ability of humans to follow objects vertically vs. horizontally through eye movement [Rot+96; MFE19; PW01] (see also Section 6.2.3). In practice, analysts need to be able to

trace points in both directions. Therefore, we use scatter plots that are small enough to trace data points without large eye movements. With these assumptions, we expect a negligible influence on task performance from vertical vs. horizontal dot movement. However, statistical tests can only detect a difference. Hence, we will perform tests for a difference and interpret a lack thereof as an indication of a negligible effect.

Hypothesis HD: *There is no direction-dependent difference in task performance.*

In addition to the purely objective task performance with points and clusters, we also want to compare the animation techniques concerning subjective user ratings. We want to evaluate whether the user sees animated scatter plot transitions as valuable. We do this in a similar fashion to Robertson et al. [Rob+08]. However, they explored whether users found animation generally beneficial.

Hypothesis HR: *There is an animation-dependent difference in subjective user rating.*

6.5 Methods

Our main goal is to comparatively evaluate the different animation techniques for transitions between different scatter plots of the same scatter plot matrix. To this end, we designed and preregistered [Rod+22a] our study before collecting and analyzing the data. In the following, we describe the setup of our study in detail.

6.5.1 Tasks

Point Task: As mentioned in Section 6.4, we expect the animation to have an impact on how well users can trace a point during the transition between two plots. To check **HP**, we show a scatter plot and highlight a single point. The participant should then follow this point along its animation path and mark a position where the point came to rest. The viewer clicks anywhere within the final scatter plot to indicate the estimated position. We determine the error by measuring the distance between the actual data point and the selected location.

We use the existing auto-mpg data set modified by Quinlan [Qui93] to measure task performance under realistic conditions. It has 398 data points with 8 attributes. Some attributes are continuous (e.g., weight), while others have only discrete values (e.g., number of cylinders). The latter data type is not particularly well suited for scatter plots or animation

as it may lead to a high degree of overplotting and, subsequently, hinder the identification of individual dots. However, in practice, data does not always fit the visualization technique, and data dimensions with few discrete values are part of realistic use cases.

Cluster Task: For this study, we identified three types of possible interactions between data clusters in different views of a SPLOM:
(1) no interaction, the cluster *remains* mostly as it was, (2) the cluster *splits* into multiple smaller clusters, and (3) the cluster *merges* with another. Outside of this study, other interactions are theoretically possible. For instance, a new cluster forms, and an existing one dissolves or spreads. However, they require users to calculate or intuitively estimate how densely the data points need to be packed concerning their surroundings to form a cluster. Depending on the individual, this might be highly subjective and yield different results. Thus, we limit our study to the three described interactions between data clusters.

Finding a real data set with only the presented three cluster interactions is not a trivial task. A variation in the total number of visible clusters might influence viewers. Even if they lose track of the cluster they are trying to follow, a decrease in total clusters might suggest a merge. In order to reduce confounding effects, we use a generative model [Sch+16] to produce consistent cluster interactions. The resulting data set contains 600 data points that are distributed into 5 clusters (80 points each) and 200 randomly positioned distractors. The cluster task has more dots than the data set for the point task. This corresponds to real-world applications, where the targets for analysis of larger data sets often shift from points to clusters.

We implement the interactions by moving half the points from one cluster to another. Depending on the point of view, this is either a *split* or a *merge*. The other clusters are not involved in any interaction (*remain*). We choose specific target clusters for the study task to ensure that all three conditions appear equally often. During the study, the initial view highlights all cluster data points that should be traced. When the animation begins, all dots revert to the same look. Afterward, the participants can select the perceived cluster interaction by clicking one of three buttons. We use the ratio of correct answers as a measurement for the study.

6.5.2 Procedure

As preparation for the study, we randomly select combinations of axes and a data point or cluster to trace. We compose three combinations of start and end views for transitions. From these, we create one horizontal and

one vertical 1D transition. Then we generate three more combinations and use them for 2D transitions. We replicate them for each start direction of rotation-based animations (horizontal first, then vertical; vertical first, then horizontal). Spline-based animations, however, have no “first” axis of movement. Therefore, we generate three additional combinations. Each animation is tested with 2×3 1D and 2×3 or six 2D transitions to provide the necessary number of trials for the subsequent statistical analysis.

We use the same tests for all animations and participants, albeit in random order. We test the point task and the cluster task in sequence to avoid frequent switches: 12 point tasks for each animation first, then 12 cluster tasks for each animation. Since every participant gets to perform every task with every animation, we have a within-subjects study design.

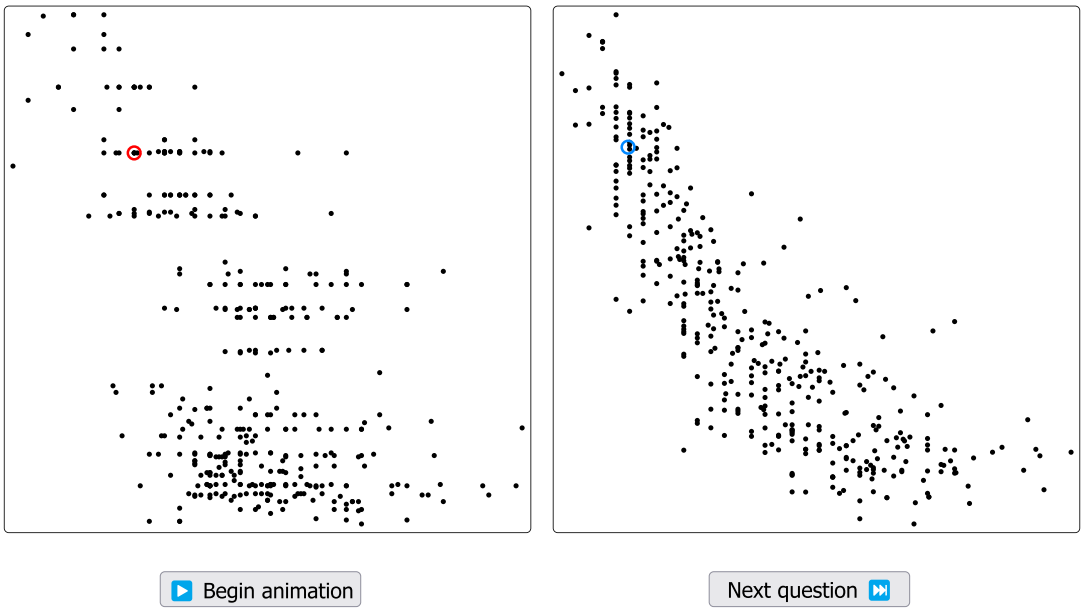
It is important to choose the dots for the point task randomly and re-use the same selection with all animations and participants. The dot density and overdraw around each data point might have a confounding influence on the task difficulty [CDF14; Bre+20], but there might be other still unknown factors. At the time of writing, we are not aware of a metric that reliably yields a complete and robust task difficulty for tracing a data point. Even if there were a metric that worked on a single static scatter plot, the target point would move during the animation. What single value would then represent the potentially time-varying task difficulty? In a nutshell, we have no method to calculate a complete and reliable metric for difficulty and, therefore, mitigate confounding effects through random selection.

During the study, participants are shown a static scatter plot with a highlighted dot or cluster (see Figure 6.6). They click a button to start the animation. Afterward, they select a target position and click to submit their answer for the point task. For cluster-related tasks, the participants single-click one of the buttons representing a cluster interaction (see Figure 6.7) to indicate what they have perceived.

We append a short questionnaire to each block of tasks for an animation. There, we ask for subjective feedback on these statements: (1) The animation was not too fast. (2) The paths of individual points / clusters are easy to follow. (3) I would like to use this animation frequently. Participants respond on a 5-point Likert scale: strongly disagree (-2), disagree (-1), neutral (0), agree (+1), strongly agree (+2). Generally, we regard the Likert scale as having ordinal values and only use the associated numbers on a rational scale when calculating mean values and standard deviation.

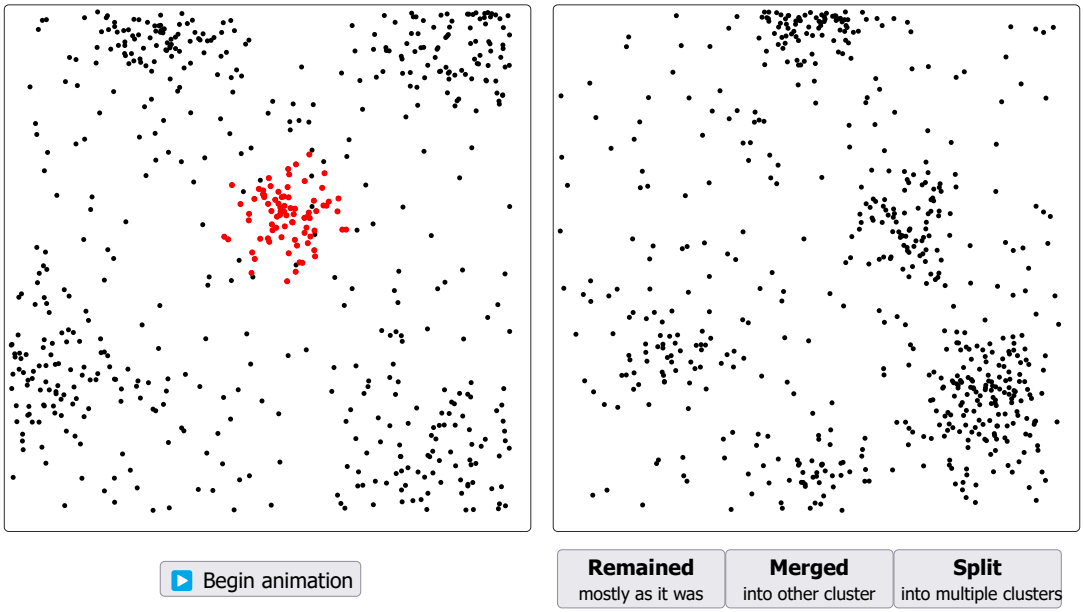
6.5.3 Training

At the beginning of each task type, we introduce how the task should be performed and how the participants can submit their answers. Before



(a) before animation **(b)** after animation

Figure 6.6: User interface for the point task. The point to follow during the transition is highlighted in red. After the transition, participants indicate a position in the scatter plot (highlighted in blue) and manually advance to the next animation.



(a) before animation **(b)** after animation

Figure 6.7: User interface for the cluster task. The cluster to follow during the transition is highlighted in red. After the transition, participants decide whether the cluster *remained* the same, *merged*, or *split*.

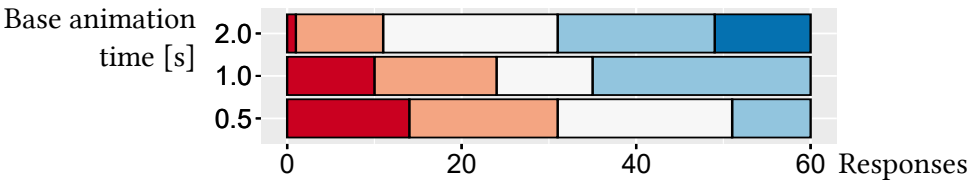


Figure 6.8: Results of the pilot study with different animation speeds. Participants rated their agreement with the statement "the animation was not too fast" on a Likert scale: strongly disagree, disagree, neutral, agree, strongly agree.

each new animation, we provide four task instances as training. They are generated in the same way as the regular tasks, but we provide feedback after the participant submits a response, i.e., for the point task, we reveal the final position of the dot that should have been followed, and for the cluster task, we show whether the provided answer was correct or wrong.

6.5.4 Animation Speed

We assume that the speed of the animations has a large effect on the measured performance. All tested animations have the same base duration. However, we extend the total duration of STA to 250 %. This allows us to maintain a similar rotation speed while providing extra time for the staged switch to and from the perspective camera. The TIM animation gets 200 % of the base runtime to limit the speed of the sequential animation of each cluster pair.

We performed a between-subjects pilot study with a similar setup to the main study but varied the movement speeds to get a range of base animation times: 0.5, 1, and 2 seconds. We had five participants for each condition but did not analyze the actual performance data to avoid researcher bias in the main study. Instead, we compared their subjective feedback regarding the animation speed ("the animation was not too fast"). Figure 6.8 shows the results. The variant with 1 second is best suited to find significant differences between the animations because it has similar agreeing and disagreeing scores. If all participants perceived the speed to be well within their limits, they could trace most data points' paths perfectly. For too low scores, the selected point positions might be random.

6.5.5 Data Set Size

As mentioned in Section 6.5.2, dot density and overdraw might have a confounding effect and are related to the number of points in the visualized data set. We argue that an overall rise of dot density has only little effect

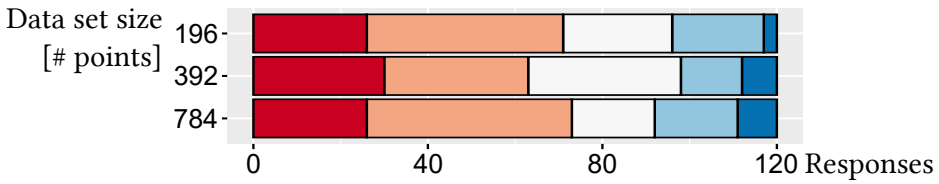


Figure 6.9: Results of the pilot study with different point counts. Participants rated their agreement with the statement "the paths of individual points are easy to follow" on a Likert scale: strongly disagree, disagree, neutral, agree, strongly agree.

on cluster tasks but might be present a crucial factor for tracing a single point. Thus, for ecological validity, it is sufficient to examine varying data set sizes for point tracing only.

We performed a between-subjects pilot study with a similar setup to the point task in the main study but varied the size of the auto-mpg data set used in the plots through random sampling and augmentation: 196 points (50 %), the original 392 points (100 %), and 786 points (200 %). The resulting data sets were similar in visual appearance and shape, varying mostly in point density. To get comparable results, we used the same target dots as in the main study and ensured their availability across all conditions.

We had 20 participants for each condition and compared subjective feedback regarding the ease of the task instead of the actual task performance to avoid researcher bias. Figure 6.9 shows that participant ratings yielded very similar results across all conditions. While the point tracing task is rated as challenging, the different dot counts have no significant effect on the perceived task difficulty.

Previous perceptual studies inferred difficulty from dot density and overdraw around individual points [CDF14; Bre+20] but had lower overall dot counts than the data sets in our pilot study. The subjective feedback from our 60 participants suggests that the random selection of target dots is sufficient to control for task difficulty. We argue that despite low subjective ease, the chosen auto-mpg data set with 392 points is a realistic fit for our main goal of evaluating animated transitions in the context of visual analytics and that the task difficulty might reach saturation as the dot count increases.

6.5.6 Power Analysis

Prior to preregistration, we performed a power analysis. We did not know the shape of the distributions of the measurements a priori. Therefore, we use a Wilcoxon signed-rank test for the point and cluster tasks. Power analysis for a medium effect size ($dz = 0.3$) yields a sample size of 170

participants. The Likert scales in the questionnaires require an analysis with χ^2 tests of independence. Power analysis for an effect size of $w = 0.3$ results in a sample size of 220 participants. We choose to go forward with only 170 participants and accept the altered effect size of $w = 0.342$ for the χ^2 test.

6.5.7 Participant Recruitment

We crowdsource participants for our 50-minute study on Amazon Mechanical Turk[2] and Prolific[3]. They earn approximately 12 EUR per hour. Of all participants, 89 are male, 80 female, and one reported "other or prefer not to say." The median age is 30 years, with a standard deviation of 9.19 (recorded in intervals of 10 years). Our data also includes self-reports on a Likert scale (-2 to 2) about *familiarity* ($Mn = 0.31$, $SD = 1.21$, $Mdn = 1$) and *regular use* of scatter plots ($Mn = -0.78$, $SD = 1.05$, $Mdn = -1$).

We employ attention tests to ensure high data quality (see Section 6.6.1). To comply with Prolific's policy for attention checks, we extend each animation-targeted questionnaire with nonsensical items, e.g., "I'm an elephant living on the moon." We reject submissions from Prolific, where less than 75 % of the nonsensical items are answered correctly. Finally, we ensure at most one submission from each participant on each platform.

When participants arrive at our website, they get a study description and a copy of our data policy. They may only continue with the study if they consent to the policy. We ask site visitors not to participate if they do not meet the requirements of (1) normal or corrected to normal vision, (2) no motor impairment, and (3) a computer in a desktop configuration with a pixel-accurate pointing device. In the second step, the participants adjust the scale of the study interface to ensure comparably sized scatter plots across all displays.

6.6 Results

In this section, we evaluate various aspects of the collected data according to the hypotheses from Section 6.4. We perform various statistical tests and report the results. Shapiro-Wilk tests showed that the recorded study data is mostly not normally distributed. Therefore, we mainly use Wilcoxon rank sum tests, as already assumed in the power analysis in Section 6.5.6. Because we inspect the data with regards to different aspects for each hypothesis, we use and report Bonferroni correction. In the interest of

[2] `https://www.mturk.com/`, accessed 2024-01-27.
[3] `https://www.prolific.com/`, accessed 2024-01-27.

brevity, not all results are included in this section. We refer the interested reader to the separately published supplemental material [Rod+23a].It contains the underlying data, all results of the statistical tests, and plots for analysis.

6.6.1 Exclusion Criterion

In addition to the platform-based rejections on Prolific, we ensure high data quality by including trivial checks in point and cluster tasks. In these, the highlighted elements are well separated from the others and have restricted movement during the animation. We exclude participants if less than half their answers to point checks are within a radius of 0.1 on the normalized scatter plot axes (0 to 1). Humans are fallible. Hence, if the regular task results are too accurate, we assume that the study participant tampered with the website or introduced some kind of automation instead of clicking manually. The threshold for exclusion is a mean offset from the actual dot position of less than one pixel over all point tasks.

We remove submissions with trivial cluster task checks that achieve less than 50 % correct responses. Additionally, we reject participants that finished the study in less than the total animation runtime. We replace rejected submissions by recruiting more participants to ensure precisely 170 samples.

6.6.2 Point Task Performance

We measure the distance between the indicated position and the actual dot within the normalized scatter plot axes $[0, 1]$ to determine how well each animation facilitates tracing data points. We split our investigation into hypothesis **HP** into multiple aspects.

First, we analyze each animation against every other, using Bonferroni correction. Shapiro-Wilk normality tests indicate that the distributions for STR, TIM, and ORT are not sufficiently close to normal. Therefore, we used Wilcoxon signed-rank tests to compare individual animations. Table 6.1 shows an overview of the results.

We aggregate the study data by animation type to compare spline-based and rotation-based animations (see Figure 6.10). The task performance with rotation-based animation was better ($Mdn = 0.264$) than with splines ($Mdn = 0.298$). An exact Wilcoxon signed-rank test showed that this difference was statistically significant ($p \ll 0.0001$, $W = 12,536$). The rank biserial correlation coefficient $rc = 0.725$ shows a large effect size.

Table 6.1: Comparison of point task performance with pairs of exact Wilcoxon signed-rank tests. Abbreviations according to Section 6.3.2. We used Bonferroni correction for the significance threshold $\alpha = 0.05/15$ and encoded the result into color: significant* and not significant.

(a) p-value

	BUN	TIM	STA	PER	ORT
STR	0.0581	$< 0.0001^*$	$\ll 0.0001^*$	0.1376	$\ll 0.0001^*$
BUN		0.0152	$\ll 0.0001^*$	0.0035	$\ll 0.0001^*$
TIM			$\ll 0.0001^*$	$\ll 0.0001^*$	$\ll 0.0001^*$
STA				0.0001^*	0.1917
PER					$\ll 0.0001^*$

(b) Difference in estimated means

	BUN	TIM	STA	PER	ORT
STR	−0.0106	−0.0241	+0.0276	+0.0073	+0.0352
BUN	+0.0106	−0.0156	+0.0384	+0.0158	+0.0433
TIM	+0.0241	+0.0156	+0.0521	+0.0292	+0.0600
STA	−0.0276	−0.0384	−0.0521	−0.0202	+0.0081
PER	−0.0073	−0.0158	−0.0292	+0.0202	+0.0281

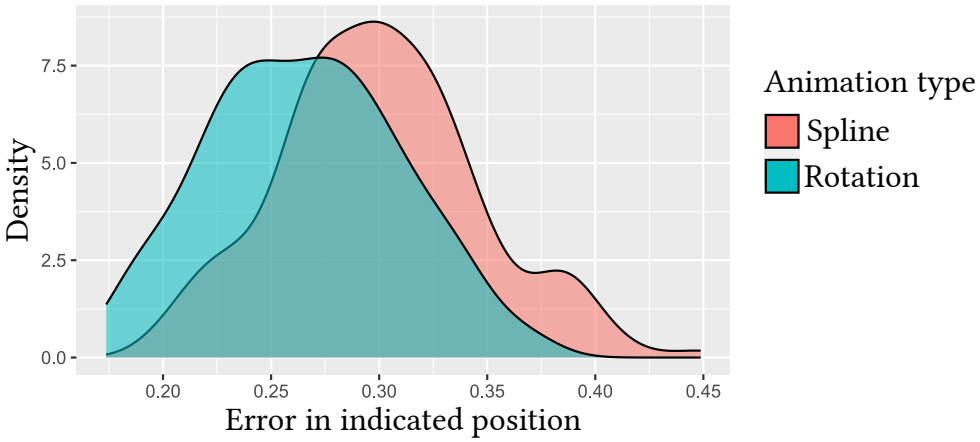


Figure 6.10: Distribution of errors in indicated dot position by animation type. Less error indicates better traceability of data points.

Table 6.2: Comparison of cluster task performance with pairs of Wilcoxon signed-rank test with continuity correction. We used Bonferroni correction for the significance threshold α =0.05/15. All results are not significant.

	BUN	TIM	STA	PER	ORT
STR	0.2765	0.9990	0.7565	0.3056	0.4724
BUN		0.3183	0.2919	0.9042	0.7762
TIM			0.9103	0.2000	0.5337
STA				0.1571	0.6668
PER					0.4666

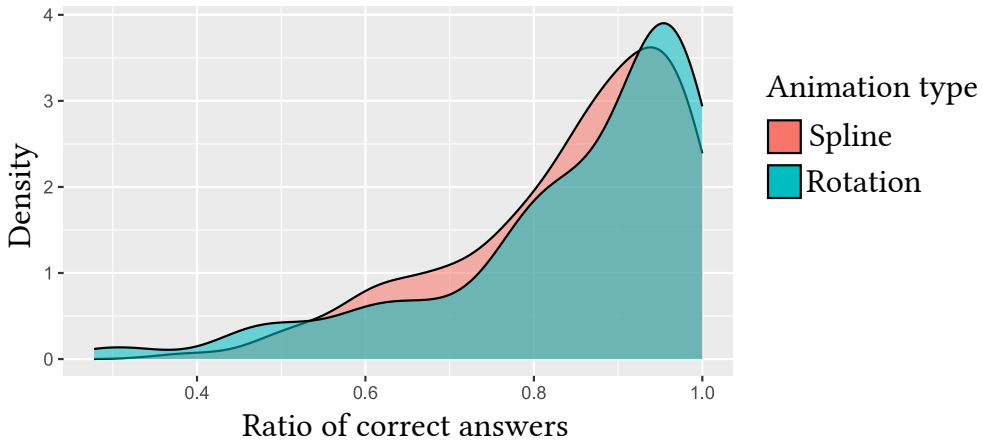


Figure 6.11: Distribution of correctly identified cluster interactions. A higher ratio of correct answers indicates better traceability of clusters.

6.6.3 Cluster Task Performance

We use the percentage of correctly identified cluster interactions to compare the animations concerning the traceability of clusters. The overall mean task performance was 84.5 %. First, we perform a pairwise analysis of the task performance by animation. Wilcoxon signed-rank tests with continuity correction (there were ties) did not show significant results (see Table 6.2).

In addition to analyzing individual pairs, we also compare the animations grouped by type (see Figure 6.11). The task performance with rotation-based animation was slightly better ($Mdn = 91.6\,\%$) than with splines ($Mdn = 87.5\,\%$). A Wilcoxon signed-rank test with continuity correction did not provide evidence for statistical significance ($p = 0.2574$, $W = 4,328.5$, $z = -1.12$, $r = -0.086$). The mean task performance of rotation ($Mn = 0.847$, $SD = 0.158$) and spline ($Mn = 0.843$, $SD = 0.134$) animations was similar.

6.6.4 Animation Direction

We assumed that the animation direction would not affect task performance. However, we still used the same amount of horizontal and vertical 1D transitions to analyze possible effects.

For 1D transitions, the performance of the point task was lower with animation in the horizontal direction ($Mdn = 0.3994$) than in the vertical direction ($Mdn = 0.2682$). A Wilcoxon signed-rank exact test showed that this difference was statistically significant ($p \ll 0.0001$, $W = 14,510$, $rc = 0.997$). The pairwise tests between all animations and with Bonferroni correction were also statistically significant (see Table 6.3).

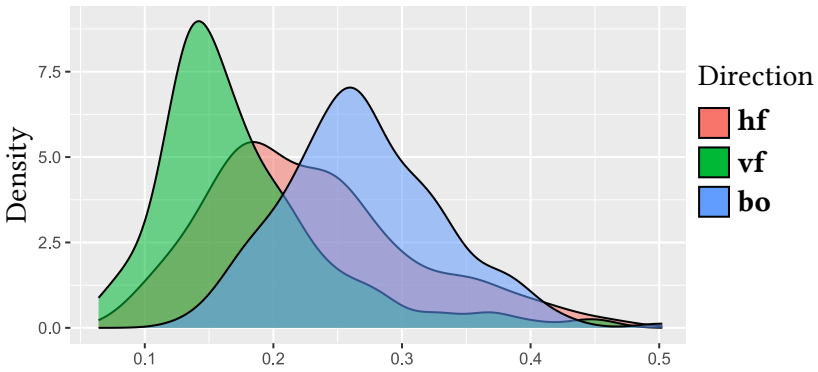
There was no significant difference in the overall performance for the cluster task with 1D transitions. However, for BUN splines, the horizontal performance was better than with vertical transitions (pseudo $Mdn = +0.3333$). A Wilcoxon signed-rank test with continuity correction showed that this difference was statistically significant ($p = 0.0047$, $W = 1,044$). The rank biserial correlation coefficient indicated a medium effect size ($rc = 0.471$). For TIM, the horizontal performance was, once more, worse than in the vertical direction (pseudo $Mdn = -0.3333$). A Wilcoxon signed-rank test with continuity correction showed that this difference was statistically significant ($p \ll 0.0001$, $W = 641$). The rank biserial correlation coefficient indicated a large effect size ($rc = -0.585$).

Directions in 2D transitions differ from the ones available in one dimension. Rotation-based animations can run *horizontal first* (**hf**), then vertical, as well as *vertical first* (**vf**), then horizontal. Spline-based animations always use *both* (**bo**) directions simultaneously. For the point task, Figure 6.12a shows large differences in performance. The error of the point task with 2D transitions was greater in **hf** direction ($Mdn = 0.2190$) than in **vf** ($Mdn = 0.1576$). A Wilcoxon signed-rank exact test showed that this difference was statistically significant ($p \ll 0.0001$, $W = 12,380$, $rc = 0.703$). The error of the point task with 2D transitions was less in **hf** direction ($Mdn = 0.2190$) than in **bo** ($Mdn = 0.2646$). A Wilcoxon signed-rank exact test showed that this difference was statistically significant ($p \ll 0.0001$, $W = 3,089$, $rc = -0.575$). There were no significant differences for the cluster task.

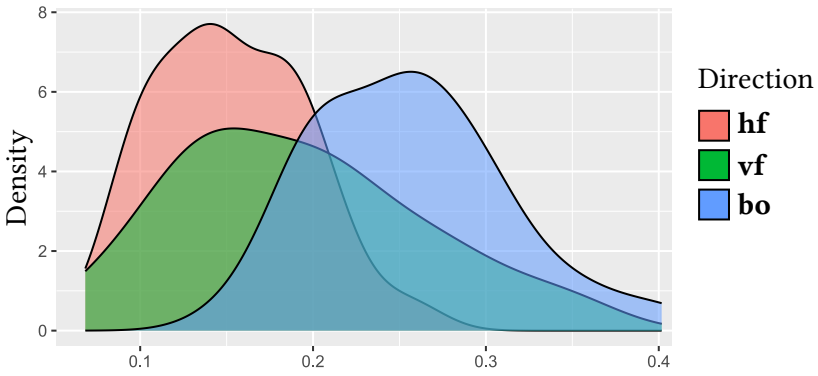
We were surprised by the strong effects of the direction of transitions. They might be linked to the point distributions in the underlying data dimensions. The random selection (see Section 6.5.2) could lead to the uneven use of dimensions with few unique values. Therefore, we repeated the point task with 30 additional participants in a smaller follow-up study. In this new study, we reused the previous selection of data dimensions but switched their roles as x-axis and y-axis for the plots. The change resulted

Table 6.3: Results of Wilcoxon signed-rank exact tests for point tasks with horizontal vs. vertical 1D transitions. We used Bonferroni correction for the significance threshold $\alpha = 0.05/6$. All differences are significant.

Animation	p	W	rc	Δ pseudo Mdn
STR	$\ll 0.0001$	$14,018$	0.929	0.1703
BUN	$\ll 0.0001$	$13,356$	0.838	0.1182
TIM	$\ll 0.0001$	$14,281$	0.965	0.1582
STA	$\ll 0.0001$	$12,595$	0.733	0.1115
PER	$\ll 0.0001$	$14,167$	0.949	0.1551
ORT	$\ll 0.0001$	$12,595$	0.733	0.1244



(a) Error in indicated position (main study)



(b) Error in indicated position (axes switched)

Figure 6.12: Distribution of point task performance with 2D transitions for each direction. Smaller error values are better. The upper diagram shows the data from the main study with 170 participants. We recorded the data for the lower diagram in a smaller study with switched axes and 30 participants.

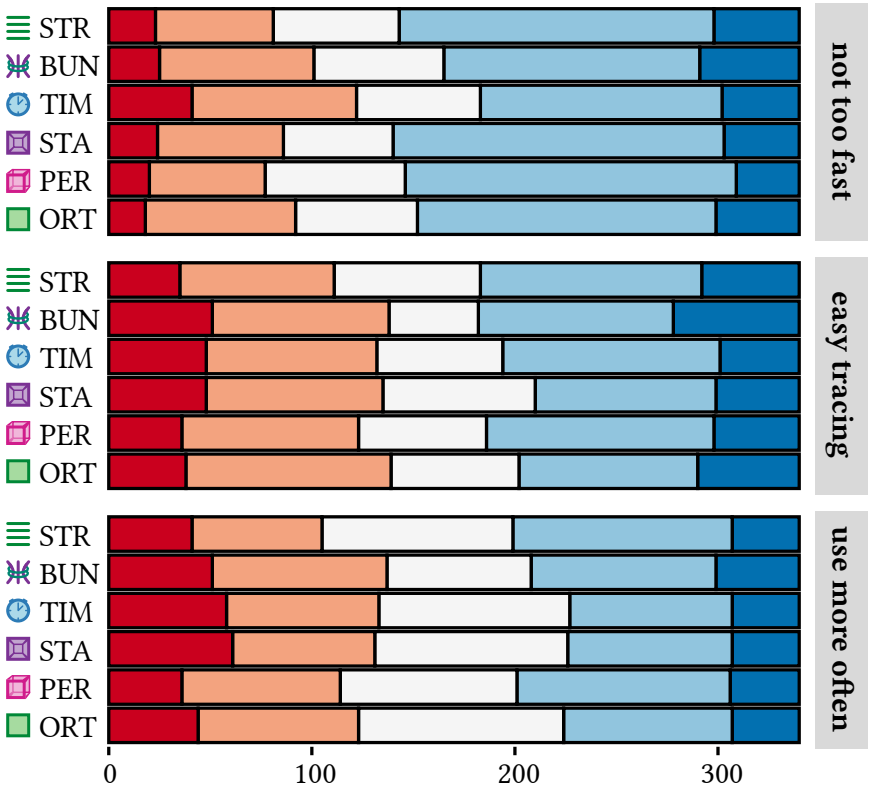


Figure 6.13: Subjective feedback from study participants to the statements from Section 6.5.2. Responses are given on a Likert scale: strongly disagree, disagree, neutral, agree, strongly agree.

in an automatic mirroring of animation directions.

In the follow-up study, the errors in the 1D point task were lower with horizontal ($Mdn = 0.2834$) than with vertical ($Mdn = 0.3738$) transitions. A Wilcoxon signed-rank exact test showed that the difference was statistically significant ($p \ll 0.0001$, $W = 0$, $rc = -1$). For 2D point tasks (see Figure 6.12b), the error in **hf** direction ($Mdn = 0.1458$) was lower than in **vf** ($Mdn = 0.1875$). A Wilcoxon signed-rank exact test confirmed that the difference was statistically significant ($p = 0.01367$, $W = 114$, $rc = -0.510$). Again, animation in **vf** direction ($Mdn = 0.1875$) was less error-prone than in **bo** direction ($Mdn = 0.2529$). A Wilcoxon signed-rank exact test confirmed that the difference was statistically significant ($p = 0.0003$, $W = 67$, $rc = -0.712$). Therefore, the simultaneous animation in both directions performed worst in the main study and the follow-up.

6.6.5 Subjective Rating

To compare the animations with regard to subjective feedback from participants, we asked for the study participants' agreement with specific

Table 6.4: P-values from χ^2 tests of independence for the pairwise comparison of subjective ease of the point task. We used Bonferroni correction for the significance threshold $\alpha = 0.05/15$ and encoded the result as color: significant* and not significant.

	BUN	TIM	STA	PER	ORT
STR	0.0019*	0.0306	0.1232	0.4832	0.1560
BUN		0.3103	0.1581	0.1607	0.1653
TIM			0.3092	0.3383	0.3455
STA				0.5006	0.3010
PER					0.9408

statements using a Likert scale (-2 to 2, see Section 6.5.2). Figure 6.13 shows the combined results. Since we are working with ordinal data, we employ χ^2 tests of independence. However, to calculate mean values, we use the numeric representation of the Likert scale.

We asked whether they found it easy to trace the points and clusters. Overall, the participants responded neutrally ($Mn = 0.064$, $SD = 1.265$). Transitions were reported to be easier for the point task with STR ($Mn = -0.2$, $SD = 1.224$) than with BUN ($Mn = -0.665$, $SD = 1.221$). A χ^2 test of independence showed that there is a significant relationship between the variables of animation (STR vs. BUN) and participant rating ($\chi^2(4, 170) = 17.085$, $p = 0.0019$). This is the only significant difference between the rated ease of tracing points or clusters with different animations, as listed in Table 6.4.

After each point and cluster task, we asked the participants whether they would like to use the animation more frequently. Overall, the responses were neutral ($Mn = -0.035$, $SD = 1.210$). Participants reported preferring animations with STR ($Mn = -0.171$, $SD = 1.167$) over BUN ($Mn = -0.653$, $SD = 1.078$) for the point task. A χ^2 test of independence showed that there is a significant relationship between the variables of animation (STR vs. BUN) and participant rating ($\chi^2(4, 170) = 18.052$, $p = 0.0012$). As listed in Table 6.5, this is the only significant difference between the preferences for animations for tracing points or clusters.

In Section 6.5.4, we reported on a pilot study that we used to find an animation speed toward which participants were neutral. The main study also asks to rate the agreement with “the animation was not too fast” after each point and cluster task. Overall, the responses were neutral ($Mn = 0.313$, $SD = 1.137$). Participants reported with statistical significance that animations with TIM were generally too fast for the point task (see Table 6.6). There were no significant differences for the cluster task.

Table 6.5: Comparison of reported preference for frequent use of animations in the point tracing task. The table contains the p-values from pairwise χ^2 tests of independence. We used Bonferroni correction for the significance threshold $\alpha = 0.05/15$ and encoded the result as color: significant* and not significant.

	BUN	TIM	STA	PER	ORT
STR	0.0012^*	0.0148	0.3291	0.5613	0.1227
BUN		0.2732	0.0114	0.0467	0.0998
TIM			0.4001	0.0246	0.2496
STA				0.0862	0.1372
ORT					0.1926

Table 6.6: P-values from χ^2 tests of independence regarding the pairwise comparison of whether the participants found the animations not to be too fast for tracing points. We used Bonferroni correction for the significance threshold $\alpha = 0.05/15$ and encoded the result as color: significant* and not significant.

	BUN	TIM	STA	PER	ORT
STR	0.0571	$\ll 0.0001^*$	0.9219	0.9329	0.6221
BUN		0.0061	0.0185	0.0184	0.2751
TIM			$\ll 0.0001^*$	$\ll 0.0001^*$	$< 0.0001^*$
STA				0.5416	0.3441
PER					0.4151

6.7 Discussion

With the results from Section 6.6, we can now revisit the hypotheses from Section 6.4.

Significant Differences in Point Task Performance

Hypothesis **HP** about point task performance is confirmed. It holds for individual pairs of animations but also when we aggregate spline-based and rotation-based transitions. We use the differences in (pseudo)median errors to create a ranking of animations: **1.** ORT, STA; **2.** PER, STR, BUN; **3.** TIM. Ranks are shared when there is no statistical significance between items. However, we still list the animations within each rank according to their pairwise difference in (pseudo)medians. The aggregated results also indicate that the transition type affects the error rate: **1.** rotations; **2.** splines. We are surprised that the more complex rotations perform better than the simple straight-line animations—even in 1D. It might be interesting to investigate possible causes in further studies.

Bonferroni correction is very conservative with regard to type I errors and quickly decreases the threshold for statistical significance. Without this correction, the difference in task performance between BUN vs. PER and TIM in Table 6.1a would have been marked as significant. The resulting hypothetical ranking would have been **1.** ORT, STA; **2.** PER, STR; **3.** BUN; **4.** TIM. However, without further dedicated studies, we cannot assume these hypothetical results to be significant.

No Significant Differences in Cluster Task Performance

There is no evidence to confirm hypothesis **HC** about performance differences in the cluster task. We found no significant results between individual pairs of animations. Even when we aggregated the data by transition type, the results were significant, but the effect size was below small levels ($|r| = 0.086 < 0.1$). In general, participants identified cluster interactions well (84.5 % correct). Therefore, scatter plots with animated transitions might provide an intuitive alternative to more complex visualization techniques for cluster interactions. We suspect that, with the same animation speed, clusters are easier to trace than points because the groups of dots might merge to a single perceptual entity during movement. It might be necessary to adjust the timing to find significant differences. In practice, however, it is improbable that the transition speeds between views of a SPLOM vary case-by-case, depending on the user's task. The analyst would have to change settings manually before each animation, or the visual analytics software would need to detect the user's task and intent autonomously. The latter is an active field of research [Gad+21; HC13], but, at the time of writing, we are not aware of robust solutions that would not lead to user irritation due to wrongly deduced animation speeds. Therefore, we argue that an ecologically valid comparison of individual animations and transition types requires the same speed for all tasks.

Possibly Significant Differences for Animation Direction

Following the results from the main study, we have to reject hypothesis **HD**. Nevertheless, we suspect that the differences in user performance between animation directions are primarily due to a combination of the data set and a "randomly bad" selection of the underlying data dimensions for display in the scatter plots. A small follow-up study with switched x-axis and y-axis shows an inversion of the results regarding the differences between horizontal and vertical directions in both 1D and 2D transitions. The effect sizes rc from both studies are roughly the same in the case of 1D view changes (0.997≈1) and both large for two dimensions (0.703 and 0.51). The difference in 2D transitions might be a real effect, or it might be a result of uncertainty due to the small sample size in the follow-up study. The task performance was consistently worst when altering both

axes simultaneously. Further research is required to determine whether this is caused by an increase in cognitive or perceptual load or is due to the generally inferior results of spline-based transitions for the point task. The primary goal of the main study is to evaluate the various animations, not directions. Please note that the analysis of the presented animations is not affected by possible horizontal or vertical effects because we used the same combinations of start and end views for all transitions.

Participants Prefer Straight Animation Paths

For hypothesis **HR**, we asked participants to state their agreement with the given statements. Regarding individual animations, there were mostly no significant differences between the agreement of participants with "the paths of individual points / clusters are easy to follow" and "I would like to use this animation frequently." We can confirm **HR** concerning the direct comparison between ≣ STR and ⋇ BUN lines (preference toward the former) for animation in the point task. The results for both statements—ease and preference—yield a ranking where ⋇ BUN is probably last.

Hypothetically, without the restrictive Bonferroni correction, 🕐 TIM would share the last rank with ⋇ BUN. We can confirm **HR** in connection with a reportedly short playtime for the 🕐 TIM animation (last rank). Without Bonferroni correction, we would have a second-to-last rank for ⋇ BUN. We cannot accept these hypothetical significance values with certainty. However, they give an indication for evaluations of interest for follow-up studies.

Worse Feedback for Animations That Highlight Clusters

Both animation techniques that we designed to highlight clusters were ranked worse. We suspect this might be due to potentially faster dot movements with 🕐 TIM and more overdraw from higher proximity with ⋇ BUN.

Note that the objectively measured error in point tracing shows significant differences between animations, but the subjective rating does not. We suspect that humans might not estimate well how good they are at visually tracking an object when there are multiple distractors. We did not provide feedback to the participant about the positional error to avoid learning effects—remember that we used the same marked points and data dimensions with all animations for better comparability. We speculate that there might be a connection to the results of the study by Robertson et al. [Rob+08], where participants liked interacting with animations independently of performance. Further analyses, however, are outside of the scope of this chapter.

The overall results from the main study confirm the findings of the pilot study: we choose the animation time so that, on average, the participants neither found the dot movements too quick nor too slow. This resulted in

a base animation time of 1 second, which was sufficient for participants to effectively track individual points and cluster interactions alike. We recommend this duration as a sensible default and encourage the use of animation in visual analytics software that provides ways of switching between different scatter plots of the same data set, e.g., via SPLOMs. To accommodate a wider range of users and their preferences, animation should not be mandated.

Recommendation for Orthographic Rotation

In real-world applications behind ecological validity, users are limited by hard-to-solve tasks. Since participants solved the cluster task mostly correctly with all animations, the point task is decisive for the evaluation of the animated transitions. We recommend the use of orthographic rotation based on the ranking from the performance of tracing points. Staged rotation, implemented to resemble the animation from related work [EDF08], shares the first rank but requires a longer playtime due to the three sequential stages. Our recommendation is in accordance with the subjective user preference, which was only directed against BUN and TIM lines.

6.8 Limitations

After conducting the study and analyzing the results, we noticed limitations of the study design.

We regard a common animation speed for the point tracing and the cluster interaction task as being realistic for currently available visual analytics software. For research purposes, however, it could have been more interesting to investigate independent animation times in order to get more diverse results for the simpler cluster task. We suspect that a speed setting that yields a ratio of correct answers around 50 % could be better suited to find significant results when comparing different animation techniques for cluster interactions.

The results from the main study indicate a large effect from the animation direction. In fact, it seems much larger than related work would suggest. Data from the follow-up study did provide evidence against such a large effect but was not able to confirm or reject our initial hypothesis **HD**. Our primary goal was not the comparison of directions but of the animation techniques. This is reflected in the study design. To definitively answer **HD**, it would be necessary to record samples for both the original combinations of data dimension as well as the x-y-switched variants. We opted against such a design to get higher-quality data for the comparison of animation techniques. We argue that it would have been counter-productive to pursue

more goals because the study duration was already 50 minutes. A study time of 100 minutes (double each task) could present a higher threshold for recruiting crowdworkers and might lead to high degrees of fatigue in participants.

While the presented scatter plots were small, we allowed data points to appear outside the plot bound during the animation. In practice, not all use cases provide unused space around the scatter plot, e.g., on mobile devices. When used with SPLOMs, however, points could temporarily move in front of the matrix.

6.9 Verdict

We designed and conducted a crowdsourced user study with 170 participants to evaluate the different transitions. The results show that rotation-based animation outperforms its spline-based counterpart. ◻ Orthographic and ◻ staged rotation worked best to allow participants to trace individual data points across transitions. Note, however, that staging requires more runtime. Cluster-centric approaches of spline-based animation did not work well for the point task. All transitions resulted in good and similar user performance for the cluster interaction task.

With respect to RQ2, we encourage using animated transitions between different views of the same underlying data set. With only a small impact on time, they provide effective ways to trace individual points and clusters. As a result, analysts can make more effective use of point-based visualization in scatter plots and SPLOMs without using additional visual attributes for mappings. Animation seems to be an intuitive approach because participants were able to follow the target elements across changing data dimensions without requiring lengthy training or extensive experience in the field. The presented transitions are part of the way toward solving RQ2. Alone, however, they do not provide sufficient support for navigating the large space of dimension pairs from multivariate data. To further help users of scatter plots and SPLOMs, we need to help them with the choice of data views.

While we were able to evaluate the animation techniques, we also encountered further questions. Future research could help to better identify and quantify effects from the direction of transitions. We required participants of our study to use desktop-like computer setups. Due to the wide spread of portable devices, it would be interesting to reevaluate task performance with animations on smaller and moving displays. Further research could explore transitions that account for more than two scatter

plots. Our framework supports such animations through intermediate views and control points for splines.

7

Gaze-Based Recommendations for the Exploration of Scatter Plot Matrices

This chapter is based on previous work and will use extracts thereof without explicit quotation:
N. Rodrigues, L. Shao, J. J. Yan, T. Schreck, and D. Weiskopf. “Eye gaze on scatterplot: Concept and first results of recommendations for exploration of SPLOMs using implicit data selection.” In: *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*. ACM, 2022, 59:1–59:7. DOI: 10.1145/3517031.3531165.

7.1 Motivation

Within the frame of  RQ2, we want to explore ways to help the user navigate 2D plots of multivariate data. In the previous Chapter 6, we were able to evaluate animation techniques that are useful for switching between views while tracing individual data elements. In this chapter, we want to help users identify the different plots that might be interesting for further analysis. While scatter plots and SPLOMs are well-known techniques for visual analytics, it is often not clear which dimensions to select for detailed inspection in order to solve a specific task.

There are computational methods to reduce the dimensionality of data back to the 2D plane and avoid the issue of having to choose between specific attribute pairs. They include techniques like principal component analysis (PCA) [Jol02], uniform manifold approximation and projection (UMAP) [MHM18], and many more. As already mentioned, these methods often lead to non-intuitive projections that are hard to trace back to the

original data attributes. However, their results might still be perfectly suitable for specific tasks, like the separation of data into clusters. We want to pursue an approach that can fit more tasks.

There are multiple quality metrics that can help find interesting patterns in the data. Most are numerical measures for objective assessment of how well a visualization is fit to solve a specific task. They can tell if a plot contains outliers or well-separable clusters but do not take user interest into account. Therefore, such metrics are limited to a set of predetermined tasks that the scientists envisioned during their development. In our work to answer RQ2, we want the proposed technique to provide a tailored solution instead of resorting to a purely off-the-shelf repertoire.

In Chapter 6, we gave users a task and noted the possibility of specific eye movement called "smooth pursuit." We expected a connection between eye movement and user task. This eye-mind hypothesis [JC80] might also work the other way around. Now, we assume that by observing the gaze position we can infer the data of interest. Such an approach is truly generic and does not limit the user task to a predetermined set.

The idea of recording the user's gaze is not new in the visualization community. Hardware is available, relatively cheap, and can be retrofitted to existing systems. Blascheck et al. [Bla+17] show a wave of more than one hundred publications about visualization for eye-tracking data between 1998 and 2016. While they divide these publications into three categories, the topic is always about using visualization to analyze the gaze data. More recent work advocates the use of eye tracking as an additional source of information for use in visual analytics systems [Sil+19], which is in line with what we want to develop in this chapter.

We want to use gaze data to give viewers of SPLOMs recommendations for scatter plots that suit their task and intent without explicit knowledge thereof. Only then is our technique truly flexible and adaptive to the user. Recommendations for pairs of dimensions provide the viewer with help in navigating the potentially vast space of D^2 scatter plots (RQ2). While we are aware of multiple recommender systems that incorporate the use of gaze data [Zha+16; RP17; Sha+17; Sil+18], none of them work in the same way as our concept: with implicit user interest on the level of individual data points.

7.2 Concept

We next describe our 3-step concept to apply real-time eye tracking to facilitate the discovery of various patterns in data of interest and support analysts in exploring multivariate data sets as part of our work on RQ2.

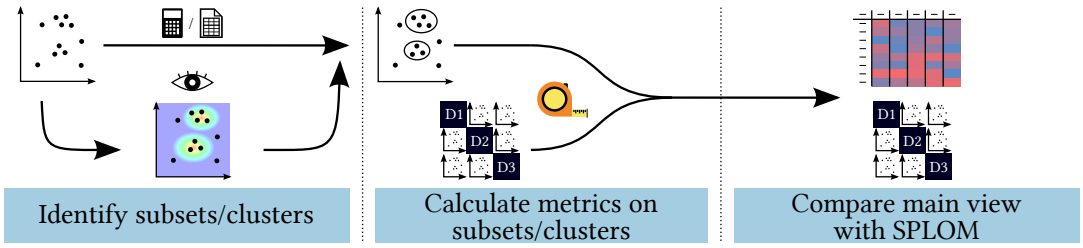


Figure 7.1: The three steps in our concept. First, we identify subsets or clusters of data (based on the preexisting classification in the source data, clustering algorithms, or visually focused points). Second, we calculate various metrics with the clusters in the main scatterplot and all other views of the SPLOM. Third, we compare the resulting values and show differences in a table and on the SPLOM.

To provide fitting recommendations, our approach focuses on specific data groups instead of the entire data set at once. As Figure 7.1 shows, the first step of our method extracts subsets of the data for further computations. These subsets can be provided as an existing classification that is stored in a dedicated dimension of the data set itself. Alternatively, users can apply clustering algorithms. To this end, our proof of concept implements the DBSCAN algorithm [Est+96]. Another option is the implicit selection of data of interest through eye tracking.

7.2.1 Gaze as Indicator for Data of Interest

While analysts observe scatter plots, they may find interesting data points or groups of data, e.g., clusters, outliers, correlations, or other patterns. They may form hypotheses about relationships between these points. However, a single scatterplot only shows one 2D subspace. To further explore the interesting data in other subspaces, the analysts have to switch views, i.e., by selecting a scatterplot from the SPLOM to become the main view (see Figure 7.3). The mental map would be disrupted as every point might appear in a completely different position due to the switch. Brushing and linking across the entire plot matrix would be of help. This requires the analyst to remember what data sparked interest and to explicitly select it. The eye-mind hypothesis [JC80] can help in this case—even though it has to be understood with the right level of caution. Visual analytics software can infer the data of interest without the need for explicit interaction through the use of eye-tracking hardware.

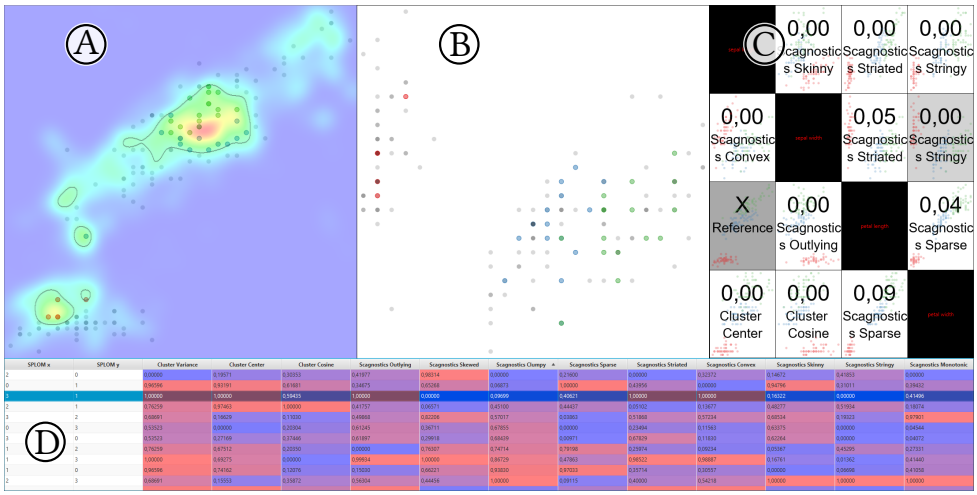


Figure 7.2: Main window of our visual analytics tool used as proof of concept and test bench for visualization recommendations based on eye tracking. We record the user's gaze, shown as a heatmap, and select clusters of data in the *main scatterplot* (A). The *secondary scatterplot* (B) is for further exploration. The axes for (B) are chosen from the *SPLOM* (C) or the *table* (D). Both (C) and (D) display differences between metrics of the clusters selected in (A) and the same data in other pairs of dimensions. The numeric values in the table are mapped to background colors using a continuous scale from blue (low) to red (high).

7.2.2 From Gaze to Clusters

Our method records gaze data 👁 on the main scatterplot and generates a heatmap (as in Figure 7.2a). We model the size of the foveal region and the uncertainty of the eye-tracking accuracy by splatting the gaze positions with a Gaussian kernel. The heatmap allows us to correlate the viewer's interest to the underlying data via display positions.

In addition to the pointwise assignment of interest, we can extract clusters. Typically, data clusters would be calculated directly from the underlying data set by use of a distance metric. However, we propose the use of interest clusters, i.e., data points that appear close to each other in a plot and have been looked at more intensely. To this end, we select a threshold in the heatmap and identify coherent regions with values above an adjustable threshold. A single interest cluster is now defined to contain all data points that are within a coherent region in the heatmap.

7.2.3 Data Metrics

Our goal is to recommend other views from the SPLOM, where the points of interest exhibit the same or different behavior, leading to a useful ex-

ploration sequence. Therefore, the second step of our proposed concept is based on the descriptive properties of data points of interest in 2D subspaces. This solves a typical problem of scatter plots and SPLOMs: The analyst needs to obtain an overall view of the data of interest, but only one subspace is well visible at a time. Once we obtain subsets of the underlying data from the previous step, we can compute various properties for each cluster or group of data points separately.

We extract such properties in both the main scatterplot as a reference and in all other plots from the SPLOM. We include established statistical measures that can describe scatterplot patterns meaningfully and allow us to compute a distance function between patterns. Specifically, we choose variance and centroids as intuitive and well-known descriptions. Scatter plots can reveal correlations between axes, visible as diagonally oriented distributions. The Pearson correlation coefficient provides a numeric representation. However, we want to calculate the orientation of the data groups. It is more generic and allows for other insights beyond possible correlation. To this end, we perform PCA [Jol02] and determine the orientation of the primary axis in the data cluster with regard to the display dimensions.

For the detection of patterns in the scatter plots, we also rely on the well-known scagnostics features [TT85]. These are graph-based measures describing the visual appearance of a set of selected points. Our proof of concept uses an implementation by Wilkinson et al. [WAG06] to provide these measures: outlying, skewed, clumpy, sparse, striated, convex, skinny, stringy, and monotonic.

The choice of metrics for our concept is meant for initial testing in a proof of concept. Many other data properties are compatible with the approach and could even reflect the viewer's perception. For example, we could use the correlation coefficient with a correction according to Rensink and Baldridge [RB10] or shift the centroids to account for data points outside the clusters Lu et al. [Lu+19].

As the previous step of our approach can yield multiple clusters, we apply the metrics to each cluster individually and sum the resulting values for each plot. Therefore, even if there are no chosen subsets, we can calculate the same generic measures on the entire data set and still use them to rank and compare without restriction.

7.2.4 (Dis)similarity for Recommendations

The third step of our concept uses previous results to give users recommendations. We can use the data of interest to give more fitting suggestions than using the entire source data unspecifically. It allows a targeted use

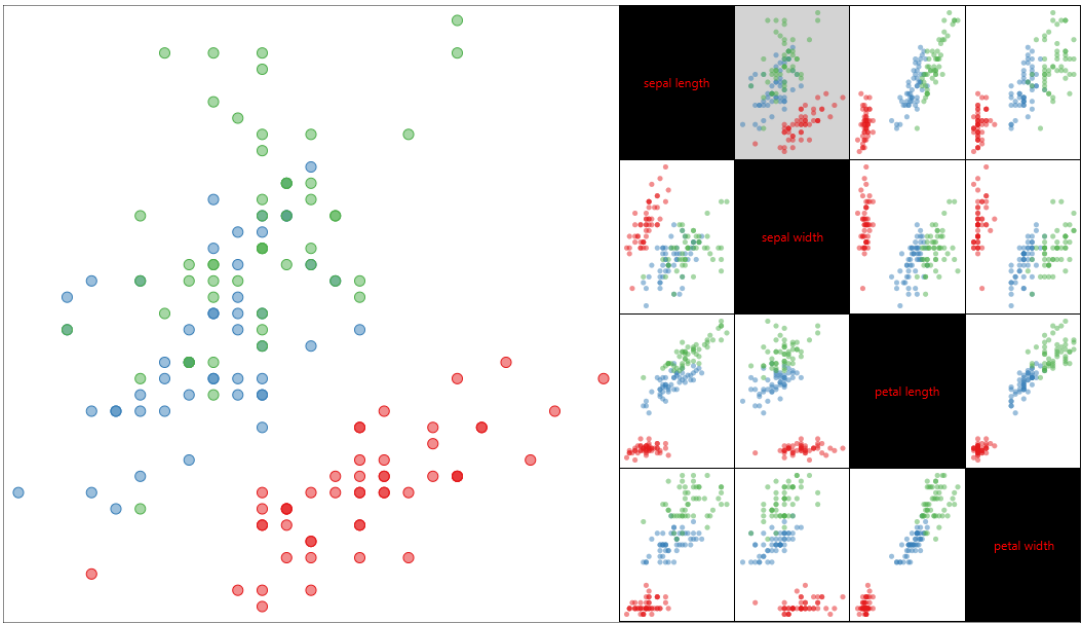


Figure 7.3: Initial view of our tool after loading a data set (iris) and selecting a scatterplot (gray) in the SPLOM. Dots are colored by the prelabeled type of flower.

of any metric to fit the user's current task without the need to actually identify the task itself.

Instead of only suggesting previously unseen combinations of dimensions, we can search for plots where the same data points exhibit similar or different behavior. While there are direct measures of distance or similarity, we want to highlight changes in specific data aspects. That is why we compute the previously mentioned data properties and use them to compare between plots: the difference between the reference plot and a view in the SPLOM results in the displayed (dis)similarity. This approach is independent of a specific similarity metric. We only require that the result is translated to a real value between 0 (similar) and 1 (dissimilar). The normalization of the distance range ensures comparability between various metrics.

We present the resulting values in a table . It is up to the user to choose dimension pairs with a high or low value as recommendations for a thorough exploration of the underlying data set. Additionally, we overlay the lowest similarity in each cell of the SPLOM. If the sort order of the table is chosen to be descending, we display the highest values in the SPLOM.

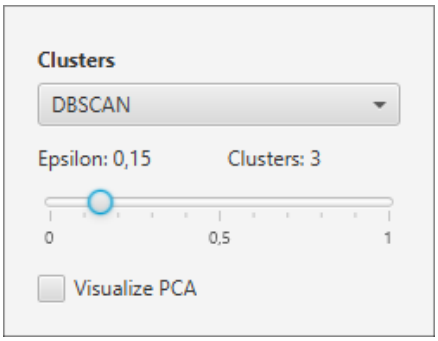


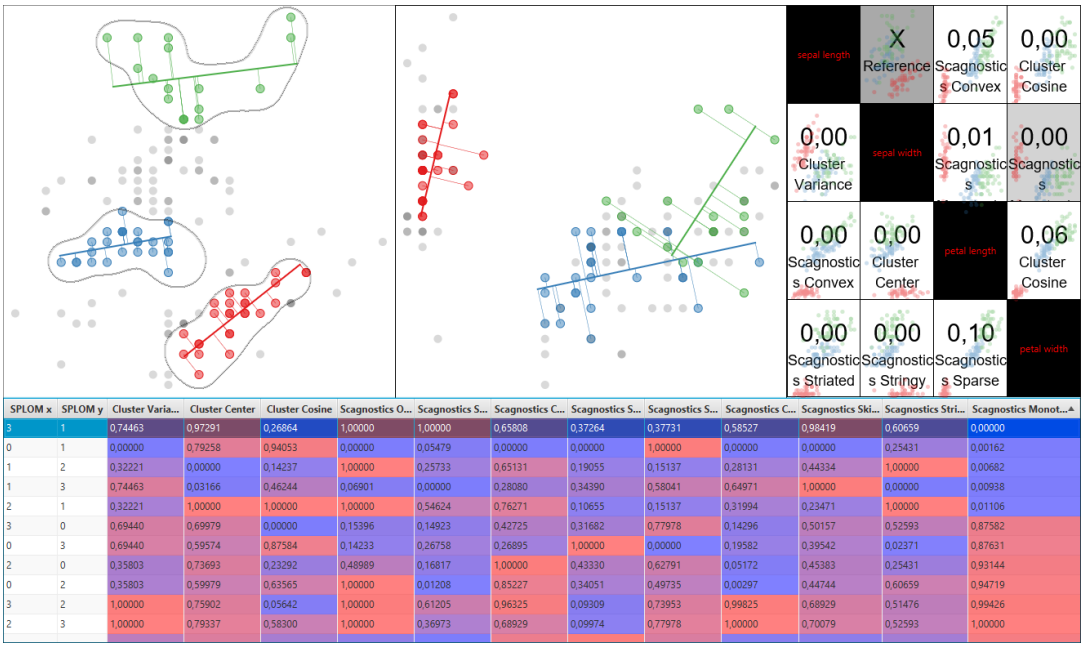
Figure 7.4: Graphical user interface (GUI) with options for subset generation. The current settings calculate clusters with DBSCAN. In this case, the user can adjust the algorithm's parameter ϵ with a slider.

7.3 Software System

We implemented a software system as a proof of concept. After loading a data set, the analyst can browse the SPLOM and choose a pair of dimensions for display in the main scatterplot, as shown in Figure 7.3. A click on another cell of the SPLOM switches the main plot to the corresponding data dimensions. Without the need for eye-tracking hardware, the user can select to color the dots of the scatterplot by clusters—either from data density (DBSCAN) or from a data column containing a precomputed classification—using the drop-down list in Figure 7.4.

Once a clustering method has been selected, the analyst enables the comparative view. This presents a table with the results of the various similarity metrics and extends the layout to accommodate a large secondary scatterplot. Each cell in the SPLOM shows the name and value of the highest metric for the corresponding plot. Selecting another pair of dimensions in the table or the SPLOM displays the corresponding visualization in the secondary scatterplot. To help the user understand the similarity of angular orientations, we provide an optional display of the main PCA axis of each cluster, as shown in Figure 7.5.

Once the user enables eye tracking support, our tool starts collecting gaze data and accumulating it in the heat map on the reference plot (similar to Figure 7.2a). We implemented support for hardware by *The Eye Tribe*. The user can pause and resume the recording at any time through a menu item. Once some gaze data is available, it is possible to choose the heatmap to extract subsets of the underlying data and update the metrics. Both the width of the kernel and the threshold for interest clusters are user-adjustable (see Section 7.2.2 and Figure 7.6). We support the display of the heatmap as an overlay on the main scatterplot, with the possibility to show the threshold as an isoline.



SPLOM x	SPLOM y	Cluster Varia...	Cluster Center	Cluster Cosine	Scagnostics O...	Scagnostics S...	Scagnostics C...	Scagnostics S...	Scagnostics S...	Scagnostics C...	Scagnostics Ski...	Scagnostics Stri...	Scagnostics Monot...
3	1	0,74463	0,97291	0,26864	1,00000	1,00000	0,65808	0,37264	0,37731	0,58527	0,98419	0,60659	0,00000
0	1	0,00000	0,79258	0,94053	0,00000	0,05479	0,00000	0,00000	1,00000	0,00000	0,00000	0,25431	0,00162
1	2	0,32221	0,00000	0,14237	1,00000	0,25733	0,65131	0,19055	0,15137	0,28131	0,44334	1,00000	0,00682
1	3	0,74463	0,03166	0,46244	0,06901	0,00000	0,28080	0,34390	0,58041	0,64971	1,00000	0,00000	0,00938
2	1	0,32221	1,00000	1,00000	1,00000	0,54624	0,76271	0,10655	0,15137	0,31994	0,23471	1,00000	0,01106
3	0	0,69440	0,69979	0,00000	0,15396	0,14923	0,42725	0,31682	0,77978	0,14296	0,50157	0,52593	0,87582
0	3	0,69440	0,59574	0,87584	0,14233	0,26758	0,26895	1,00000	0,00000	0,19582	0,39542	0,02371	0,87631
2	0	0,35803	0,73693	0,23292	0,48989	0,16817	1,00000	0,43330	0,62791	0,05172	0,45383	0,25431	0,93144
0	2	0,35803	0,59979	0,63565	1,00000	0,01208	0,85227	0,34051	0,49735	0,00297	0,44744	0,60659	0,94719
3	2	1,00000	0,75902	0,05642	1,00000	0,61205	0,96325	0,09309	0,73953	0,99825	0,68929	0,51476	0,99426
2	3	1,00000	0,79337	0,58300	1,00000	0,36973	0,68929	0,09974	0,77978	1,00000	0,70079	0,52593	1,00000

Figure 7.5: Comparative view with table and SPLOM. The main scatterplot shows data in clusters and overlays it with the first PCA axis as a thick central line. The spread of data points along the second axis is highlighted using orthogonal lines.

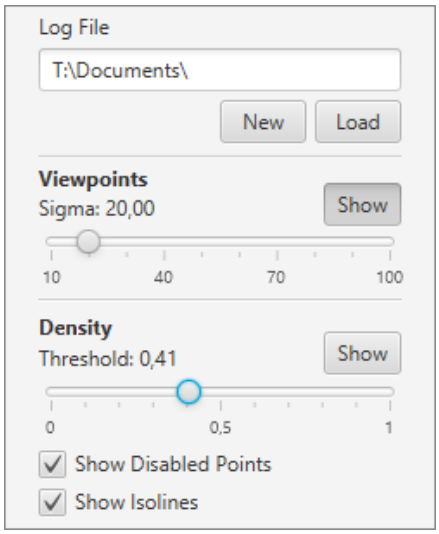
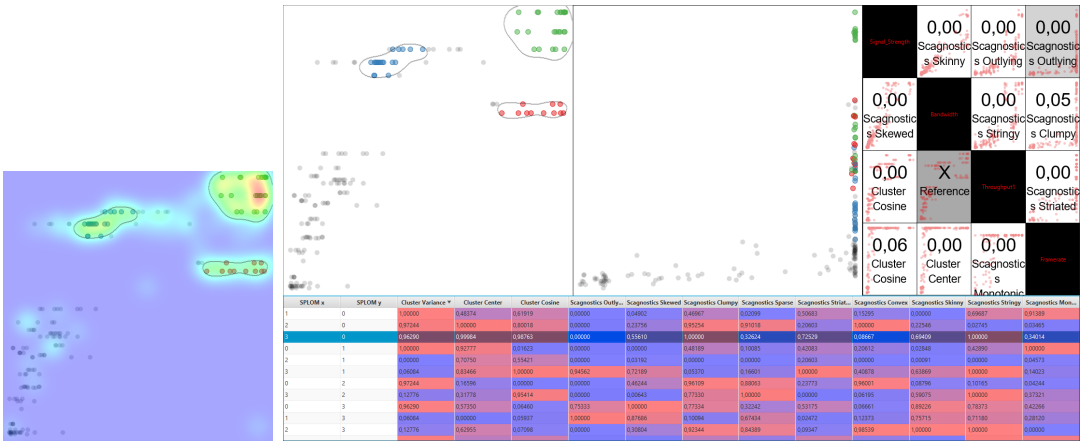


Figure 7.6: Graphical user interface controls for the eye-tracking heatmap and threshold for data subsets and isolines.



(a) Eye-tracking heatmap

(b) Comparative view. Thresholds for gaze data are set, but the heatmap is hidden.

Figure 7.7: Analysis of surprising data in network performance data. The colored clusters in the scatter plots have been selected through eye tracking. Differences in metrics suggest that the plot in the center might show the data points in a different constellation. Checking the recommended pair of dimensions yields the secondary scatterplot in the center. Here, we can see the selected data points in a vertical column and in unexpected order.

Our tool serves as a proof of concept and as a test bench alike. For the latter functionality, it supports storing and loading gaze data as well as emulating an eye-tracker by clicking the main scatterplot with the mouse. To reset the heatmap, the analysts need to start a new recording.

7.4 Use Case

To give an example of how our concept can be used, we analyze a network performance data set that was already used in prior work [Rod+20; Pal+14]. We load the data and see the available data dimensions: signal strength, bandwidth, throughput, and framerate. We browse multiple plots from the SPLOM and stay on the view of bandwidth (x-axis) and throughput (y-axis). As the main scatterplot in Figure 7.7b shows, it contains data along a curved diagonal. We start recording the eye-tracking data and explore this plot further. We notice three data clusters in the region of high bandwidth and high throughput. This seems surprising, and while looking at the points in the plot, we wonder: What is the meaning of these clusters? Why do the points not form a line indicating a positive correlation, as might be expected due to the names of the dimensions?

At this point, we want to investigate these three groups of data points. Therefore, we stop recording further gaze data and enable the comparative view. The heatmap in Figure 7.7a reflects our interest. We enable clustering from gaze, adjust the threshold for data selection to contain the three distinct groups, and disable the display of the heatmap to get an unobstructed view. Since we are interested in dense clusters of data, we order the bottom table descending by *Cluster Variance*. A high value indicates that our data of interest exhibit a dissimilar behavior in other plots. The top five results are all related to the signal strength dimension. We start browsing these and remain on the scatterplot of signal strength by framerate (x-axis by y-axis). Without information about the interest clusters, we only notice that the framerate stays at a high constant value unless the signal strength becomes too weak. The green data cluster appears spread along very high and medium signal strength. The red cluster has medium values, and the blue one occupies the medium-low region. Comparing both scatter plots, we notice that the blue color has higher throughput than the red one. This seems to be an anomaly.

While, initially, we noticed abnormal behavior in one pair of dimensions, our tool enabled us to find an additional dimension that added to the anomalous data. It showed us that low signal strength could appear with higher throughput than some samples with high bandwidth. Without help in finding dissimilar behavior of the data of interest, it would have taken longer to analyze all scatter plots separately and would require a detailed mental map of the data set.

7.5 Discussion

Eye-tracking technology provides an indirect and interaction-efficient means for data selection, logging interaction, and inferring user interest and tasks in principle. Our proposed method works in combination with the previous Chapter 6 to provide an adequate answer to RQ2. It uses gaze on scatter plots for recommender systems working with SPLOM, while the animation helps keep the mental model and context between changing dimension pairs of the main plot.

In the spirit of open and reproducible science, we provide a sample implementation [Yan+22] of our concept (shown in Figure 7.2). It serves as a proof of concept and test bench for a variety of metrics. The metrics serve to compare the entire data set in pairs of dimensions as well as subsets of the data. Using the results of the implicitly selected data of interest and the metrics, analysts can use the tool to get recommendations to explore the data set and find interesting patterns in the data.

The implementation only provides the first steps to help navigate the space of 2D scatter plots in multivariate data. But the underlying design with integration of gaze information is broadly generic and, at the same time, specific to the user task. It presents a promising basis for further development into a full-fledged visual analytics tool. We did not go further because the work behind implementation, testing, and long-term software support goes beyond the core of research toward RQ2 and into the realm of software engineering.

A finished—possibly commercial—product could include the integration of further data modeling approaches, e.g., regression models, outlier detection, subspace search, etc., to cover a broader range of analytical tasks and to find complementary data patterns for confirmation or contradictory patterns for falsification. In the previous chapter about animation, we focused on quantitative evaluation. For the development of a visual analytics tool, a qualitative approach with domain experts might be adequate. Once implemented, the recommender system should be evaluated in a user study with predefined tasks by comparing completion time and error rate. A thorough evaluation should always use two modes: implicit data selection through eye tracking and explicit selection with pointing devices, e.g., a computer mouse.

Further research could investigate additional metrics, prioritization of metrics, and automatic extension of selected data points [Gad+21]. On the one hand, a visual analytics tool could combine our approach with profiling and a history of user interactions to achieve better-fitting recommendations. On the other hand, this might narrow down the range of supported tasks and would require user consent and careful consideration of data privacy [Sil+19]. In the future, even without personal profiles, gaze data might provide useful insights into the intended analytical user task. Similar to intent prediction in the context of assistive devices for patients with reduced capabilities [KN19; KN18], such a development would be in line with the overarching goal of this thesis—improved knowledge gain—by making visual analytics of multivariate data more accessible.

8

Excursion: Eye Tracking for Immersive and Situated Analytics

Our prototype of a recommender system for scatter plot matrices (Chapter 7) showed how information about gaze could be integrated with a visual analytics application. The idea of enhancing visualization systems with eye tracking goes back to a theoretical framework by Silva et al. [Sil+19]. It shows how each part of Keim et al.'s [Kei+08] visual analytics model can be extended to benefit from having gaze data. In our previously discussed case, we only used a 2D eye tracker on a 2D visualization. However, a growing amount of software and hardware are built to support immersive analytics [Mar+18]. In this chapter, we go on an excursion beyond the specific goal of  RQ2 and explore hypothetical possibilities and actual prerequisites for the use of eye tracking with immersive and situated analytics.

This excursion is based on previous work and will use extracts thereof without explicit quotation:
N. Silva, T. Blascheck, R. Jianu, N. Rodrigues, D. Weiskopf, M. Raubal, and T. Schreck. "Eye tracking support for visual analytics systems: Foundations, current applications, and research challenges." In: *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. ACM, 2019, 11:1–11:10. DOI: 10.1145/3314111.3319919, and
S. Öney, N. Rodrigues, M. Becher, T. Ertl, G. Reina, M. Sedlmair, and D. Weiskopf. "Evaluation of gaze depth estimation from eye tracking in augmented reality." In: *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*. ACM, 2020, 49:1–49:5. DOI: 10.1145/3379156.3391835.

8.1 Hypothetical Use Case in Virtual Reality: FiberClay

In this section, we are going to discuss *FiberClay*, a system for immersive visual analysis of works with head-mounted displays and hand-held controllers by Hurter et al. [Hur+19]. We start with a brief introduction of the software and its usage to provide a self-contained reading experience. Then we discuss the possible advantages and challenges of eye tracking integration on the levels of software and hardware.

8.1.1 The Original Unchanged System

FiberClay works with head-mounted displays and hand-held controllers. The authors developed the system with air traffic controllers in mind. Such a group of people has specific domain knowledge of traffic patterns, regulations, and geographic features. However, FiberClay is not restricted to air traffic data. It is a general visual analytics system for trajectories. Users navigate the virtual 3D space by orienting the camera with their head posture and use controllers to move and scale the view. Alternatively, a 2D plane is used for adjusting continuous view and layout parameters. The plane is filled with tiles that show thumbnails to guide analysts in possible adjustments, including the switch to 2D projections of the trajectories, allowing them to follow an airplane's history of flight levels.

Once analysts have chosen their view on the data, they can select large data ranges or specific trajectories using virtual rays that represent hand-held physical controllers. They press buttons and select or deselect all intersecting trajectories, essentially sculpting a query in 2D or 3D. The system gives an intuitive representation of spatial data as it matches the display dimensions. Trajectories of airplanes and neural cell connections can be shown and analyzed. Simultaneously, time series of single parameters can be visualized in 2D, matching regular and well-known techniques from outside the virtual environment. The system runs smoothly on current hardware, and it is well adapted for intuitive interaction through head and hand movement. It can be operated while sitting on a chair and does not require free body movement.

8.1.2 Benefits and Challenges of Eye Tracking Support

Integration of eye tracking in the FiberClay system could lead to several changes in multiple areas. Hurter et al. minimize body movement and potential collisions by providing interaction through hand-held controllers and head movements, allowing an analyst to be stationary. Arm movements could be reduced to pressing buttons by using gaze as an interaction

technique. Visually focusing on virtual objects that are near or far could be suitable for depth selection within virtual reality (VR) headsets. The actual focal distance of drawn objects is physically limited to a static value, so the eyes are always accommodated to a constant depth. But the system could calculate a point where the gaze rays from both eyes intersect and find the data item displayed at that position. With this information, the analytics tool could automatically move the camera to a location that provides the least obstructed view of the focused data. However, frequent automatic view changes could confuse analysts, destroy their mental map, and make the entire system unusable. Therefore, we recommend triggering such a camera move through explicit interaction, e.g., by pressing a button on the controller.

Eyes have a narrow field of view at full resolution. Outside of the retina's fovea, vision becomes blurred. Virtual environments often have high performance demands, and keeping the IT infrastructure of larger organizations up-to-date with the latest developments in computing hardware can be a costly and never-ending task. Using eye movement data, developers can implement dynamic foveated rendering [Gue+12] to prioritize the focused image parts and reduce the graphics resolution in the periphery. This method increases frame rates on older hardware, thus ameliorating issues with simulator sickness [MUT20]. The same concept of foveated rendering could also be extended beyond the rendering resolution. Procedurally generated scenes, for example, could be extended only where the user is looking. Data for visualization could be loaded on demand with priority on the areas of the most attention.

At the same time, gaze reflects the personal interest and could give the system an indication of what data items are most interesting to analysts [JC80]. The 2D parameter navigation, described in the previous section, could highlight the interesting data points and help analysts manually find more suitable representations. Alternatively, the system could automatically select glyphs with less overdraw in the region of interest or suggest completely different views and visualizations that make the target data more readable. Notifications are a possible way of making such suggestions and are often placed in a specific location in regular desktop environments. In an immersive environment, however, there might not be a single suitable static placing option. Notifications might require explicit input to become visible and could be coupled with sound alerts.

People with refraction errors need to wear glasses that might not fit inside head-mounted displays. Additional optics could correct such defects and might be automatically adjusted to the person when combined with eye tracking. To achieve this, the underlying software could show test patterns with increasingly finer shapes and check real-time gaze data for

their visibility.

There are more use cases for eye tracking beyond the post hoc analysis of recorded data in immersive environments. Air traffic controllers—one of the originally intended target users of FiberClay—are subject to a high-workload environment. Eye tracking can reveal the level of cognitive load [ZPR16; Kre+18; Duc+18]. The software could adapt the level of support to keep the controller engaged during slow hours or call for additional personnel to avoid overwhelming a single worker. Fatigue is a known problem in the context of safety in air travel [Cal+09]. Lapses in judgment can lead to accidents with high loss of life and generally affect many people beyond the passengers on board of aircraft. Exceeding levels of fatigue, inferred from eye metrics [BH21], could alert controllers before their cognitive capacity is too low for the tasks at hand.

Overall, gaze data could help in various areas. It could streamline interaction with the data and the analytics tool itself. The combination of the known distance to virtual objects and foveated image parts could steer adaptive optics to replace the use of glasses in head-mounted displays. Inferred data about the cognitive state of the user could help adapt the workload or caution a worker that is potentially unfit for safety-critical tasks.

8.2 Accuracy of Gaze Depth for Augmented Reality

In the previous section, we discussed how gaze information can be used in immersive environments. One example of hypothetical benefits (Section 8.1.2) made use of two gaze rays that intersect to select objects in a 3D virtual space. In this section, we are going to analyze actual measurements of gaze depth with an optical see-through augmented reality (AR) setup.

8.2.1 Motivation

Immersive analytics does not only encompass completely virtual environments. It also spans display setups with a large field of vision and AR. AR can, in principle, substitute every display and also allows visualizations to be situated in close proximity to sources of the underlying data [ElS+15; Tho+18]. Optical see-through head-mounted displays (HMDs) place virtual objects in the real environment of their wearer while avoiding potential drawbacks of video see-through setups: no reduction of the resolution, real objects stay at their actual depth, there is no latency for unchanged visual elements, etc.

Prominent examples of such HMDs are Microsoft's *HoloLens* and *HoloLens 2*. Both can be used for visual analytics with eye tracking support. The second generation comes with integrated eye-tracking hardware for a single gaze ray and provides a depth map of the real environment and virtual objects. Software developers can infer the gaze depth from the intersection of the gaze ray with the object depth. However, this is not suitable for 3D visualization with transparency, such as volume rendering. While the first-generation HoloLens is not equipped with eye-tracking hardware, there is a stereoscopic retrofit kit from PupilLabs that provides 3D gaze positions—including depth.

We already discussed the hypothetical benefits of eye tracking support for immersive visual analytics. There are, however, potential issues for realizing an implementation of the described approaches. Firstly, the gaze rays most often do not intersect. In such a case, we can only calculate a point in space where the rays are closest to each other. Still, the inaccuracy from individual eye measurements leads to much noise in the derived depth and requires filtering [Duc+11]. Attempts to further process the eye-tracking data with a machine-learning approach still yield an average error distance of more than 0.4 m [Lee+17]. Therefore, the depth information is only useful in specific contexts.

Secondly, AR combines real and virtual objects side-by-side. Previous research shows differences in depth perception between real and virtual environments, with the vergence-accommodation conflict (VAC) as a contributing factor [DM96; Duc+14]. HoloLens uses a stereoscopic image to position virtual objects at varying distances from the viewer, but the focal plane remains constant at 2 m. Some experimental designs for HMDs shift the focal plane and avoid the VAC [Joh+16; KCW16], but they are not available as regular consumer-level devices. The conflict of vergence and accommodation is suspected of causing an offset in *subjectively* perceived object depth in virtual scenes [RGA95; Rol+02; SSE15; KCS17].

In this section, we wanted to investigate whether the *objectively* measured gaze depth also varies between real and virtual objects in optical see-through AR. To this end, we designed an AR scene to record gaze data at specific depths.

8.2.2 Experiment

Based on previous research, we assumed that there might also be differences in objectively measured gaze depth between real and augmented reality. Therefore, we conducted an experiment to investigate our hypothesis.

HG: Gaze depth differs between viewing real and augmented objects, as measured by eye tracking.

Apparatus

We opted to use Microsoft HoloLens together with an eye tracker from Pupil Labs for multiple reasons. In this way, we were able to keep the gaze measurement device identical between the real and augmented scenarios, avoiding device dependency as a confounding variable. Using an optical see-through device, such as HoloLens, allowed us to keep the HMD and the eye tracker on the participant's head and continue measuring without the need for switching hardware or recalibration. Video see-through was not an option for our experiment, as it presents images of the real environment on the same focal plane as the virtual augmentations. The hardware of the Pupil Labs HoloLens Binocular Add-on integrates well with the HMD. It is small, unintrusive, and does not add much weight. Its right camera stopped functioning during initial work. We replaced it with a compatible camera from a Pupil Core setup. While the original camera supported capturing at up to 200 Hz, the replacement part was only capable of 120 Hz. Therefore, the Pupil Capture software defaulted to the lower frame rate, providing gaze data at up to 120 Hz. We selected low capturing resolutions of 192×192 (left) and 320×320 (right) because they yielded the highest accuracy in preliminary tests.

We used the Unity game engine for creating and rendering virtual objects. It only called our custom code for data recording once for each rendered frame. Thus, the logging frequency was variable and depended on the rendering performance of HoloLens. An open source Unity package from Pupil Labs facilitated communication and calibration for the eye tracker. The calibration scene was set up to include calibration points on four ellipses at various depths to improve accuracy in the third dimension. The same hardware setup measured gaze depth with real and virtual objects, i.e., the quality of the measurements depended on the alignment between both types of objects. Several ArUco markers at known positions served to align the virtual scene with the real environment.

Stimulus and Task

As a stimulus, we designed a common scene for the study and built a virtual and a real version of it. It consisted of five books $B1$ to $B5$ that stood on a table (see Figures 8.1c and 8.2). They were positioned at different distances and angles, as shown in Figure 8.1a. All books were close to or within the recommended comfort zone of HoloLens (1.25 m to 5 m).

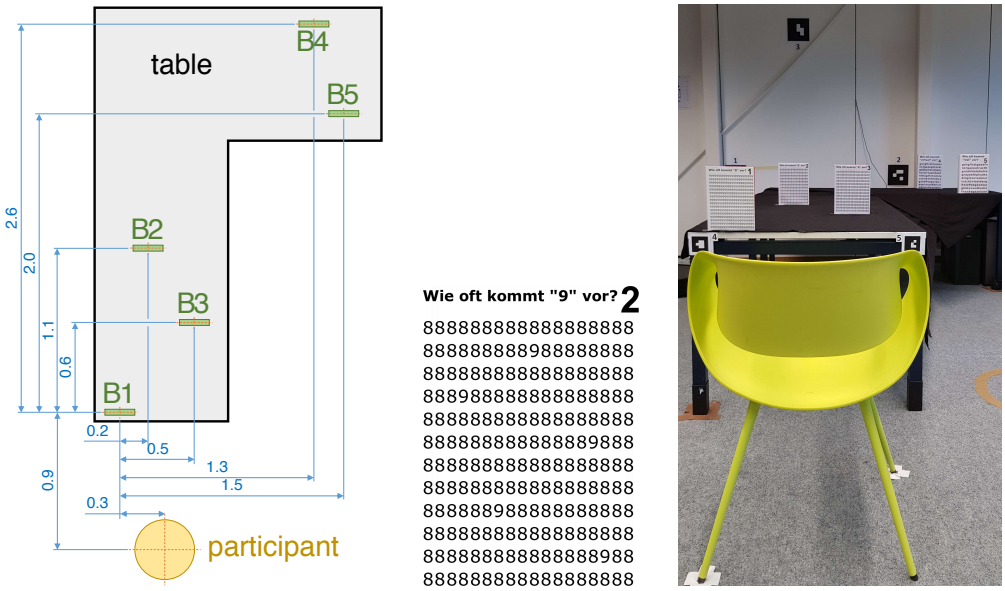


Figure 8.1: (a) General layout of the experimental environment, with books ($B1$, ..., $B5$) placed on a table. (b) We covered the books with visual search puzzles: “How many nines are there?” (c) We placed ArUco markers on the table and wall to match the virtual scene to the real environment. The adhesive tape helped to position the furniture.

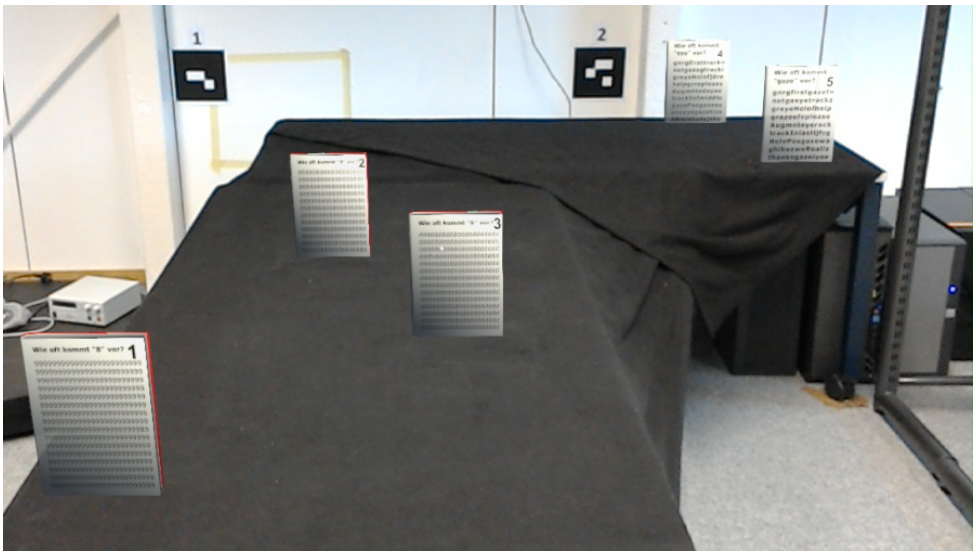


Figure 8.2: Augmented scene from the experiment, as seen through the HMD. We put a black cloth on the real table to avoid light shining through the augmented books.

Books $B4$ and $B5$ were farther away and needed to be larger to remain readable: 21.6 cm × 28.7 cm versus 17.4 cm × 24.1 cm.

We created puzzles that consisted of visual search tasks with letters and numbers (see example in Figure 8.1b) in order to engage participants and make them fixate on different parts of the scene. The tasks were designed to let participants view different objects in the scene at different depths, which served as the data of interest for our study. We replaced the original book covers with these visual puzzles to make the participants look directly at the objects and asked participants to look at the books one after the other.

We would like to provide the source code of the scene and its surrounding software online. Unfortunately, the diverse and extensive licensing agreements of the Unity game engine and third-party assets—necessary for compilation and at runtime—restrict our ability for redistribution.

Study Design

Our experiment adopted a within-subject design. The independent factor was the type of presented books with two levels: real vs. virtual (within AR). The dependent variable was the error, i.e., the difference between measured gaze depth and actual distance to the currently observed object. The within-subject design led to each participant experiencing both conditions with the same hardware setup, providing measurements without the disadvantages of recalibration. The order of conditions was counterbalanced to avoid systematic effects of learning.

Participants

We estimated the required number of participants based on power analysis. With the significance level 0.05, power 0.8, and effect size 0.8 (Cohen's d), we arrived at a minimum sample size $n \geq 25.52$ for a two-sided t-test. We recruited 26 participants (12 female, 14 male). The age ranged from 18 to 33 years (average 23.3 years). Among the participants, there were 22 students, one person with completed vocational training, and one person with a university entrance qualification. Out of these, 16 persons had their major in a field in, or related to, computer science. Twenty reported having some previous experience with AR/VR devices, including three regular AR/VR users.

We used a Snellen chart to check and confirm that all participants had normal or corrected-to-normal vision. Eleven participants wore glasses, and two had contact lenses. Participants received 10 EUR as compensation.

We had to abort the experiment with one participant due to a hardware failure. Therefore, valid study data is only available for 25 participants.

Procedure

First, we asked participants to read and sign a consent form—which included a short introduction—before giving them monetary compensation. They drew a random number, which became their ID for pseudo-anonymous data. Then, we went through a demographic questionnaire and checked their eyesight.

We asked participants to sit on a chair and fasten the HoloLens strap tightly to their heads to minimize movement during the experiment. This helped with the quality of eye tracking and positioning of virtual objects in the real environment. However, we could not record objective quality metrics—other than a general accuracy of 1°—because they were not available in the provided software. Once the HMD was strapped to a participant's head, we performed a calibration of HoloLens and the eye tracker.

We started the actual experiment with either the real or virtual scene, depending on the participant's ID, to counterbalance learning effects. We allowed for a short break—without removing the HMD—so that the conductor of the experiment could manually set up or remove the physical books for the next scene. During the experiment, participants were supposed to perform the search task by targeting a book with HoloLens' head-gaze cursor and pressing an external clicker to mark the beginning of their task. This hid the cursor and started gaze recording. We stopped recording as soon as the participant provided their answer to the puzzle and pressed the clicker again. At the end of the 45 to 60-minute experiment, we asked for subjective feedback in a structured questionnaire.

8.2.3 Results

Having recorded the study data with all participants, we can now analyze the results.

Data Filtering

The measured gaze depth in both scenes contains outliers that extend to -150 km. This would mean that participants looked through their own heads and far behind them. The eye tracker is accurate to about 1°. Therefore, a gaze depth of more than 2 m could theoretically result in a measured distance of infinity. The room in which we conducted the experiment was

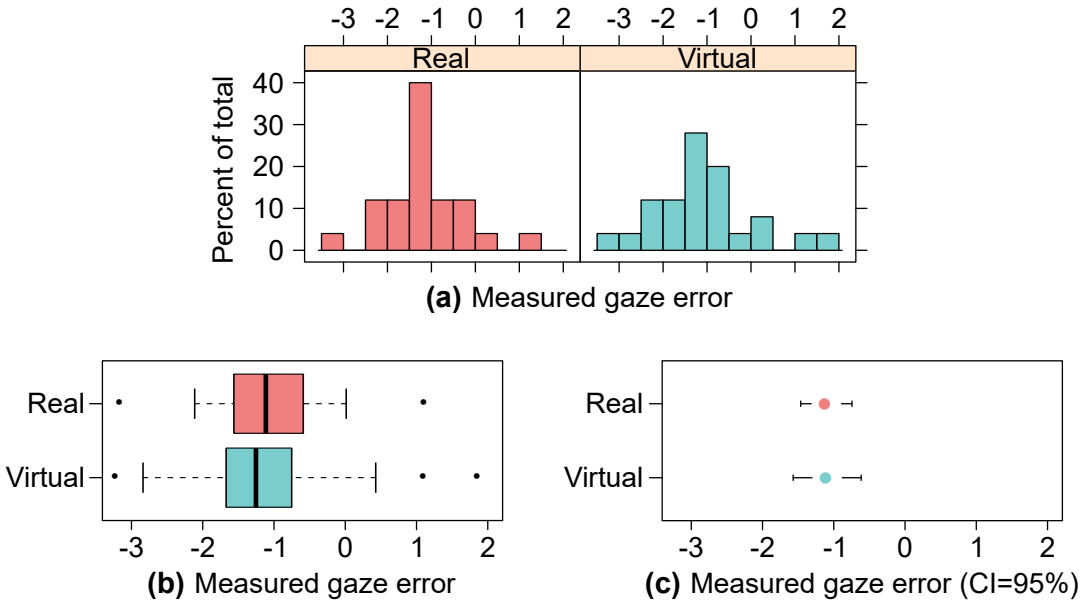


Figure 8.3: Error in measured gaze depth in the real and virtual scene. A histogram (a) and box plot (b) show the distribution, while the 95 % confidence intervals of means are encoded as a dot-and-whisker plot (c).

not longer than 10 m, and participants were seated approximately in the center. Therefore, we discard measurements that result in a gaze depth of more than [-10;10] m. However, we do not remove any other data points to allow for inaccuracies from overestimation and underestimation of depth to cancel each other out.

Analysis of Gaze Data

The filtered data includes over 380,000 gaze depth measurements. The number of recorded measurements varies between each person and book because we did not enforce a specific time for the completion of each puzzle, and the rendering performance on the HMD was not constant. First, we average the error between measured gaze depth and the actual positions of the books for each participant, book, and scene. Then, we use the mean of observations between all books to get the final data by participant and scene. Through this aggregation, we obtain 25 samples (i.e., participants) for each condition (i.e., real vs. virtual). Figure 8.3 shows the resulting data distributions. The histograms (see Figure 8.3a) exhibits a tendency toward underestimation when measuring the gaze depth via eye tracking. The median depth error for real books is -1.11 m ($M = -1.10$, $SD = 0.87$). For augmented books, this value is higher: -1.25 median error ($M = -1.09$, $SD = 1.16$). While the values differ, the standard deviation indicates very wide distributions in both scenes. The confidence intervals

(see Figure 8.3c) do not exhibit significant differences. The recorded data indicates that the error increases with the book's distance. This seems plausible, given that the measured depth scales with the tangent of the gaze angle.

A Shapiro-Wilk test confirmed the impression from the histogram in Figure 8.3a: the data seems normally distributed, both for the real ($W = 0.97$, $p = 0.72$) and the virtual scene ($W = 0.96$, $p = 0.35$). A paired t-test shows that there is no statistically significant difference between the two conditions ($t = -0.06$, $df = 24$, $p = 0.95$, 95 %-CI $= [-0.37, 0.35]$, mean of differences $= -0.01$).

Subjective Feedback

In the questionnaire, participants generally reported good visibility of all books. However, there was one person who had trouble with real $B2$. In the virtual scene, one participant had issues with $B1$, whereas two had problems with $B2$ and $B3$. Five persons reported having difficulties with focusing on the books, and one restricted the answer to only the virtual objects. Only two participants were not able to discern real and virtual books. The others mentioned that the virtual ones looked translucent and less clear. Two reported that the books seemed to be glossy. We also asked participants whether they agreed with the statement that "real and virtual books look similar." They responded on a Likert scale from 1 (*strongly disagree*) to 5 (*strongly agree*). The high mean of 4.04 indicates a good similarity between the conditions.

Only four people reported they felt absolutely fine during the experiment. Most complained of burning eyes, an uncomfortably warm HMD, headaches, or pressure on their head. Three participants reported nausea.

8.2.4 Discussion

The t-test shows that the differences between real and virtual books do not seem to be sufficiently large to confirm our hypothesis **HG** from Section 8.2.2. Our results suggest that, despite *subjective* differences in depth perception found in previous work [RGA95; SSE15; KCS17; Duc+14], *objective* measurements from eye tracking do not seem to be affected by the VAC in AR. This outcome is positive with our original motivation in mind: supporting gaze for intuitive interaction in situated analytics, e.g., volumetric 3D selection.

However, the recorded depth data contains much noise, and the eye tracker tends toward consistent underestimation of gaze depth with all books. The average deviation between measured and actual depth increases

with the object's distance. It is already more than 1 m when the focused target is only 3.5 m away from the viewer. Please note that HoloLens was firmly strapped to the participants' heads—causing some discomfort over a longer period of time—and they were seated on a chair. Despite our best efforts, the eye tracker did not stay tightly attached without any movement. In practice, the time-dependent degradation of the quality of measurements would require frequent interruptions of situated analytics for recalibrations. Additionally, the infrared radiation from the eye trackers in very close proximity to the participants' eyes seems to have caused dryness and strain. Combined with discomfort from the HMD, this might have led to a loss of focus, fatigue, and an increased blink rate, which might have also decreased the precision of our measurements.

Useful integration of 3D gaze information for situated analytics requires more precise eye trackers or is restricted toward very specific use cases. If, however, the intersection of a single gaze ray with an object at a known distance is sufficient, both first- and second-generation HoloLens are well suited. Despite difficulties with the eye-tracking hardware, the visual appearance of the virtual books was accepted well. Participants noticed differences from the real objects but generally found them to be a close match and were able to complete the tasks. This indicates that the study design itself is sound and could be applied to future hardware.

We are confident that further developments, e.g., calibration-free setups with advanced machine-learning algorithms [Pup20], will one day enhance the quality of the depth data to a degree that could help to realize our envisioned role of 3D gaze in situated analytics. Until such time, we advocate the use of 2D eye tracking for its already existing benefits in supporting analysts to achieve their tasks—for example, with situated 2D visualizations of the setup in Chapter 7.

Part III

Parallel Coordinates Plots

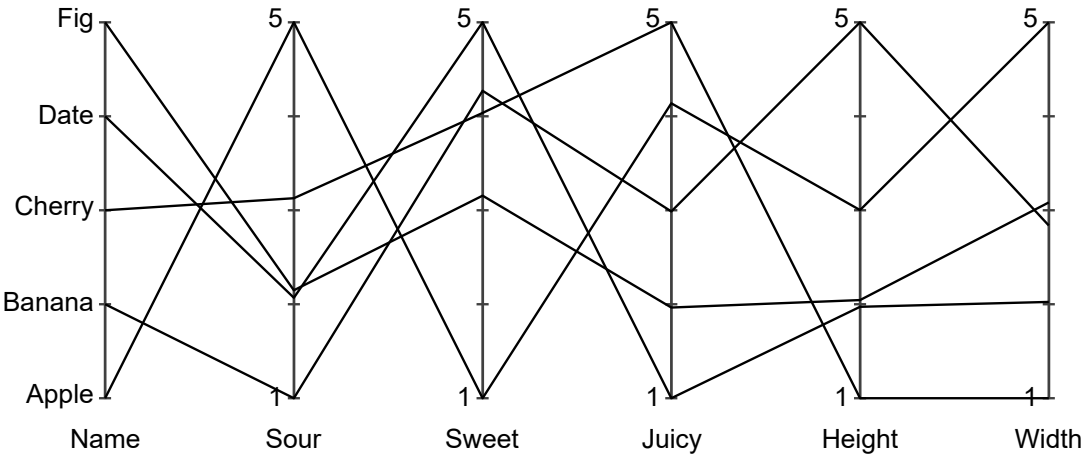


Figure III-1: Parallel coordinates plot of fictitious fruit data. Data points become polylines. Intersections of lines and axes encode the underlying data values.

In Part II, we only use animation for transitions between static scatter plots. Such an approach is suitable for individual analysts to explore data. But how can they communicate findings with their peers if more than two data dimensions are involved? How can they re-trace their own findings and continue analysis from a previous state? The animated sequences of plots do not show all information at once and might require custom analysis software or video-editing to annotate. While animation can be of great help for the interactive exploration, there are benefits of static visualization. All information within images is visible at the same time. Viewers can add arbitrary annotations by writing on a print-out or with relatively simple image editing tools that are even available on mobile devices. Non-animated visualizations reserve the interaction channel for other operations, e.g., brushing. Stationary plot elements are easier to select with pointing devices for further analysis.

On one hand, scatter plots are best suited for 2D data and provide an intuitive display for data clusters in dimension pairs. On the other hand, they require switching between views to analyze multivariate data. But there are alternative plot types that can represent multivariate data with static images to answer RQ3. To visualize multiple data dimensions, we can place axes in a parallel instead of a perpendicular arrangement. Such parallel coordinates plots (PCPs) draw a polyline for each data point (see Figure III-1). The intersection of the line with each axis determines the underlying value of the data point in the corresponding dimension. It is possible to follow the lines and read data values of individual points in all dimensions. But compared to scatter plots, it is much more difficult to see small data clusters.

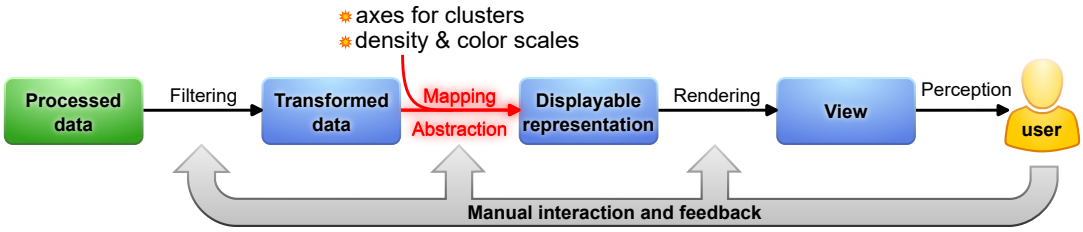


Figure III-2: The visualization pipeline (adapted from [CMS99]) with additions for the proposed techniques from this part. Affected elements are highlighted in red. Clusters are represented by duplicating PCP axes. Custom overlaid density rendering reveals trends and individual data points, allowing color to encode additional information.

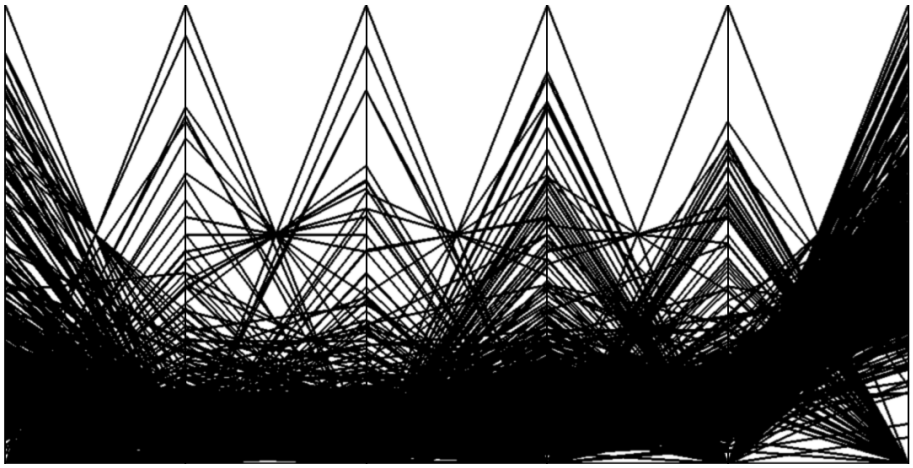
In this part of the dissertation, we will first discuss how line-rendering in PCPs affects readability. Then we will build upon the concept of parallel coordinates to target the need for a static visualization of information that would otherwise be visible in multiple 2D scatter plots. The presented techniques will answer RQ3 by introducing a new abstract mapping for fuzzy data clusters and by adding custom opacity maps from line density that allow the viewer to follow individual lines in situations with dense overdraw (see Figure III-2).

Rendering Lines for PCPs

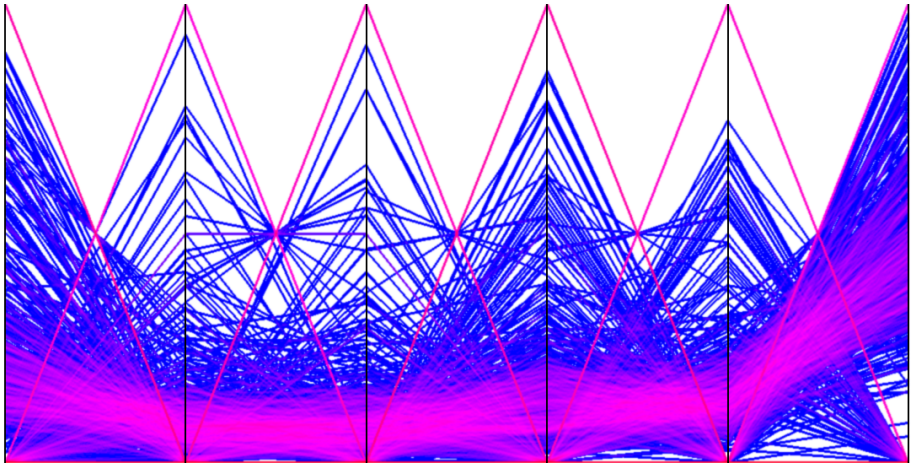
In a scatter plot, each data point is shown as a single glyph. Most often, that glyph is a small circle and does not require much “ink” or pixels to be drawn. In comparison, the polylines used in PCPs are much less efficient. When the number of samples in the underlying data set increases, PCPs tend to quickly become cluttered and full of overdraw (see Figure III-3a). It is a common interaction technique to restrict the number of drawn data points by applying a range filter on individual axes. A live demonstration of this approach is available on Heinrich’s website [Hei16], where the user can easily drag along an axis to define a range, and the polylines of non-matching data points turn almost completely transparent.

But what if it becomes necessary to view the entire data set without filtering? Instead of reducing clutter by removing lines, we can use splatting to show how often each pixel has been used (see Figure III-3b). To this end, we split line rendering into two parts: First, we draw all lines into a buffer where each pixel counts how often it has been used. Then, we map this numeric buffer to color.

Alternatively, we can reduce the RGB values of each line and use additive blending without any additional changes. For example, if we want to draw n



(a) Regular black lines.



(b) Splatted with custom color map:  .

Figure III-3: Comparison of rendering techniques for PCPs. The data set contains 900 samples of word relevance from multiple text sources.

lines in a 24-bit RGB image, we set the color channels of each line to $255/n$. This additive approach has high performance and is simple to implement. However, it only works with a limited number of lines and restricts the usable colors for the designers of PCPs. Additionally, if the lines do not all overlap in a single pixel, the available color space is not fully utilized.

In the following chapter, we will use a combination of additive blending and custom scaling to create a technique that supports more data points and can also be used with multiple distinct colors.

9

Cluster-Flow Parallel Coordinates Plots

This chapter is based on previous work and will use extracts thereof without explicit quotation:
N. Rodrigues, C. Schulz, A. Lhuillier, and D. Weiskopf. “Cluster-flow parallel coordinates: Tracing clusters across subspaces.” In: *Proceedings of Graphics Interface*. Canadian Human-Computer Communications Society, 2020, pp. 382–392. DOI: 10.20380/GI2020.38.

9.1 Motivation

We aim to answer RQ3 by creating a static visualization with the advantages of scatter plots for 2D subspace clustering and animation for tracing individual points and clusters between pairs of data dimensions. PCPs provide traceability for individual lines and go back to Inselberg’s seminal work from the 1980s [Ins85; ID87]. His book provides a comprehensive and more detailed presentation of PCPs [Ins09].

A high number of data points can quickly lead to overdraw and, thus, visual clutter in PCPs [HW13]. This makes it hard for users to explore and analyze patterns in the data set. We need to tackle this problem before we can commit to using PCPs as the basis for an answer to RQ3.

Beyond the already discussed techniques that focus on filtering and rendering, there are also multiple approaches that group the lines and then only need to draw a single element for each group. Fua et al. [FWR99] reduced the number of visible lines by applying hierarchical clustering and allowing the user to adjust the desired level of detail. Inselberg [Ins85] himself suggested only drawing the outer convex hull of the lines between two axes. More recently, Palmas et al. [Pal+14] proposed precomputing

1D clusters for each dimension using a kernel density estimation approach and then linking neighboring axes using compact tubes in which the width encodes the number of data points in the cluster. We do not accept the suggestions from these publications because showing only a few lines or large polygons does not allow viewers to follow arbitrary individual data points.

Another concept to reduce clutter is using curves instead of straight lines. It might seem unintuitive at first because curved lines are always longer than straight ones and, therefore, cover more pixels. However, the technique behind the curves can compensate for the increased length by directing overdraw into less important areas of the plots, e.g., the center space between axes [Net+17]. Additionally, the longer line paths can emphasize otherwise invisible information. Edge bundling, for example, has shown to be an effective technique that helps users find implicit clusters and patterns within PCPs [LHT17; Hei+12]. Illustrative parallel coordinates (IPCs) [MM08] bundle PCPs in image space by preclustering the data set (via k-means) and then rendering the lines with B-splines. Zhou et al. [Zho+08] used a variant of force-directed edge bundling to directly compute the clusters based on the patterns emerging from the bundling algorithm [HW09]. Heinrich et al. [Hei+12] extended previous work by providing C^1-continuity between B-splines to emphasize end-to-end tracing of data points.

Bundling fits our goal to transfer the advantages of multiple scatter plots to a static visualization: it reduces clutter and can emphasize clusters in PCPs. However, the axis order is what inherently determines the visibility of patterns for the viewer to explore [Weg90]. It is similar to analyzing a SPLOM through a sequence of scatter plots and related to the problem investigated in Part II within RQ2. Showing all possible pairs of dimensions is not feasible for high numbers D of data dimensions.

Ordering the axes in a PCP is a known area of research. Pargnostics [DK10] uses metrics such as the number of crossings, the angle of crossings, or the parallelism between dimensions for ordering. A method by Tatu et al. [Tat+09] ranks axes using features of the Hough transform and subspace similarity measures. Alternatives from other researchers include approaches that work with dimension similarity from Euclidean distance [ABK98] or clutter-based measures [PWR04]. Zhao and Kaufman optimize the axis order through several clustering techniques, e.g., k-means [ZK12]. In general, all these methods focus on subspaces. This fits our goal within RQ3, for which we want an axis order that emphasizes the same patterns that are visible in multiple scatter plots. We, too, will need to define similarity measures that work on the 2D subspaces of adjacent PCP axes. However, we will use Fuzzy DBSCAN to determine clusters

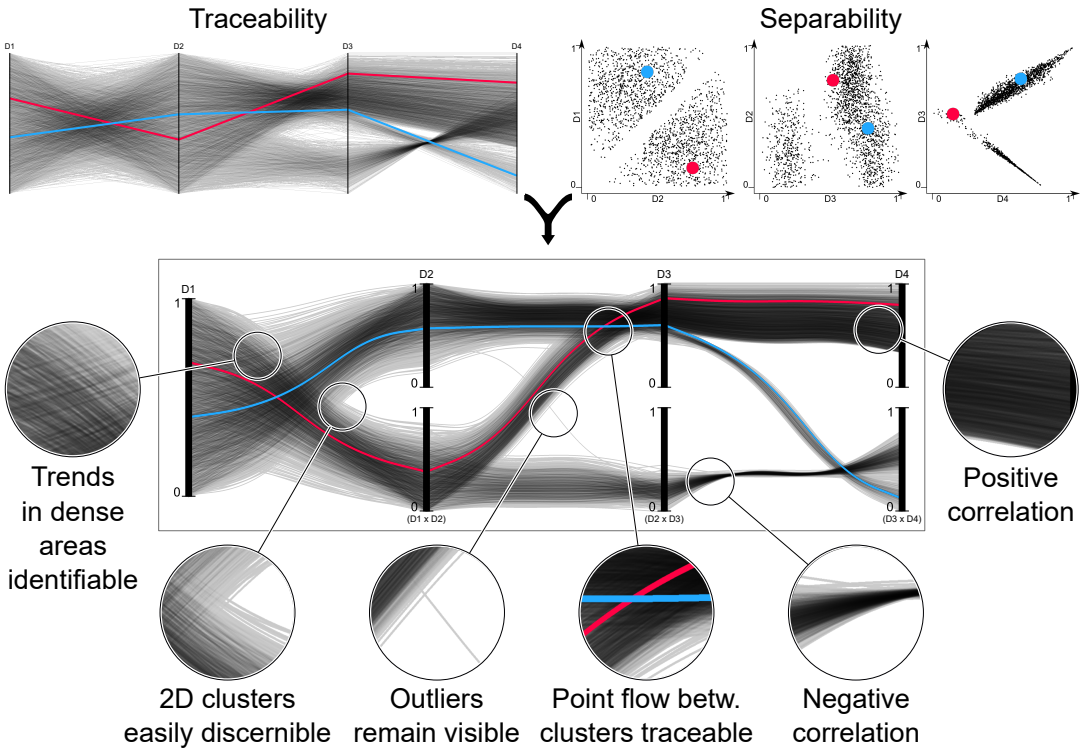


Figure 9.1: Feature overview of CF-PCPs using 2,000 generated data points in dimensions $D1$–$D4$. They combine advantages of regular PCPs and scatter plots. CF-PCPs are read from left to right: The data is grouped by pairwise dimensional clustering, i.e., stacked axes beneath D_i show all clusters from subspace $D_{i-1} \times D_i$. D_1 depicts no clusters due to the lack of a preceding neighbor. CF-PCPs allow for salient illustration of clusters and traceability across multiple dimensions alike. Individual data points can be traced over all data dimensions (see red and blue lines).

without restricting their number [IB18].

To answer research question RQ3 and inspired by previous work on RQ2, we will develop a novel approach in this chapter that does not require interaction: cluster-flow parallel coordinates plots (CF-PCPs). As depicted in Figure 9.1, our technique combines the traceability of points and cluster interactions with the clear separability of clusters, which would otherwise be a specialty of scatter plots with animated transitions. At the same time, CF-PCPs maintain the readability of correlations and other data characteristics for which regular PCPs are well suited. Despite visual similarities at first glance, our approach is unique. Kosara et al. [KBH06] visualize data flow but for categorical data instead of clusters, and their portrait layout of PCPs does not calculate an optimal order for dimensions or categories. Nested PCPs [Wan+17] look very similar but have a global clustering instead of using subspaces because they are meant for ensemble

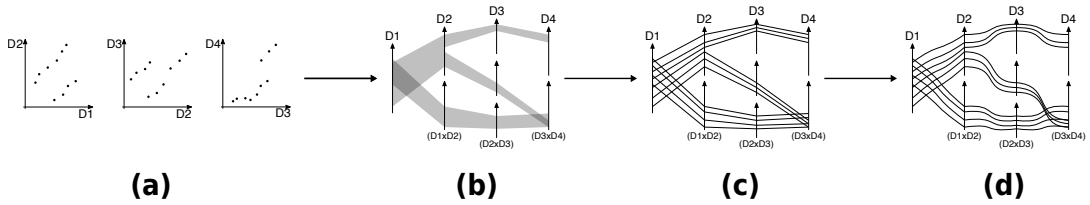


Figure 9.2: Construction of CF-PCPs. Scatter plots are well suited to show clusters of multivariate data in 2D subspaces (a) but do not readily show cluster relations across more dimensions. We extend PCPs by creating an individually duplicated axis for each cluster to show high-level flow between clusters (b). Showing the underlying data points as individual lines increases the available level of detail (c). To maintain visual data patterns and the perception of correlations, we restore the original PCP's line angles near the axes (d).

visualization in conjunction with other plots.

9.2 Model and Overview

We first provide an overview and outline the intermediate steps of our technique (see Figure 9.2). We assume that we have a set of data points in a multivariate data set as input: $P = \{\vec{p_i} \in \mathbb{R}^n\}$. An algorithm assigns each of these points a degree of membership to clusters, resulting in tuples $(\vec{p_i}, m_{k,i})$. Here, $m_{k,i} \in [0, 1]$ describes the degree to which data point $\vec{p_i}$ belongs to the cluster with index k. Hard clustering is a special case of this, with $m_{k,i} \in \{0, 1\}$. In contrast, soft labeling allows for partial memberships. A single cluster is then defined as $c_k = \{(\vec{p_i}, m_{k,i}) \,|\, m_{k,i} > 0\}$ and $C = \{c_k\}$ comprises all clusters. We focus on subspace clustering because distance measures lose expressiveness when dimensionality increases, leading to superfluously fuzzy or inconceivable clusters. Hence, we compute clusters for each pair of dimensions $(\mathbb{R}_i, \mathbb{R}_j)$ as shown in Figure 9.2a, i.e., we look at sets of 2D subspaces. Our current implementation employs Fuzzy DBSCAN [IB18], which is an extension of the classic and popular DBSCAN algorithm, for fuzzy clustering. However, our visualization is independent of the chosen clustering algorithm; it just assumes that clustering provides labels for each data point. The data dimensions that serve as the source for clustering are shown beneath the axes.

The goal of CF-PCPs is to show information about subspace clusters and the actual data points alike. On a coarse level, they show the number of clusters in the subspaces and how data elements virtually flow from one cluster to another (see Figure 9.2b). Our approach to showing this flow is based on axis duplication: instead of a single (vertical) axis for a data

dimension, we place clones of this axis on top of each other, i.e., at the same horizontal position. Each data cluster gets its own axis clone. Through this duplication, we can render the stream of data elements between clusters, which are visible even if one does not focus on individual lines in the CF-PCPs but only looks at the visualization as a whole. Section 9.3 describes the details of the axis duplication and how we can achieve an optimal layout of the data dimensions and cluster order.

By rendering an individual line for each data point, we include further details (see Figure 9.2c) in a fine-grained level of our proposed visualization. We use a curve model (see Figure 9.2d), as described in Section 9.4, that allows us to infer correlations and other data characteristics for individual data elements, similar to regular PCPs. Furthermore, we present a density rendering and color mapping approach that allows us to visualize large data sets and uncertainty from fuzzy clustering (see Sections 9.4.2 and 9.4.3).

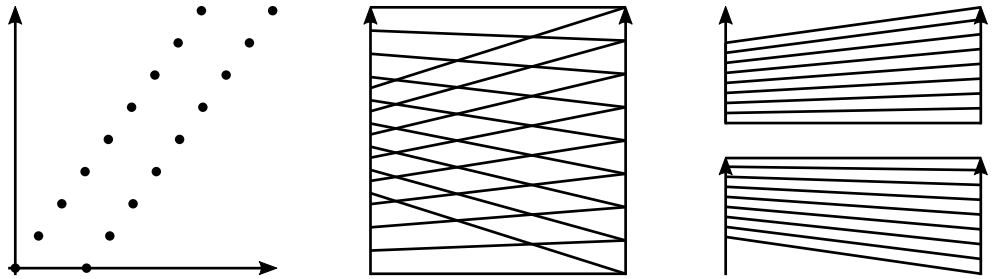
9.3 Layout

The key aspect of our layout of CF-PCPs is the duplication of axes (Section 9.3.1) because this serves as a basis to visually separate clusters and show the flow between clusters in the different subspaces. Each ordering of axes results in a unique flow pattern between clusters. Hence, we present a new method to optimize the horizontal axis order according to patterns in data flow (Section 9.3.2). Section 9.3.3 introduces a technique to order the clusters on each data axes (y-axis) vertically.

9.3.1 Axis Duplication

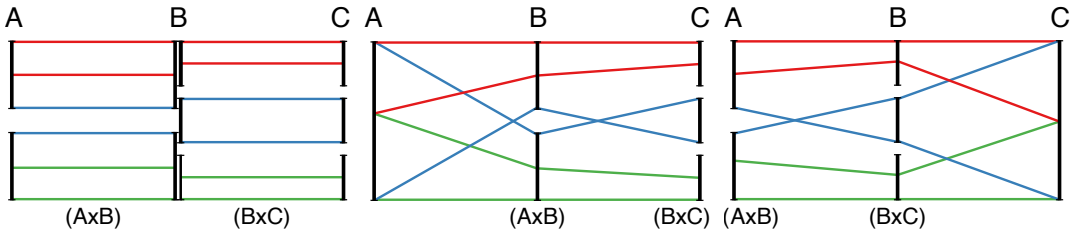
CF-PCPs extend traditional PCPs by using the vertical image space to show cluster assignments. Unlike any other PCP variant, we visualize each cluster on its own (local) axis. As Figure 9.3 shows, we create two duplicates of the axes to draw the two clusters and stack them vertically.

Axis duplication becomes necessary because we aim to show clusters in 2D subspaces, not just in 1D [Pal+14]. For the latter case of 1D clustering, we could just squeeze the data vertically to bundle the line into clusters on each axis individually. However, this approach will only work if the clusters are already separable in a single dimension. Figure 9.3a shows an example with two clusters that are visible in a scatter plot and linearly separable in two dimensions but not along 1D axes. Here, traditional PCPs lead to intertwined visualizations of the two clusters, and even axis scaling could not separate them (Figure 9.3b). With axis duplication (Figure 9.3c), we now see two clearly distinguishable bundles of lines that correspond to



(a) Cartesian (SP) **(b)** Parallel (PCP) **(c)** Stacked (CF-PCP)

Figure 9.3: Illustration of two clusters using different coordinate systems. The scatter plot (a) uses Cartesian coordinates to clearly show them as dots placed along two straight lines. The clusters are hard to distinguish in a regular PCP (b) without additional visual variables, e.g., color. Our cluster-flow layout duplicates PCP axes for each cluster to reduce clutter and increase readability (c).



(a) No direction **(b)** Left-to-right **(c)** Right-to-left

Figure 9.4: Various reading directions with data points colored according to clusters in $B \times C$. Without a reading direction, we obtain overlapping axes and discontinuous lines (a) because the 2D subspaces $A \times B$ and $B \times C$ do not have the same number of clusters. With a reading direction, axes show the clusters within the subspace of the current dimension and its neighbor to the left (b) or right (c).

the two clusters, i.e., clusters are easy to recognize. However, there is yet another important benefit: the PCP lines can now be inspected for each cluster separately, and thus, we can investigate correlations or other data characteristics independently within each cluster.

Up to now, we have only handled a pair of two data dimensions, but parallel coordinates support multivariate data. Figure 9.4 shows an example sequence with three dimensions. In subspace $A \times B$, there are two clusters, so we create two copies. $B \times C$ contains three clusters, so we draw three copies. This results in overplotted axes for dimension B and interrupted polylines for the data points (see Figure 9.4a). Therefore, we introduce a reading direction left-to-right (LTR) to create as many copies of axis B as there are clusters in $A \times B$. We duplicate axis C as often as there are

clusters in $B \times C$ (see Figure 9.4b). Since axis A has no pair to the left, there is no explicit clustering and no duplication. While the opposite reading direction works accordingly (see Figure 9.4c), we chose a consistent LTR layout for all figures in this chapter. Beneath the axes, CF-PCPs also show the two data dimensions that were used for clustering to help the viewer identify the reading direction.

Cloning the axes and stacking them vertically increases the plot area and vertically shears the lines that represent the data points, especially for large numbers of clusters. We address both issues by scaling down cloned axes with a root function. Assuming the height h_1 of a single regular PCP axis and n as the number of stacked axes, we get a total height of

$$h(n) = n \cdot h_1 \cdot (1/n)^r \ , \tag{9.1}$$

where $r \in [0, 1]$ controls the root scaling. Selecting $r = 0$ will result in no downscaling at all, while $r = 1$ will keep the original height without any growth. Spaces between axes are determined with

$$s(n) = s_2 \cdot h_1 \cdot (1/(n-1))^r \ , \tag{9.2}$$

where s_2 is the percentage of h_1 we want to use as space between two duplicates. The remaining space is then used for the actual axis clones. We chose $r = 0.8$ and $s_2 = 0.1$ for figures in this chapter to balance between growing height and readability.

Axes in regular PCPs are normalized to only display the used range of each data dimension. Multiple labeled tick marks enable viewers to read values from data points at their intersections with the axes. As the number of clusters rises, axes in CF-PCPs shrink. Retaining the same amount of labeled tick marks as in regular PCPs would add more overdraw to the already densely packed area around the axes. Therefore, we limit the tick marks to the maximum and minimum of the data range in each data dimension and progressively scale the label size according to the space between axes.

9.3.2 Dimension Ordering

Just as with regular parallel coordinates and animated scatter plot transitions, the visual patterns in a CF-PCP depend heavily on the order of data axes. In the case with scatter plots from Part II we relied on user interaction and provided recommendations. It was then up to the viewer to decide on a sequence of data dimensions. Later changes to a static visualization are difficult to accomplish. For this reason, we propose a method for optimizing the order automatically in CF-PCPs.

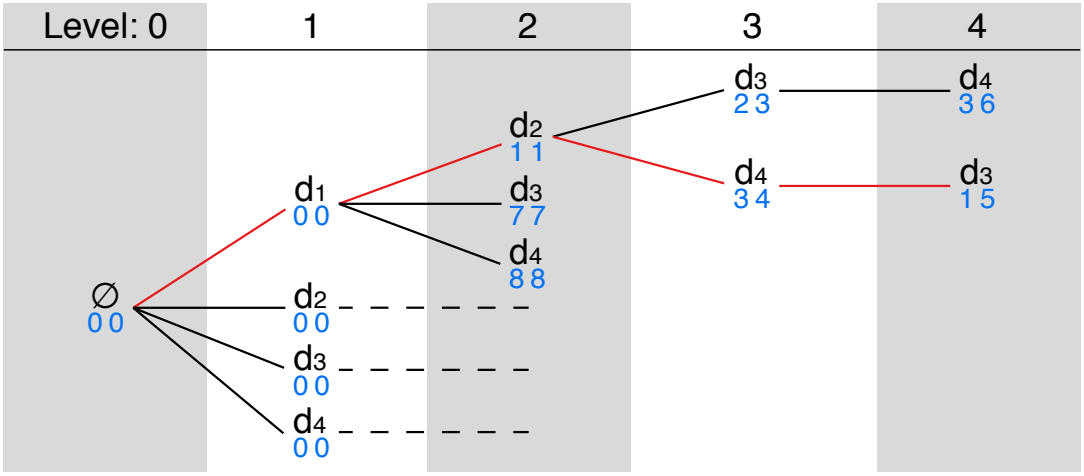


Figure 9.5: Tree representation of our model for ordering dimensions d_1 to d_4. The blue numbers beneath the dimension names represent costs (left: individual; right: accumulated). Levels 0 and 1 have no costs because there are no pairs of clusters yet. The path d_1, d_2, d_4, d_3 is optimal as it is at the deepest level, and others are at least as expensive. Missing paths at levels 3 and 4 have not been expanded yet, because their costs were never minimal.

Previous work investigated various metrics—like the number of crossings, angles between lines, correlation strength, and more—to optimize the order [DK10; JC08]. At first glance, we could have simply reused existing algorithms or defined dimension ordering as an overlap-removal problem for the polylines in the PCP. However, CF-PCPs do not only show the underlying data points but also depict the flow between subspace clusters on a coarse scale. For example, to calculate the costs of chaining two sets of clusters between display axes a_i and a_{i+1}, we also need to know the previous axis a_{i-1}. This is due to our 2D subspace clustering and reading direction LTR: clusters shown at a_i are calculated with data from dimensions $d_{i-1} \times d_i$. Only then can we calculate costs with the next axis a_{i+1}, which contains clusters from $d_i \times d_{i+1}$.

As shown in Figure 9.5, we choose a tree T as the primary data structure for our proposed order optimization. We start by creating the root node at level 0 and adding a child for each data dimension at level 1. Then, we expand each leaf recursively by attaching a node for each unused data dimension. Each level corresponds to an index in the sequence of display dimensions. This way, for D data dimensions, we obtain a tree with $D + 1$ levels, where each node has $D - level$ children. The tree contains $D!$ different paths—corresponding to all permutations of the data dimensions. Costs from one set of clusters to the next depend only on three consecutive

data dimensions. Thus, there are only $D \cdot (D-1) \cdot (D-2)$ different costs to compute, which can be cached and reused in the tree traversal.

We apply the A* algorithm [HNR68] to compute the shortest path in T and implement it by lazily expanding a tree node when it is the leaf with the lowest cumulative costs. The heuristic $D - level$ estimates the remaining costs. This has implications for the metric we use to define the distance between two sets of clusters: The minimum cost from a parent to a child has to be 1 for the heuristic to work. With this model, the A* algorithm is guaranteed to find an optimal solution. We combine lazy evaluation with caching of cost values to compute the A* optimization efficiently.

Having decided on a basic optimization algorithm, we choose a measure for the costs of displaying clusters next to each other. CF-PCPs are targeted at showing how clusters evolve from one subspace to the other, i.e., how the data points flow from one to another. Therefore, we propose a metric that focuses on the similarities and differences of clusters. Let c_i and c_j be two sets of clusters. The number of elements in them is $n = |c_i|$ and $m = |c_j|$. In the first step, we generate a similarity matrix using the Jaccard indices [Jac01]:

$$S_{c_i,c_j} = \begin{pmatrix} J\left(c_{i,1}, c_{j,1}\right) & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & J\left(c_{i,n}, c_{j,m}\right) \end{pmatrix} . \tag{9.3}$$

The best match of cluster sets would yield few elements with value 1 and many with 0. Bad matches have many data points changing between clusters, leading to an array with many values < 1. Since we prefer to see dimension orders with many good matches, we subtract the matrix's mean value from all its elements $s_{x,y}$. We then square all results and sum them up into the grand sum to get a scalar similarity value between the sets of clusters:

$$similarity_{i,j} = \sum_{\substack{1 \le x \le n \\ 1 \le y \le m}} \left[s_{x,y} - mean\left(S_{c_i,c_j}\right) \right]^2 . \tag{9.4}$$

To obtain a cost function that fits the requirements of A* and our tree T, we invert the similarity. However, $sim(c_i, c_j)$ can be 0 when all matrix entries have the same value or when comparing sets of size 1. Thus, we define the cost function as

$$horizontalCost_{i,j} = 1 + \left[similarity_{i,j} + 1 \right]^{-1} \qquad \geq 1 \tag{9.5}$$

to avoid division by 0. Using our proposed metric helps the optimization algorithm in avoiding adjacent dimensions with no or only one cluster between them.

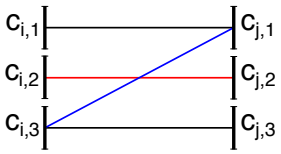


Figure 9.6: Simplified CF-PCP with cluster sets c_i and c_j. Top and bottom (black) never cross any other bundles. A red reference bundle only crosses lines that start below and end above (blue).

9.3.3 Cluster Ordering

We separate the optimization of horizontal dimension and vertical cluster order because CF-PCPs aim to primarily display the data flow between subspace clusters. The list of available clusters for display is controlled by the sequence of dimensions but not by their internal vertical arrangement. Therefore, the order of dimensions is our highest priority. Afterward, we still have flexibility in choosing the vertical arrangement along each data dimension. Our approach to optimizing the vertical sequence is similar to the dimension ordering: employ A* to find an optimal path from tree level 1 to a leaf at level D. The main difference lies in the way we generate the children of each node. When there are m clusters to be arranged, we generate all permutations and add them to the parent node, which is itself a node among permutations of the previous clusters.

We adopted a metric based on the number of line crossings as a cost function—which is popular for ordering in PCPs [DK10]—and adapted it for the display of data flow from one cluster to another. The comparison of Figures 9.3b and 9.3c shows that lines within cluster pairs move closer to each other. This is caused by the axis scaling from Section 9.3.1 and creates the impression of bundles. Intertwined lines within such a bundle do not interfere with tracing the data flow, but crossing bundles are difficult to follow visually. Therefore, we define our metric to penalize these inter-bundle crossings.

A bundle between clusters contains all shared points, e.g., $c_{i,1} \cap c_{j,1}$. We assume that each line has a certainty $l \in [0, 1]$ that describes its—possibly fuzzy—cluster membership. Fuzzy lines will not be as visible as certain ones and contribute less to overall clutter. See Section 9.4.3 for more details on how the certainty will affect line rendering. Our metric sums up the certainty within each bundle into L and uses it as an adjusted line count. This new count is used to populate a matrix with values for all bundles

between neighboring dimensions i and j:

$$B_{i,j} = \begin{pmatrix} L(c_{i,1} \cap c_{j,1}) & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & L(c_{i,n} \cap c_{j,m}) \end{pmatrix} . \tag{9.6}$$

Our approach calculates the total number of crossings using a method by Rit [Rit86]: bundles only cross if they start lower and end higher (see Figure 9.6). The opposite case (from higher to lower) is just a different view of the intersection between the same lines. The actual calculation is done by multiplying each element $b_{x,y}$ of $B_{i,j}$ with the submatrix ${}_{x,y}B_{i,j}$ to its lower left and then getting the grand sum. The first column and last row do not have a submatrix to the lower left, so we do not need to calculate crossings for them. The resulting values are written into a matrix $C_{i,j}$, and all its elements $c_{x,y}$ are summed up. The final cost for placing one sequence of clusters next to another is then

$$verticalCost_{i,j} = 1 + \sum_{\substack{1 \le x \le n \\ 1 \le y \le m}} c_{x,y} \qquad \geq 1 . \tag{9.7}$$

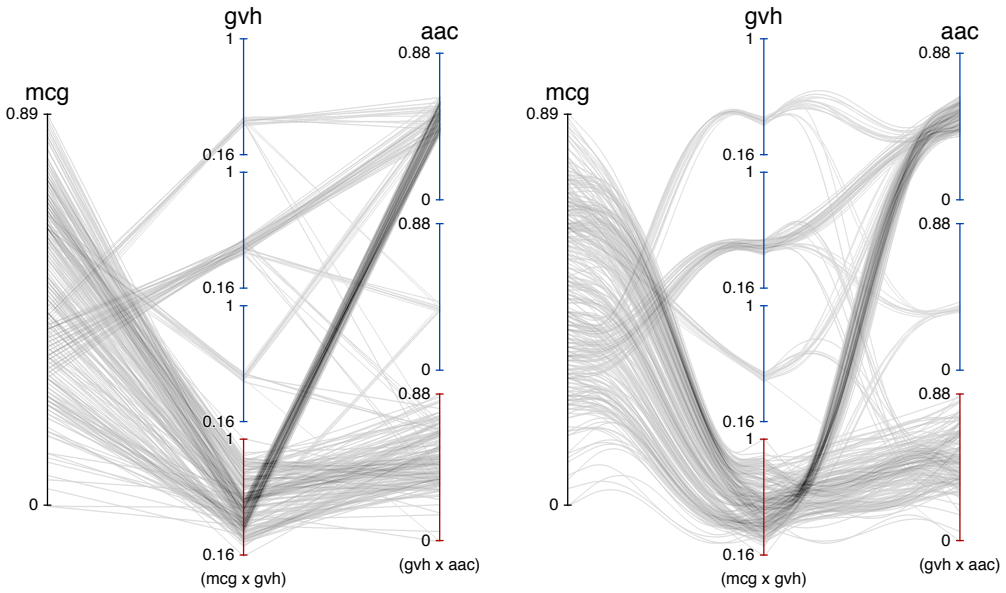
We start with a minimal vertical cost of 1 to ensure that the metric fits the requirements set by the level-based heuristic for A*. The width of the primary tree structure T grows much faster than in Section 9.3.2. Therefore, we recommend using the lazy A* algorithm for the optimization of up to 15 clusters per 2D subspace and reverting to a greedy approach for more complex problems. For example, sorting clusters by the mean value of contained data points approximates the vertical line layout from PCPs whilst retaining the advantages of their cluster-flow counterparts.

9.4 Line Mapping and Rendering

At this point, CF-PCPs exist as a model that does not completely specify how to display lines and encode uncertainty. To be able to render actual visual output, we now address these aspects.

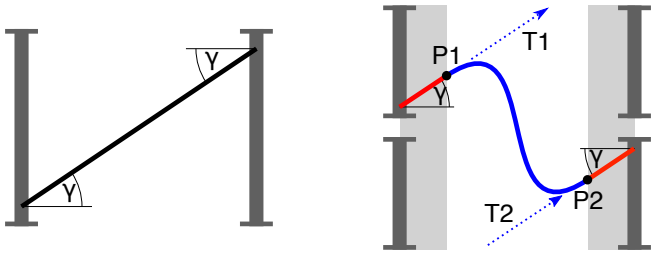
9.4.1 Curve Geometry

Traditional PCPs draw data points as straight lines between two neighboring dimensions. If we do the same in CF-PCPs, we get a relatively clean visualization without much clutter (Figure 9.7a). Unfortunately, the naive approach of axis duplication also changes the line slopes with respect to traditional PCPs. In fact, the slope depends no longer just on the data



(a) Straight lines **(b)** Hermite splines

Figure 9.7: Comparison of line geometry in CF-PCPs. Straight lines (a) make it hard to recognize correlations within clusters. Curved composite lines (b) preserve information on correlations by starting and ending with the same angles as in regular PCPs. Adjusting tangents used in the central Hermite splines separates lines that would cover and occlude each other completely.



(a) **(b)**

Figure 9.8: Composite line geometry in CF-PCPs. In regular parallel coordinates (a), lines have an angle γ, depending on the underlying data point's values in each display dimension. Lines in CF-PCPs (b) have segments with the same angle in a zone close to the axes (red lines on gray background). These are then connected by a cubic Hermite curve with tangents T1 and T2.

values but much more on the cluster membership. A related issue is that the angular resolution is reduced because lines become “compressed” (i.e., closer in angles) if they belong to the same cluster. Therefore, it becomes harder to recognize correlations. An eye-tracking study [Net+17] showed that participants focus primarily on the area around the axes when reading PCPs. Another study with extreme bundling in PCPs [Hei+12] showed that the perception of data characteristics and correlation was still possible when participants could not use the center parts between axes. These findings give us an opportunity to address the aforementioned problems: we replace straight lines with composite connections (see Figure 9.8). The key to our model is that we start and finish the connections with short straight line segments with the same slope as in original PCPs. This creates a zone in the very important focal area around the axes that retains the same information as in regular PCPs.

In the center region—in-between data axes—we connect the straight segments with a cubic Hermite curve. Choosing the tangents of the curve to be identical to the original slope guarantees a smooth transition between segments. The result of this approach is shown in Figure 9.7b. Instead of using the same factor to scale the Hermite curve’s tangents $T1$ and $T2$, we vary it between data points. The relative index of each row in the source data set is added to a base factor on $T1$ and subtracted for $T2$. This spreads the composite lines along the direction they would have had in a classic PCP. This is very helpful when there are multiple rows with the same values in both neighboring data dimensions: the wider the lines spread, the more rows share the same values. The effect can be observed in the curves between the dimensions *Oil* and *Biomass* in Figure 9.15. In regular PCPs, this information would be lost in overplotting or would require a density rendering technique.

9.4.2 Density Rendering

CF-PCPs are designed to work with large data sets that can incur the problem of overplotting. We address this issue by adopting the established method of splatting [Zho+09] the lines and showing their density [AOL04; Joh+05]. This is achieved by splitting the rendering process into separate passes. The first pass uses additive blending to compute an intermediate density field. Since the densities typically cover a large dynamic range, we apply a nonlinear transformation before we render the final visualization. In our implementation, we apply a logarithmic mapping together with a logistic function to guarantee user-specified extrema within $[0, 1]$. The result is then multiplied by the line color’s alpha value. The result is best demonstrated in Figure 9.1, where individual lines can be traced in areas

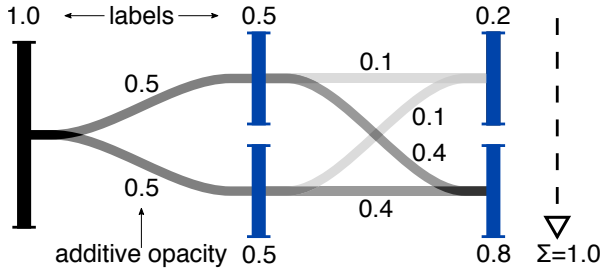


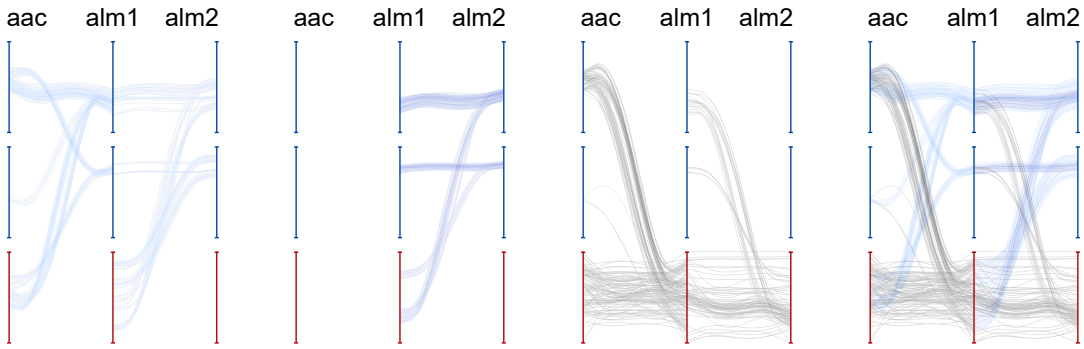
Figure 9.9: Fuzzy clustering provides a single data point with multiple soft labels in various clusters. Our technique draws lines between all possible pairs of these clusters. Their opacity is calculated by multiplying the point's labels in both connected clusters. This spreads each data point's constant total density of 1 over all possible connections.

of both high and low density. We discuss the choice of color in the following context of uncertainty visualization.

9.4.3 Uncertainty

Fuzzy clustering involves the soft labeling of individual data points. In our implementation, we distinguished between no clusters at all (black axes in Figure 9.10), noise (red), and actual clusters (blue). This gives viewers additional information about the underlying algorithm's success in clustering the source data. Going further, more levels of certainty could be visualized as long as the axis colors remain well distinguishable.

Soft clustering can also select multiple labels for a single point, weakly assigning it to multiple clusters instead of a single strong result. We treat these labels as probabilities, and, by definition, the total probability that a point exists is 100 %, i.e., the sum of all assigned soft labels has to be 1. The clustering technique does not supply information on the flow between sets of fuzzy clusters in separate subspaces. Therefore, we cannot infer whether a point moved between specific pairs of neighboring clusters. The probability that it belongs to one cluster or the other is our only information. We take this into account by rendering a single source data point as multiple lines to obtain a faithful visual representation. When a data row is labeled more than once in a 2D subspace, we draw a line between all neighboring clusters it belongs to (see Figure 9.9). We determine the opacity of each line by multiplying the label values of the clusters it connects. Our method corresponds to providing a field with probable paths for the movement of data points and results in an invariant total opacity of 1 for each point. This property is compatible with the rendering technique from Section 9.4.2 because it does not affect the total density.



(a) Low **(b)** Medium **(c)** High **(d)** Combined

Figure 9.10: Visualizing uncertainty in the clustering results. The label weights are binned into three ranges. Each bin is separately rendered to a density field and then mapped to color (c)–(a). Finally, the results are sorted by uncertainty and alpha blended (d).

Up to this point, our approach only renders the probability that a data point flows from one cluster to the other. Color is a strong visual variable with many distinguishable values, but we have not used it for line rendering yet. Hence, CF-PCPs encode the numerical uncertainty from the clustering algorithm by mapping it to color. Each line between two axes is rendered according to its underlying data point's label in the target cluster (right axis for reading direction LTR). This means that the color of lines for the same data row may change from one side of an axis to the other, leaving the positional attributes as hints for tracing the line paths. We choose a sequential color scale (from light blue to black) to map the label weights in our examples (see Figure 9.10). The gradient from light blue to black only varies significantly in saturation and value, making it suitable even for viewers with the most common color vision deficiencies.

CF-PCPs apply binning for good readability [Pad+17] because smooth color gradients can be problematic for reading exact values [Bre96]. Another benefit of binning becomes evident when creating the density fields (see Section 9.4.2): additive blending would mix and distort the used colors. Furthermore, it would make multiple overlaid fuzzy lines look like a single certain one. To address these issues, our approach creates a separate density field for each bin in the uncertainty color map. As depicted in Figure 9.10, CF-PCPs convert them separately into regular images with the RGB values from the color map, while the alpha channel encodes the field's density. In the final render pass, these individual fields are sorted by fuzziness and alpha-blended with the *over* operator [PD84].

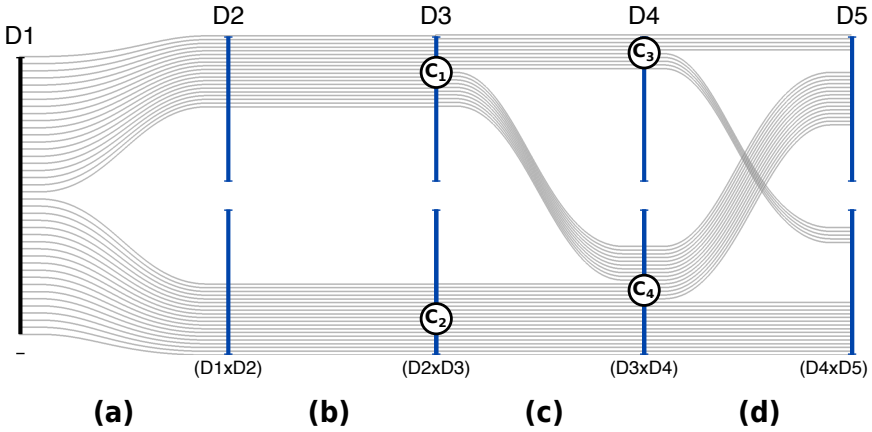


Figure 9.11: Inter-cluster patterns. Clustering in subspaces $D1 \times D2$ and $D2 \times D3$ gives the exact same result ((a), (b)). Cluster C_1 splits into C_3 and C_4 in $D3 \times D4$ (c). The displayed clusters in (d) have very low similarity with the results from the previous subspace.

9.5 Visual Patterns

As discussed earlier, CF-PCPs show streams of 2D clusters across the subspaces of a data set while still displaying the original correlations between dimensions. As such, our visualization aims at providing two levels of granularity: overview and detail. The overview level shows similar 2D subspaces, clusters, and inter-cluster patterns, i.e., how data flows between clusters across different subspaces. The detail level shows intra-cluster patterns, i.e., the correlation within pairs of clusters from neighboring dimensions.

Ordering: Our horizontal ordering, coupled with our A* approach, implies strong similarities of clusters in neighboring dimensions because we minimize the total dissimilarity according to our metrics. More specifically, this means that the two subspaces between three neighboring dimensions contain similar clusters.

Clusters: The duplication of axes allows for easy identification of the number of clusters within each 2D subspace. In turn, this helps us identify data classes across different pairs of dimensions. Coupled with soft labeling, this makes it easy to recognize well-separated classes (with a high level of certainty) over fuzzy ones.

Inter-Cluster Patterns: We created Figure 9.11 as an example with five dimensions that always have two clusters in their subspaces. In Figures 9.11a and 9.11b, the flow of data between clusters in $D1 \times D2$ and $D2 \times D3$ shows highly similar subspaces: each cluster on the left is associated with a unique cluster on the right. These one-to-one relationships

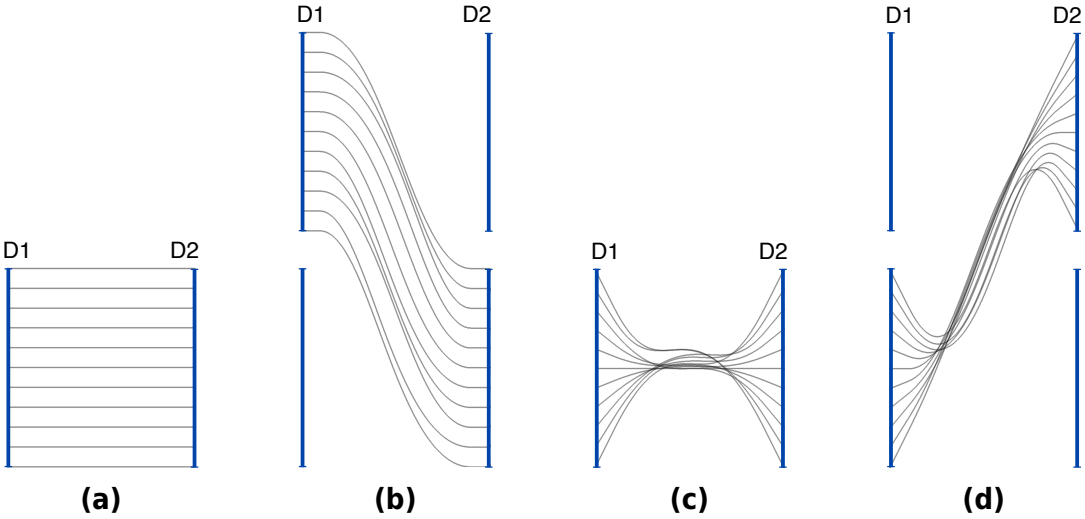


Figure 9.12: Intra-cluster patterns. When CF-PCP axes are at the same height ((a) and (c)), patterns from positive and negative correlations look very similar to their counterparts in regular PCPs. The line geometry discussed in Section 9.4 preserves these patterns close to the axes, even if they are at different heights ((b) and (d)).

denote high or total similarity between subspaces. The situation is different in Figure 9.11c, where the CF-PCP shows that cluster C_1 from $D2{\times}D3$ splits into C_3 and C_4 in $D3{\times}D4$. It would also be correct to say that C_4 splits into C_1 and C_2, depending on the point of view. Finally, in Figure 9.11d, we see two very dissimilar subspaces. Here, both clusters C_3 and C_4 split and mix in $D4{\times}D5$.

More generally, if each cluster in the subspace of two dimensions is uniquely associated with every other cluster of a neighboring subspace (i.e., the similarity matrix in Equation (9.3) contains only ones and zeros), it means that they behave similarly. If, in turn, all clusters are linked with each other (i.e., the similarity matrix contains many values <1 and >0), this shows a completely dissimilar clustering behavior between the three dimensions that span the two subspaces. Additionally, an axis having different incoming clusters is equal to a merge or split, depending on the point of view. Overall, in CF-PCPs, the number of splits or merges between clusters conveys the connectedness of their two subspaces.

Intra-Cluster Patterns: Atop showing the high-level information on the flow of data between clusters across subspaces, our visualization also maintains patterns known from traditional parallel coordinates. Since the duplication retains the entire value range of an axis, our cluster-flow layout shows the distribution of each cluster's data points in the original dimensions. As demonstrated by the clusters labeled $D3$ in Figure 9.1, we can see

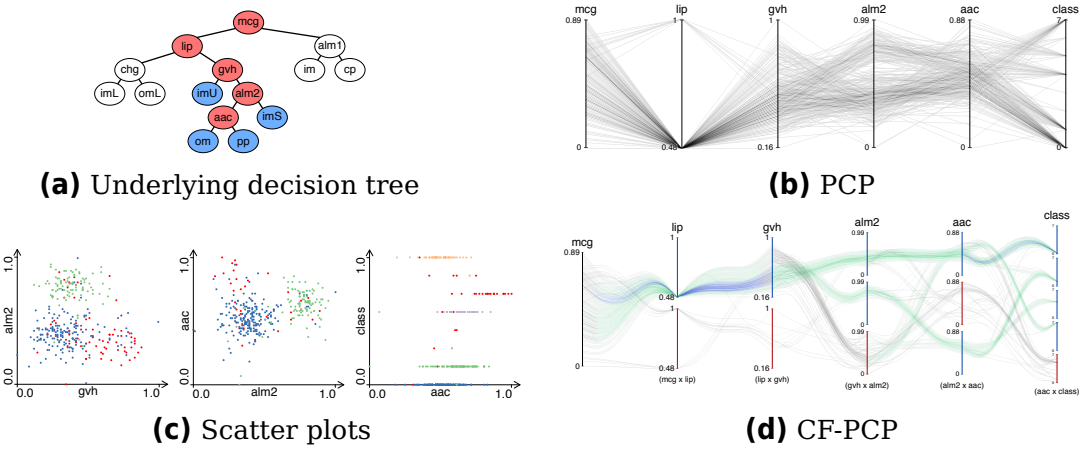


Figure 9.13: Classification of E. coli bacteria according to a tree (a) where nodes correspond to metrics and leaves to inferred classes [HN96]. All red metrics are necessary to distinguish the blue classes. These are shown as dimensions in the regular PCP (b). The scatter plots show color-coded fuzzy subspace clusters between the last three metrics (c). Our CF-PCP (d) technique combines aspects of both previous visualizations. Its line colors encode certainty: 0 100 %.

where the values are positioned in regard to the whole range: larger values are in the upper cluster, and smaller ones are in the lower. This approach also visualizes whether the data points move closer or further apart, e.g., the bundle from $D3$ spread out in $D4$.

The examples in Figure 9.12 illustrate how CF-PCPs show data correlations. If two neighboring clusters are aligned ((a) and (c)), then positive and negative correlations look very similar to the classic PCP approach. In the case of non-aligned clusters ((b) and (d)), our line geometry from Section 9.4 ensures that the slopes near the axes remain identical to the ones in regular PCPs. Hence, our approach preserves the most important areas of PCPs [Net+17] and adds additional information on clusters and data flow.

Overall, the visual patterns of our approach can be seen in different places of the visualization: inter-cluster patterns are located in the entire space between two display dimensions, whereas the vicinity of duplicated axes reveals intra-cluster patterns.

9.6 Examples

To demonstrate the applicability of CF-PCPs, we provide examples for typical real-world data sets and compare our results to previous approaches.

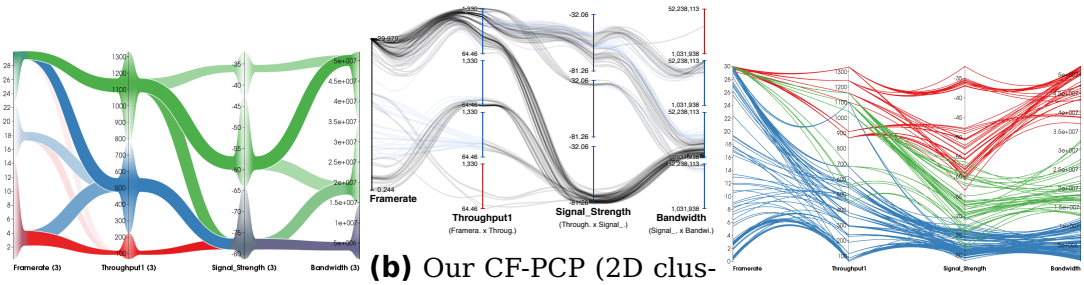
9.6.1 Escherichia Coli

Horton and Nakai used machine learning to automatically predict the localization sites of proteins [HN96]. Their results included a decision tree that uses multiple measured metrics for Escherichia coli bacteria in order to arrive at a predicted classification. Figure 9.13a shows this tree and highlights five red nodes that we selected for visualization. They correspond to the metrics that define four classifications (blue). We start our analysis with a regular PCP and density rendering (Figure 9.13b). It shows that the dimension *lip* is only binary. There seem to be negative correlations between *alm2* and its neighbors, but it is difficult to map the resulting class to a specific influence of each previous dimension. The CF-PCP in Figure 9.13d uses 2D subspace clustering and shows additional information. The first impression is of the mostly green color, which shows low certainty. This is due to a low degree of separation between the clusters. The second impression is of the flow between clusters. The observable flow from the CF-PCP matches the tree structure from Figure 9.13a. Using Fuzzy DBSCAN with dimensions *mcg*, *lip*, and *gvh* yields a single cluster and noise each time. The classification tree confirms that they are not sufficient to infer any specific classes. Subspace $gvh \times alm2$ shows the first split into two actual clusters, just as the tree also arrives at its first class *imU*. Class *imS* is inferred next, and the CF-PCP also contains two clusters in $alm2 \times aac$. Combining the information from $aac \times class$ reveals four final clusters, which match the classification count in the highlighted subtree.

The scatter plots with color-coded clusters do not show this data flow between dimensions because the selected colors are not linked between each plot. Even the display of the fuzziness poses a problem: some points belong to multiple clusters. Plotting them multiple times at the same position only retains the color of their last label.

9.6.2 NetPerf

To help understand the characteristics of CF-PCPs, we also compare our technique with two clustering-oriented PCP approaches in Figure 9.14: IPCs [MM08] and an edge-bundling layout (EBL) for PCPs [Pal+14]. All visualizations are of the NetPerf data set [War05]. In contrast to our proposed pairwise fuzzy method, the clustering in IPCs is calculated and rendered for all dimensions at once. Conversely, the EBL method clusters the data separately in every single dimension. Both approaches use color to encode clusters consistently over every dimension. Our selected sample visualizations show four dimensions and three clusters in the NetPerf data set. To increase comparability, we limit ourselves to the four dimensions



(a) EBL (1D clustering) **(b)** Our CF-PCP (2D clustering) **(c)** IPCs (nD clustering)

Figure 9.14: Comparison of different clustering and PCP techniques using the NetPerf data set [War05]. The EBL (a) replaces lines entirely and only draws a single band between each pair of neighboring 1D clusters [Pal+14]. Our CF-PCPs (b) draw individual lines between 2D clusters of neighboring axes. Clusters are arranged vertically with our crossing minimization algorithm from Section 9.3.3. IPCs (c) emphasize clusters over all dimensions using colored lines and force-based bundling [MM08]. Figures (a) and (c) © 2014 IEEE. Reprinted, with permission, from [Pal+14].

used in the previously published visualizations of the same data. Dimension ordering is disabled, while the vertical axis order minimizes our metric of line crossings from Section 9.3.3.

In Figure 9.14c, IPCs bundle PCP polylines based on the cluster they belong to. This reduces inter-cluster overdraw and emphasizes visual cluster separation. Line distortion is applied at the expense of clarity of pairwise axis correlation because their direction no longer corresponds to the original angle in regular PCPs. Clustering in IPCs works across all visible dimensions and shows the overlap between clusters, for instance, in the signal strength dimension. In contrast, the EBL [Pal+14] reduces clutter in PCPs by bundling edges within pairs of 1D clusters that are computed with a k-means algorithm. Similarly to our approach, this method highlights subspace clusters and allows viewers to follow the flow of bundles between dimensions. As such, the method avoids overdrawing of clusters on axes. However, the 1D approach to clustering simplifies the visualization to the detriment of details when it comes to diverging clusters. For example, edges in the top cluster of the throughput axis split into three different clusters in signal strength, but without further interaction, users cannot infer the distribution of data points that belong to the highest cluster in the signal strength dimension.

Similarly to both alternative approaches, our cluster-flow layout shows three main clusters per axis pair (see Figure 9.14b). However, our visualization goes beyond the alternative techniques and is able to show overlap between clusters while allowing the viewer to trace complex cluster pat-

terns. More specifically, our plot shows that low frame rates are always associated with low throughput. Just as with logical consequence, the opposite is *not* true. Similarly to IPCs (Figure 9.14c), the CF-PCP shows that the main cluster of high throughput is associated with three clusters of high to low values of signal strength. Contrary to the broad bands in EBLs, fine details in CF-PCPs allow the viewer to see that the highest values of signal strength are always associated with the highest values of throughput. The large cluster at the top of the throughput dimension in the EBL of Figure 9.14a does not support this conclusion.

A commonality with the EBL is the top cluster between signal strength and throughput, which forks into two different clusters when combining the data on signal strength and bandwidth: one with high and the other with low values. In the CF-PCP, we can quickly determine that there are many data points with high framerate, despite having only medium to low throughput and minimal signal strength and bandwidth. Color coding fuzziness, we are also able to see that there are uncertain assignments, e.g., the third cluster between frame rate and throughput is completely fuzzy, and many of its data points are more likely to be noise or belong to the second cluster. Overall, our implementation avoids the inconvenience of overlap from n-dimensional clustering and allows for detailed analysis by tracing individual lines and clusters across dimensions.

9.6.3 Energy Production

Lastly, we visualize a larger data set of electricity production by primary energy source [Fra19; Rod+17b]. The current data set is more complex than the one used in Part I (only information about the percentage of renewable energy). It contains 1750 data rows and lists a timestamp, 12 sources of primary energy, as well as international imports and exports as dimensions we can use in the CF-PCP. While the restriction to subspaces generally reduces the number of clusters, this is not the case with the uniformly distributed data in the time dimension: it yields 12 clusters. We removed it to get a cleaner and less cluttered plot for further analysis. We provide further visualizations of the original and reduced data set in the supplemental material of the original publication [Rod+20].

Clustering between pairs of dimensions and separating them vertically can help with the visual detection of dependencies in the data. From Figure 9.15, we can extract information between hard and brown coal. On the one hand, low energy production from the latter only occurs when hard coal also has lower values. On the other hand, power production on the left-hand side varies greatly, while brown coal is often at a high output level. Therefore, it seems that between these two, brown coal is

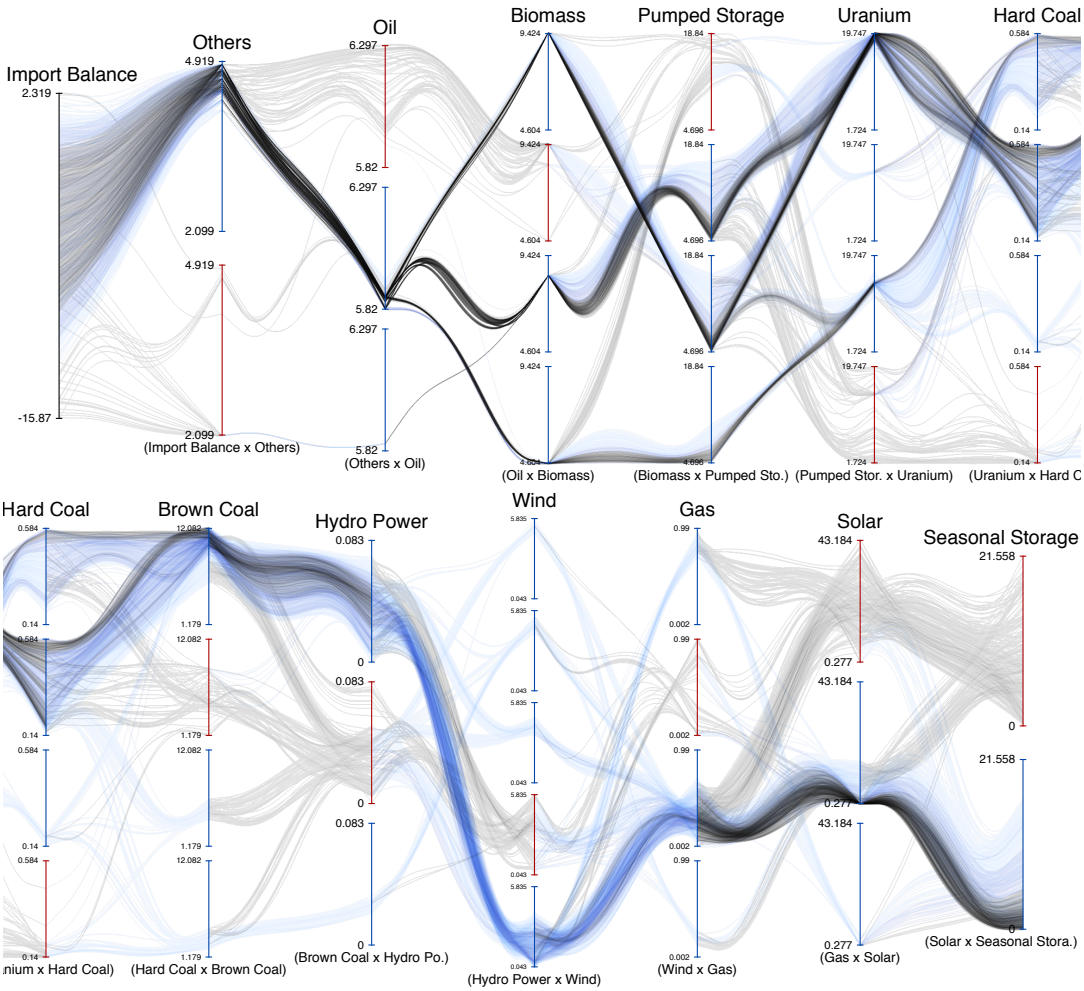


Figure 9.15: CF-PCP of electric power production by primary energy source. Clusters are vertically ordered by their mean value. Line colors encode certainty from fuzzy clustering: 0 100 %. To show more detail, we enlarged the plot and split it into two lines.

burned more constantly while hard coal is preferred for adjustments to changing levels in power production from unsteady sources. Even with density rendering, the high degree of overplotting would create an almost constant background between both axes and thus cannot facilitate these observations.

9.7 Discussion

We consider CF-PCPs to be an answer to RQ3. They build on a solid foundation of regular PCPs with density-based rendering and incorporate the advantages of a sequence of animated scatter plots from a SPLOM. The

main idea of our technique is to deliberately duplicate axes for each cluster to show the data flow between 2D subspaces. At the same time, this also creates an opportunity to display fuzzy clusters with parallel coordinates.

A SPLOM can be explored in a visual analytics application, and we have shown in Chapter 7 how eye tracking supports the viewer in navigating the space of possible dimension pairs. But we designed CF-PCPs to be static, with no need for user interaction. Therefore, we developed a metric and algorithm that sorts the data dimensions in order to optimize the visible data flow between 2D subspace clusters. The resulting plot allows tracing cluster interactions and individual points, similar to the tasks that we used to evaluate animated transitions between scatter plots (see Chapter 6). Although not originally intended, manual interaction with the axes and dimensions remains a viable alternative for interactive data exploration. This could be helpful in the contexts of model peeling, regression, and machine learning.

Furthermore, our layout is an improvement over traditional PCPs when clusters are not linearly separable over a single dimension or all dimensions together, and the number of clusters in subspaces is small. We implemented CF-PCPs as a web-based visualization for RQ3 using vector graphics on top of rasterized images. The source code is available online [Rod20; Sch18].

Extensions of our work might look into hierarchical clustering, similar to the example in Figure 9.13 with data about Escherichia coli. Here, we would enforce a reading direction and progressively create subclusters that mimic the classification tree from Horton and Nakai's work [HN96].

Our goal of visualizing 2D subspaces led to a reading direction in CF-PCPs. But the technique could easily be extended to a symmetrical clustering with both neighboring axes—or any odd number of data dimensions. Another change in the spirit of RQ3 could be to cluster all displayed dimensions with a common reference dimension. This would be similar to selecting a row or column from a scatter plot matrix and visualizing it with parallel coordinates.

10

Conclusion

With this thesis, we wanted to improve knowledge gain by adapting existing concepts to extend their applicability and widen their target audiences. Visualization is ubiquitous, and there are more people who use simple representations of relatively small data than workers analyzing big data. To reach a large audience, we focused on relatively simple concepts and limited ourselves to plots that only show a single data dimension or combine at most two attributes in each part of the visualization. We discussed different types and examples of data and how they could be visualized in 2D. We formulated three research questions that fit the presented data and visualization types that helped us come closer to the overarching goal.

10.1 Answers to Research Questions

We will now summarize how we approached the research questions and put the results of this thesis into perspective.

How to scale unit visualization for 1D data? In Part I, we adressed question RQ1. First, we extended the applicability of dot plots to larger high-dynamic-range data sets in Chapter 3. The result was a layout that could represent outliers and give an indication of data frequency alike. It allowed dot plots to be used in cases where previously nonlinear histograms would be the visualization of choice. The countable nature of dots even has advantages over the bars in histograms. With high-frequency data, however, the plots exhibit moiré effects from many small circles in close proximity. The displayed data frequency in dense areas was ragged, similar to the effect caused by dynamic binning that becomes too narrow.

Chapter 4 presented a second step, where we removed the constraint of straight columns for dot plots. This had three advantages: it removed a

prerequisite for viewers to understand the visualization, it allowed better control of the outer shape that indicates data frequency, and it enabled us to address the issue of moiré effects. First, we had to find an outer envelope shape. To this end, we performed bounded kernel frequency estimation with a bandwidth based on dot size. Using mathematical transformations (detailed in a later excursion in Chapter 5 to not interrupt the flow of text), we derived new scaling functions for the size of dots and the height of traditional dot columns from the underlying data frequency. This allowed us to scale the envelope to fit the included dots tightly and represent the data density without narrow spikes. By adapting Lloyd relaxation, we were able to distribute the circles with less positional error than in column-based dot plots and avoided moiré effects from regular patterns. We confirmed the benefits of the relaxed layout over the nonlinear variant with a crowdsourced user study.

In summary, we provided an answer to  RQ1 by adapting dot plots. The new variant not only supports data sets with larger sample counts but also improves the visualization by making it more faithful to the underlying values and more aesthetically pleasing. In the context of Richer et al.'s work on scalability in visualization [Ric+23], the new technique is suitable for an increased problem size S (i.e. sample count), loosens assumptions A about the viewers (i.e. requires less previous knowledge), and reduces the necessary effort E through visibility of outliers and a more faithful data representation. To achieve this, we had to loosen the definition of dot plots from page 19: They no longer use constant diameters (rule **DP2**) nor straight columns (**DP4**) but remain a unit visualization (**DP1**) with higher positional correctness of all dots (**DP3**).

How to improve navigation between 2D visualizations of multivariate data? We addressed question  RQ2 in Part II, where we used visualizations with two data dimensions from multivariate data sets. While switching between different views, users can easily lose their mental map of the data. Animation can help in such a context by allowing them to trace points and clusters through the transition between plots. In Chapter 6, we evaluated animations with respect to user performance and identified that orthogonal rotation works best for tracing points in scatter plots. There was no significant difference for tracing cluster interactions. To help navigate SPLOMs, we recommended always animating the switches between cells because it does not take much time but provides important benefits for the viewers.

Animated transitions between plots are beneficial, but to fully navigate and explore multivariate data, it is also necessary to know which subspaces

are relevant. In Chapter 7, we used eye tracking to identify the data of interest to the user and applied a range of (dis)similarity metrics to compare the same data in all cells of the SPLOM. The results served as recommendations for the viewer to find the dimension pairs that fit their analysis task. The presented proof of concept was generic and adapted to arbitrary user tasks.

Animation and recommendations together answer RQ2 and help humans in the combined use of scatter plots and SPLOMs. They are tools to analyze multivariate data while only using two dimensions at a time. As such, they fit the overarching goal by extending the number of data dimensions that can be analyzed without needing to learn new and complicated visualization techniques.

Having seen the benefits of eye tracking in 2D visual analytics, we went on an excursion to explore its use for immersive and situated analytics. Chapter 8 presented an existing VR tool and discussed hypothetical benefits and challenges of added support for eye tracking. Information about 3D gaze positions could be especially helpful in immersive environments and for situated analytics. However, previous work showed that the subjective depth of virtual objects differed from real ones. Using a see-through HMD, we analyzed whether objective measurements of gaze depth with binocular eye tracking were also affected. We did not find a significant difference in measured 3D gaze but found too much noise in the depth component for reliable use without preprocessing. The tested setup was better suited for situated 2D visualization than for the 3D variant.

How to combine the advantages of multiple 2D views in a single static visualization for multivariate data? In Part III, we approached question RQ3. First, we briefly introduced the concept and a selection of rendering techniques for parallel coordinates plots. Then, in Chapter 9, we developed the cluster-flow parallel coordinates plot as a visualization that extended the concept of PCPs by duplicating and stacking axes. It showed correlations near the axes and 2D subspace clusters in the space between neighboring data dimensions. Custom density rendering enabled tracing individual data points over all axes and showed how clusters interacted. The static layout could be used with non-interactive media, e.g., in print, and was calculated automatically to emphasize data flow between clusters, leading to support but no requirement of manual intervention.

Thus, CF-PCPs provide an answer to RQ3 by duplicating axes in PCPs for each 2D subspace data cluster. The technique combines advantages of PCPs (data correlations) with the benefits of animated scatter plots (cluster separability and interactions) while retaining the traceability of

individual data points. They even exceed the requirements by supporting the visualization of soft clusters. The automated sorting of dimensions and axes supports users in the exploration of multivariate data.

Availability of Results In this thesis, we present the necessary theory to answer the research questions. All practical implementations and study data are also available:

- The core algorithm [Rod17] and an R package [DKF23] for nonlinear dot plots.
- A *D3.js* plugin [DBR22], study data, mathematical transformations, and additional figures [Rod+22b] for relaxed dot plots.
- A *D3.js* plugin for generic animations between scatter plots [BR23], study data (including statistical evaluation) [Rod+23a], and a demonstration of the study tasks [Rod23].
- The proof-of-concept implementation for scatter plot recommendations from eye tracking [Yan+22].
- A web-based implementation for the creation of cluster-flow parallel coordinates plots [Rod20].

We advocate open and reproducible science. We publish our source code and data for the benefit of researchers that might want to build upon our work.

10.2 Future Directions

We tackled the overarching goal of this thesis from two perspectives: extending visualization techniques to make them applicable to more data and widening the target audiences. Our research questions focused on specific topics, but the general approach of extension and widening can be taken further.

Wider Audience Accessibility is not only an issue for mobility but also knowledge. There is already an active and thriving research community that identifies potential problems and develops solutions for accessible visualization. But there is still much work ahead as the general task is challenging [Lee+20; LLS19]. Their goals overlap with the one in this dissertation: if more users can benefit from a visualization, we widen the target audience. For example, there are hundreds of millions of people with color vision deficiencies [Lee+20; Ang+22]. Visualization of 1D data with dot plots does not need color and can be printed as physical objects, making it accessible to blind users. Instead of encoding a secondary data attribute with color, different dot shapes could be used. Generally, simpler

plotting techniques reach wider audiences and should not be overlooked in future research.

Larger Data Data sets grow ever larger as the available capacity for storage and computation increases. It is important to investigate how the visualization of multivariate data scales to millions of samples or even more. For example, it would be possible to extend the approach from Part II with a density-based rendering of scalar fields [vd03] or *Splatterplots* [MG13]. It would be more difficult to discern and trace individual data points in such plots. But existing techniques from volume rendering could serve for rotation transitions as in Chapter 6. Flat animation of scalar fields would continue to show cluster interactions. The eye-tracking approach in Chapter 7 could be extended to soft data selection without hard thresholds. User interest would be attributed to data by calculating the intersection of Gaussian distributions centered on data points and gaze positions.

Empirical Evaluation Extending the audience and the applicability to data are not the only directions for future developments. Research is generally driven by the motivation to improve the status quo. But how can we decide whether the introduced changes actually achieve the desired improvement? In Part II, we tested the various animations empirically in a crowdsourced user study. But we did not have a finished product for the eye-tracking-based recommendations to evaluate with experts and novice users. We ourselves saw benefits from CF-PCPs, but how difficult would it be for others to interpret the visualization? The *grand tour* was a nice technique but never proved how well new users could learn it [Asi85]. Future work could encompass the replication and evaluation of the techniques presented here.

We see a need for empirical evaluation with all new developments. It should not be a prerequisite for the publication of research. That would hinder the dissemination of ideas. Evaluation can take place after the publication of techniques and is more trustworthy if performed by third parties, as it is customary in other fields of research. Such an approach would encourage authors to provide all essential information to replicate results. Our animations in Chapter 6 also included a concept by Elmqvist et al. [EDF08]. But replication is generally not seen often in the field of visualization and is more challenging to publish at high-tier venues.

Explainability Evaluation works best with large sample sizes but the number of people that understand overly convoluted plots is probably small. Simple concepts in visualization are important and necessary, es-

pecially as research in the field evolves. The rate at which visualization techniques can become more complex is much higher than the rate at which human cognitive abilities can, if at all, grow. Computational approaches to simplifying data, e.g., dimensionality reduction and deep learning, often lack explainability. But how can humans rely on visualization when they have no understanding and trust in the underlying algorithms? Visualizing ever larger and more complex data with simple and widely understood methods will be increasingly challenging. Explainable AI is an active field of research. As the used models and concepts grow in complexity, so tend the visual depictions thereof. How can a few images represent millions of neurons and convolutions matrices effectively? Such research already is and will probably be a big challenge. To really achieve explainability, visualizations in these contexts should focus on simple concepts, possibly also limited to points and lines as drawing primitives.

Bibliography

[ABK98] M. Ankerst, S. Berchtold, and D. A. Keim. “Similarity clustering of dimensions for an enhanced visualization of multidimensional data.” In: *Proceedings of the IEEE Symposium on Information Visualization*. 1998, pp. 52–60. DOI: 10.1109/INFVIS.1998.729559.

[Ang+22] K. Angerbauer, N. Rodrigues, R. Cutura, S. Öney, N. Pathmanathan, C. Morariu, D. Weiskopf, and M. Sedlmair. “Accessibility for color vision deficiencies: challenges and findings of a large scale study on paper figures.” In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2022. DOI: 10.1145/3491102.3502133.

[AOL04] A. O. Artero, M. C. F. de Oliveira, and H. Levkowitz. “Uncovering clusters in crowded parallel coordinates visualizations.” In: *Proceedings of the IEEE Symposium on Information Visualization*. 2004, pp. 81–88. DOI: 10.1109/INFVIS.2004.68.

[Arb+17] C. Arbesser, F. Spechtenhauser, T. Mühlbacher, and H. Piringer. “Visplause: Visual data quality assessment of many time series using plausibility checks.” In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 641–650. DOI: 10.1109/tvcg.2016.2598592.

[Asi85] D. Asimov. “The grand tour: A tool for viewing multidimensional data.” In: *SIAM Journal on Scientific and Statistical Computing* 6.1 (1985), pp. 128–143. DOI: 10.1137/0906011.

[Aur91] F. Aurenhammer. “Voronoi diagrams—A survey of a fundamental geometric data structure.” In: *ACM Computing Surveys* 23.3 (1991), pp. 345–405. DOI: 10.1145/116873.116880.

[Bak04] A. Bakker. “Design Research in Statistics Education: On Symbolizing and Computer Tools.” PhD thesis. Utrecht University, 2004.

[BDJ14] R. Borgo, J. Dearden, and M. W. Jones. “Order of magnitude markers: An empirical study on large magnitude number detection.” In: *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 2261–2270.

[Beh+14] M. Behrisch, F. Korkmaz, L. Shao, and T. Schreck. “Feedback-driven interactive exploration of large multidimensional data supported by visual classifier.” In: *Proceedings of the Conference on Visual Analytics Science and Technology*. IEEE, 2014, pp. 43–52. DOI: 10.1109/vast.2014.7042480.

[BGS01] P. Baudisch, N. Good, and P. Stewart. “Focus plus context screens.” In: *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, Nov. 2001. DOI: 10.1145/502348.502354.

[BH21] T. Bafna and J. P. Hansen. “Mental fatigue measurement using eye metrics: A systematic literature review.” In: *Psychophysiology* 58.6 (2021). DOI: 10.1111/psyp.13828.

[Bla+17] T. Blascheck, K. Kurzhals, M. Raschke, M. Burch, D. Weiskopf, and T. Ertl. “Visualization of eye tracking data: a taxonomy and survey.” In: *Computer Graphics Forum* 36.8 (2017), pp. 260–284. DOI: 10.1111/cgf.13079.

[BOH11] M. Bostock, V. Ogievetsky, and J. Heer. “D^3 Data-driven documents.” In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (2011), pp. 2301–2309. DOI: 10.1109/TVCG.2011.185.

[BR23] V. Brandt and N. Rodrigues. *D3-plugin for animated scatter plot transitions*. https://github.com/NilsRodrigues/d3-scattertrans, accessed 2023-05-06. 2023.

[Bre+20] M. Brehmer, B. Lee, P. Isenberg, and E. K. Choe. “A Comparative Evaluation of Animation and Small Multiples for Trend Visualization on Mobile Phones.” In: *Transactions on Visualization and Computer Graphics* 26.1 (2020), pp. 364–374. DOI: 10.1109/tvcg.2019.2934397.

[Bre96] C. A. Brewer. “Prediction of simultaneous contrast between map colors with Hunt’s model of color appearance.” In: *Color Research and Application* 21.3 (1996), pp. 221–235.

[Bur+14] M. Burch, S. Lohmann, F. Beck, N. Rodrigues, L. D. Silvestro, and D. Weiskopf. “RadCloud: visualizing multiple texts with merged word clouds.” In: *Proceedings of the 18th International Conference on Information Visualisation*. IEEE, 2014. DOI: 10.1109/iv.2014.72.

[Cal+09] J. A. Caldwell, M. M. Mallis, J. L. Caldwell, M. A. Paul, J. C. Miller, and D. F. Neri. “Fatigue countermeasures in aviation.” In: *Aviation, Space, and Environmental Medicine* 80.1 (2009), pp. 29–59.

[Cao+23] Y. Cao, J. L. E, Z. Chen, and H. Xia. “DataParticles: Block-based and language-oriented authoring of animated unit visualizations.” In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2023. DOI: 10.1145/3544548.3581472.

[CDF14] F. Chevalier, P. Dragicevic, and S. Franconeri. “The Not-so-Staggering Effect of Staggered Animated Transitions on Visual Tracking.” In: *Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 2241–2250. DOI: 10.1109/tvcg.2014.2346424.

[Cha+83] J. M. Chambers, W. S. Cleveland, B. Kleiner, and P. A. Tukey. *Graphical Methods for Data Analysis*. Belmont, California, USA: Wadsworth, 1983. DOI: 10.1201/9781351072304.

[Che+21] J. Chen, M. Ling, R. Li, P. Isenberg, T. Isenberg, M. Sedlmair, T. Möller, R. S. Laramee, H.-W. Shen, K. Wunsche, and Q. Wang. “VIS30K: a collection of figures and tables from IEEE visualization conference publications.” In: *IEEE Transactions on Visualization and Computer Graphics* 27.9 (2021), pp. 3826–3833. DOI: 10.1109/tvcg.2021.3054916.

[Chi+22] F. Chiossi, J. Zagermann, J. Karolus, N. Rodrigues, P. Balestrucci, D. Weiskopf, B. V. Ehinger, T. M. Feuchtner, H. Reiterer, L. L. Chuang, M. O. Ernst, A. Bulling, S. Mayer, and A. Schmidt. “Adapting visualizations and interfaces to the user.” In: *it - Information Technology* 64.4&5 (2022), pp. 133–143. DOI: 10.1515/itit-2022-0035.

[Cle85] W. S. Cleveland. *The Elements of Graphing Data*. Monterey, California, USA: Wadsworth, 1985.

[Cle93] W. S. Cleveland. *Visualizing Data*. Summit, New Jersey, USA: Hobart Press, 1993.

[CM85] W. S. Cleveland and R. McGill. “Graphical perception and graphical methods for analyzing scientific data.” In: *Science* 229.4716 (1985), pp. 828–833. DOI: 10.1126/science.229.4716.828.

[CM88] W. S. Cleveland and M. E. McGill. *Dynamic Graphics Statistics*. Pacific Grove, California, USA: Wadsworth & Brooks/Cole, 1988.

[CMS99] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. San Francisco, California, USA: Morgan Kaufmann Publishers Inc., 1999.

[DBR22] S. Döring, D. Baumgartner, and N. Rodrigues. *Relaxed dot plots*. https://github.com/NilsRodrigues/relaxeddotplots, accessed 2022-07-19. 2022.

[Deu] Deutscher Wetterdienst. *Climate data centre - CDC*. https://opendata.dwd.de/climate_environment/CDC/observations_global/CLIMAT/monthly/qc/air_temperature_mean_of_daily_max/, accessed 2023-05-20.

[DF15] S. Drucker and R. Fernandez. *A Unifying Framework for Animated and Interactive Unit Visualizations*. Tech. rep. MSR-TR-2015-65. 2015.

[DK10] A. Dasgupta and R. Kosara. “Pargnostics: Screen-space metrics for parallel coordinates.” In: *IEEE Transactions on Visualization and Computer Graphics* 16.6 (2010), pp. 1017–1026. DOI: 10.1109/TVCG.2010.184.

[DKF23] S. Döring, M. Kubica, and I. Fourati. *nonLinearDotPlot: Non Linear Dot Plots*. R package version 0.5.0, https://CRAN.R-project.org/package=nonLinearDotPlot, accessed 2023-05-20. 2023.

[DM96] D. Drascic and P. Milgram. “Perceptual issues in augmented reality.” In: *Stereoscopic Displays and Virtual Reality Systems III*. Ed. by M. T. Bolas, S. S. Fisher, M. T. Bolas, S. S. Fisher, and J. O. Merritt. Vol. 2653. International Society for Optics and Photonics. SPIE, 1996, pp. 123–134. DOI: 10.1117/12.237425.

[DSZ17] O. Deussen, M. Spicker, and Q. Zheng. “Weighted Linde-Buzo-Gray stippling.” In: *ACM Transactions on Graphics* 36.6 (2017), 233:1–233:12. DOI: 10.1145/3130800.3130819.

[Du+15] F. Du, N. Cao, J. Zhao, and Y.-R. Lin. “Trajectory Bundling for Animated Transitions.” In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2015. DOI: 10.1145/2702123.2702476.

[Duc+11] A. T. Duchowski, B. Pelfrey, D. H. House, and R. Wang. “Measuring gaze depth with an eye tracker during stereoscopic display.” In: *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization*. ACM, 2011, pp. 15–22. DOI: 10.1145/2077451.2077454.

[Duc+14] A. T. Duchowski, D. H. House, J. Gestring, R. Congdon, L. Świrski, N. A. Dodgson, K. Krejtz, and I. Krejtz. “Comparing estimated gaze depth in virtual and physical environments.” In: *Proceedings of the 8th ACM Symposium on Eye Tracking Research & Applications*. ACM, 2014, pp. 103–110. DOI: 10.1145/2578153.2578168.

[Duc+18] A. T. Duchowski, K. Krejtz, I. Krejtz, C. Biele, A. Niedzielska, P. Kiefer, M. Raubal, and I. Giannopoulos. “The index of pupillary activity: measuring cognitive load vis-à-vis task difficulty with pupil oscillation.” In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2018. DOI: 10.1145/3173574.3173856.

[DWA10] T. N. Dang, L. Wilkinson, and A. Anand. “Stacking graphic elements to avoid over-plotting.” In: *IEEE Transactions on Visualization and Computer Graphics* 16.6 (2010), pp. 1044–1052. DOI: 10.1109/tvcg.2010.197.

[Eck+17] M. K. Eckstein, B. Guerra-Carrillo, A. T. M. Singley, and S. A. Bunge. “Beyond eye gaze: What else can eyetracking reveal about cognition and cognitive development?” In: *Developmental Cognitive Neuroscience* 25 (2017), pp. 69–91. DOI: 10.1016/j.dcn.2016.11.001.

[EDF08] N. Elmqvist, P. Dragicevic, and J. Fekete. “Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation.” In: *IEEE Transactions on Visualization and Computer Graphics* 14.6 (2008), pp. 1539–1148. DOI: 10.1109/TVCG.2008.153.

[ElS+15] N. ElSayed, B. Thomas, K. Marriott, J. Piantadosi, and R. Smith. “Situated analytics.” In: *Proceedings of Big Data Visual Analytics*. IEEE, 2015. DOI: 10.1109/bdva.2015.7314302.

[Eme+12] J. W. Emerson, W. A. Green, B. Schloerke, J. Crowley, D. Cook, H. Hofmann, and H. Wickham. “The generalized pairs plot.” In: *Journal of Computational and Graphical Statistics* 22.1 (2012), pp. 79–91. DOI: 10.1080/10618600.2012.694762.

[Epa69] V. A. Epanechnikov. “Non-parametric estimation of a multivariate probability density.” In: *Theory of Probability & Its Applications* 14.1 (1969), pp. 153–158. DOI: 10.1137/1114019.

[Est+96] M. Ester, H. Kriegel, J. Sander, and X. Xu. “A density-based algorithm for discovering clusters in large spatial databases with noise.” In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. Ed. by E. Simoudis, J. Han, and U. M. Fayyad. AAAI Press, 1996, pp. 226–231.

[ET21] A. Eklund and J. Trimble. *beeswarm: The bee swarm plot, an alternative to stripchart*. R package version 0.4.0, https://CRAN.R-project.org/package=beeswarm, accessed 2023-05-20. 2021.

[Fec60] T. G. Fechner. *Elemente der Psychophysik*. Leipzig, Germany: Breitkopf & Härtel, 1860. DOI: 10.2307/1411906.

[Fis36] R. A. Fisher. “The use of multiple measurements in taxonomic problems.” In: *Annals of Eugenics* 7.2 (1936), pp. 179–188. DOI: 10.1111/j.1469-1809.1936.tb02137.x.

[FOO23] FOOTOVISION. *Footovision: Level up your game*. https://www.footovision.com/, accessed 2023-07-09. 2023.

[FR91] T. M. J. Fruchterman and E. M. Reingold. “Graph drawing by force-directed placement.” In: *Software: Practice and Experience* 21.11 (1991), pp. 1129–1164. DOI: 10.1002/spe.4380211102.

[Fra19] Fraunhofer Institute for Solar Energy Systems. *Energy-charts*. https://energy-charts.info/, accessed 2024-01-27. 2019.

[FWR99] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner. “Hierarchical parallel coordinates for exploration of large datasets.” In: *Proceedings of the IEEE Conference on Visualization*. IEEE, 1999, pp. 43–50. DOI: 10.1109/VISUAL.1999.809866.

[Gad+21] K. Gadhave, J. Görtler, Z. Cutler, C. Nobre, O. Deussen, M. Meyer, J. M. Phillips, and A. Lex. “Predicting intent behind selections in scatterplot visualizations.” In: *Information Visualization* 20.4 (2021), pp. 207–228. DOI: 10.1177/14738716211038604.

[Gla69] L. Glass. “Moiré effect from random dots.” In: *Nature* 223.5206 (1969), pp. 578–580. DOI: 10.1038/223578a0.

[Gue+12] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder. “Foveated 3D graphics.” In: *ACM Transactions on Graphics* 31.6 (2012), p. 164. DOI: 10.1145/2366145.2366183.

[Har16] A.-W. Harzing. *Publish or perish*. http://www.harzing.com/pop.htm, accessed: 2023-01-20. 2016.

[HC13] A. Haji-Abolhassani and J. J. Clark. “A computational model for task inference in visual search.” In: *Journal of Vision* 13.3 (2013), 29:1–29:24. DOI: 10.1167/13.3.29.

[Hei+12] J. Heinrich, Y. Luo, A. E. Kirkpatrick, H. Zhang, and D. Weiskopf. “Evaluation of a bundling technique for parallel coordinates.” In: *Proceedings of the International Conference on Computer Graphics Theory and Applications and International Conference on Information Visualization Theory and Applications*. SciTePress, 2012, pp. 594–602. DOI: 10.5220/0003821205940602.

[Hei16] J. Heinrich. *PACO*. http://parallelcoordinates.de/paco/, accessed 2023-05-13. 2016.

[Hir05] J. E. Hirsch. “An index to quantify an individual’s scientific research output.” In: *Proceedings of the National Academy of Sciences of the United States of America*. Vol. 102. 46. National Academy of Sciences, 2005, pp. 16569–16572. DOI: 10.1073/pnas.0507655102.

[Hla+13] M. Hlawatsch, F. Sadlo, M. Burch, and D. Weiskopf. “Scale-stack bar charts.” In: *Computer Graphics Forum* 32.3 (2013), pp. 181–190. DOI: 10.1111/cgf.12105.

[HM90] R. B. Haber and D. A. McNabb. “Visualization idioms: A conceptual model for scientific visualization systems.” In: *Visualization in scientific computing* 74 (1990), p. 93.

[HN96] P. Horton and K. Nakai. “A probabilistic classification system for predicting the cellular localization sites of proteins.” In: *Proceedings of the International Conference on Intelligent Systems for Molecular Biology*. Vol. 4. Association for the Advancement of Artificial Intelligence, 1996, pp. 109–115.

[HNR68] P. E. Hart, N. J. Nilsson, and B. Raphael. “A formal basis for the heuristic determination of minimum cost paths.” In: *IEEE Transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107. DOI: 10.1109/TSSC.1968.300136.

[HR07] J. Heer and G. G. Robertson. “Animated transitions in statistical data graphics.” In: *IEEE Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1240–1247. DOI: 10.1109/TVCG.2007.70539.

[Hur+19] C. Hurter, N. H. Riche, S. M. Drucker, M. Cordeil, R. Alligier, and R. Vuillemot. “FiberClay: Sculpting three dimensional trajectories to reveal structural insights.” In: *IEEE Transactions on Visualization and Computer Graphics* 25.1 (2019), pp. 704–714. DOI: 10.1109/TVCG.2018.2865191.

[HW09] D. Holten and J. J. van Wijk. “Force-directed edge bundling for graph visualization.” In: *Computer Graphics Forum* 28.3 (2009), pp. 983–990. DOI: 10.1111/j.1467-8659.2009.01450.x.

[HW13] J. Heinrich and D. Weiskopf. “State of the art of parallel coordinates.” In: *Eurographics 2013 – State of the Art Reports*. Ed. by M. Sbert and L. Szirmay-Kalos. The Eurographics Association, 2013, pp. 95–116. DOI: 10.2312/conf/eg2013/stars/095-116.

[I G15] I G. A. R. S. Padmi. “From sample to population: The use of dot plot to support the emergence of informal inferential reasoning.” In: *Proceedings of the 3rd South East Asia Design/Development Research International Conference*. Palembang, South Sumatra, Indonesia: Sriwijaya University, 2015, pp. 314–324.

[IB18] D. Ienco and G. Bordogna. “Fuzzy extensions of the DBSCAN clustering algorithm.” In: *Soft Computing* 22.5 (2018), pp. 1719–1730. DOI: 10.1007/s00500-016-2435-0.

[ID87] A. Inselberg and B. Dimsdale. “Parallel coordinates for visualizing multi-dimensional geometry.” In: *Proceedings of Computer Graphics International*. Springer-Verlag, Tokyo, Japan, 1987, pp. 25–44. DOI: 10.1007/978-4-431-68057-4_3.

[Ins09] A. Inselberg. *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. New York: Springer-Verlag, 2009.

[Ins85] A. Inselberg. “The plane with parallel coordinates.” In: *The Visual Computer* 1.2 (1985), pp. 69–91. DOI: 10.1007/BF01898350.

[Jac01] P. Jaccard. “Étude comparative de la distribution florale dans une portion des alpes et des jura.” In: *Bulletin de la Société Sciences Nat* 37 (1901), pp. 547–579.

[Jai+20] A. Jain, H. Patel, L. Nagalapatti, N. Gupta, S. Mehta, S. Guttula, S. Mujumdar, S. Afzal, R. S. Mittal, and V. Munigala. “Overview and importance of data quality for machine learning tasks.” In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020. DOI: 10.1145/3394486.3406477.

[JC08] J. Johansson and M. D. Cooper. “A screen space quality method for data abstraction.” In: *Computer Graphics Forum* 27.3 (2008), pp. 1039–1046. DOI: 10.1111/j.1467-8659.2008.01240.x.

[JC80] M. A. Just and P. A. Carpenter. “A theory of reading: From eye fixations to comprehension.” In: *Psychological Review* 87.4 (1980), pp. 329–354. DOI: 10.1037/0033-295x.87.4.329.

[Joh+05] J. Johansson, P. Ljung, M. Jern, and M. D. Cooper. “Revealing structure within clustered parallel coordinates displays.” In: *Proceedings of the IEEE Symposium on Information Visualization*. 2005, pp. 125–132. DOI: 10.1109/INFVIS.2005.1532138.

[Joh+16] P. V. Johnson, J. A. Parnell, J. Kim, C. D. Saunter, G. D. Love, and M. S. Banks. “Dynamic lens and monovision 3D displays to improve viewer comfort.” In: *Optics Express* 24.11 (2016), pp. 11808–11827. DOI: 10.1364/OE.24.011808.

[Jol02] I. T. Jolliffe. *Principal Component Analysis*. New York, USA: Springer-Verlag, 2002. 488 pp. DOI: 10.1007/b98835.

[KA05] R. J. Karunamuni and T. Alberts. “A generalized reflection method of boundary correction in kernel density estimation.” In: *Canadian Journal of Statistics* 33.4 (2005), pp. 497–509. DOI: 10.1002/cjs.5550330403.

[KBH06] R. Kosara, F. Bendix, and H. Hauser. “Parallel sets: interactive exploration and visual analysis of categorical data.” In: *IEEE Transactions on Visualization and Computer Graphics* 12.4 (2006), pp. 558–568. DOI: 10.1109/tvcg.2006.76.

[KCH19] Y. Kim, M. Correll, and J. Heer. “Designing Animated Transitions to Convey Aggregate Operations.” In: *Computer Graphics Forum* 38.3 (2019), pp. 541–551. DOI: 10.1111/cgf.13709.

[KCS17] J. W. Kelly, L. A. Cherep, and Z. D. Siegel. “Perceived space in the HTC Vive.” In: *ACM Transactions on Applied Perception* 15.1 (2017). DOI: 10.1145/3106155.

[KCW16] R. Konrad, E. A. Cooper, and G. Wetzstein. “Novel optical configurations for virtual reality.” In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2016. DOI: 10.1145/2858036.2858140.

[Kei+08] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. “Visual Analytics: Scope and Challenges.” In: *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*. Ed. by S. J. Simoff, M. H. Böhlen, and A. Mazeika. Berlin, Heidelberg: Springer, 2008, pp. 76–90. DOI: 10.1007/978-3-540-71080-6_6.

[KK89] T. Kamada and S. Kawai. “An algorithm for drawing general undirected graphs.” In: *Information Processing Letters* 31.1 (1989), pp. 7–15. DOI: 10.1016/0020-0190(89)90102-6.

[KN18] F. Koochaki and L. Najafizadeh. “Predicting intention through eye gaze patterns.” In: *Proceedings of the IEEE Biomedical Circuits and Systems Conference*. IEEE, 2018. DOI: 10.1109/biocas.2018.8584665.

[KN19] F. Koochaki and L. Najafizadeh. “Eye gaze-based early intent prediction utilizing CNN-LSTM.” In: *Proceedings of the 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2019. DOI: 10.1109/embc.2019.8857054.

[Kob12] S. G. Kobourov. *Spring embedders and force directed graph drawing algorithms*. 2012. DOI: 10.48550/arXiv.1201.3011.

[KPS12] S. Kriglstein, M. Pohl, and C. Stachl. “Animation for time-oriented data: An overview of empirical research.” In: *Proceedings of the 16th International Conference on Information Visualization*. IEEE Computer Society, 2012, pp. 30–35. DOI: 10.1109/IV.2012.16.

[Kre+18] K. Krejtz, A. T. Duchowski, A. Niedzielska, C. Biele, and I. Krejtz. “Eye tracking cognitive load using pupil diameter and microsaccades with fixed gaze.” In: *PLOS ONE* 13.9 (2018), e0203629. DOI: 10.1371/journal.pone.0203629.

[Kur+20] K. Kurzhals, N. Rodrigues, M. Koch, M. Stoll, A. Bruhn, A. Bulling, and D. Weiskopf. “Visual analytics and annotation of pervasive eye tracking video.” In: *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*. ACM, 2020. DOI: 10.1145/3379155.3391326.

[LA94] Y. K. Leung and M. D. Apperley. “A review and taxonomy of distortion-oriented presentation techniques.” In: *ACM Transactions on Computer-Human Interaction* 1.2 (1994), pp. 126–160. DOI: 10.1145/180171.180173.

[LBG80] Y. Linde, A. Buzo, and R. M. Gray. “An algorithm for vector quantizer design.” In: *IEEE Transactions on Communications* 28.1 (1980), pp. 84–95. DOI: 10.1109/TCOM.1980.1094577.

[Lee+17] Y. Lee, C. Shin, A. Plopski, Y. Itoh, T. Piumsomboon, A. Dey, G. Lee, S. Kim, and M. Billinghurst. “Estimating gaze depth using multi-layer perceptron.” In: *Proceedings of the International Symposium on Ubiquitous Virtual Reality*. IEEE, 2017, pp. 26–29. DOI: 10.1109/ISUVR.2017.13.

[Lee+20] B. Lee, E. K. Choe, P. Isenberg, K. Marriott, and J. Stasko. “Reaching broader audiences with data visualization.” In: *IEEE Computer Graphics and Applications* 40.2 (2020), pp. 82–90. DOI: 10.1109/mcg.2020.2968244.

[LHT17] A. Lhuillier, C. Hurter, and A. C. Telea. “State of the art in edge and trail bundling techniques.” In: *Computer Graphics Forum* 36.3 (2017), pp. 619–645. DOI: 10.1111/cgf.13213.

[Llo82] S. Lloyd. “Least squares quantization in PCM.” In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137. DOI: 10.1109/tit.1982.1056489.

[LLS19] A. Lundgard, C. Lee, and A. Satyanarayan. “Sociotechnical considerations for accessible visualization design.” In: *Proceedings of the IEEE Visualization Conference*. IEEE, 2019. DOI: 10.1109/visual.2019.8933762.

[Lu+19] V. T. Lu, C. E. Wright, C. Chubb, and G. Sperling. “Variation in target and distractor heterogeneity impacts performance in the centroid task.” In: *Journal of Vision* 19.4 (2019), p. 21. DOI: 10.1167/19.4.21.

[Mar+18] K. Marriott, F. Schreiber, T. Dwyer, K. Klein, N. H. Riche, T. Itoh, W. Stuerzlinger, and B. H. Thomas. *Immersive Analytics*. Springer International Publishing, 2018. DOI: 10.1007/978-3-030-01388-2.

[MFE19] L. Mcilreavy, T. C. A. Freeman, and J. T. Erichsen. “Two-dimensional analysis of smooth pursuit eye movements reveals quantitative deficits in precision and accuracy.” In: *Translational Vision Science and Technology* 8.5 (2019), 7:1–7:17. DOI: 10.1167/tvst.8.5.7.

[MG13] A. Mayorga and M. Gleicher. “Splatterplots: Overcoming overdraw in scatter plots.” In: *IEEE Transactions on Visualization and Computer Graphics* 19.9 (2013), pp. 1526–1538. DOI: 10.1109/tvcg.2013.65.

[MHM18] L. McInnes, J. Healy, and J. Melville. *UMAP: Uniform manifold approximation and projection for dimension reduction*. 2018. DOI: 10.48550/arXiv.1802.03426.

[Mil56] G. A. Miller. “Human memory and the storage of information.” In: *IRE Transactions on Information Theory* 2.3 (1956), pp. 129–137. DOI: 10.1109/TIT.1956.1056815.

[MM08] K. T. McDonnell and K. Mueller. “Illustrative parallel coordinates.” In: *Computer Graphics Forum* 27.3 (2008), pp. 1031–1038. DOI: 10.1111/j.1467-8659.2008.01239.x.

[MUT20] S. L. Matthews, A. Uribe-Quevedo, and A. Theodorou. “Rendering optimizations for virtual reality using eye-tracking.” In: *Proceedings of the 22nd Symposium on Virtual and Augmented Reality*. IEEE, 2020. DOI: 10.1109/svr51698.2020.00066.

[Net+17] R. Netzel, J. Vuong, U. Engelke, S. O’Donoghue, D. Weiskopf, and J. Heinrich. “Comparative eye-tracking evaluation of scatterplots and parallel coordinates.” In: *Visual Informatics* 1.2 (2017), pp. 118–131. DOI: 10.1016/j.visinf.2017.11.001.

[Net+19] R. Netzel, N. Rodrigues, A. Haug, and D. Weiskopf. “Compensation of simultaneous orientation contrast in superimposed textures.” In: *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2019. DOI: 10.5220/0007356800480057.

[Ngu+20] Q. V. Nguyen, N. Miller, D. Arness, W. Huang, M. L. Huang, and S. Simoff. “Evaluation on interactive visualization data with scatterplots.” In: *Visual Informatics* 4.4 (2020), pp. 1–10. DOI: 10.1016/j.visinf.2020.09.004.

[NTY01] K. Nakakoji, A. Takashima, and Y. Yamamoto. “Cognitive effects of animated visualization in exploratory visual data analysis.” In: *Proceedings of the 5th International Conference on Information Visualization*. IV. IEEE Computer Society, 2001, pp. 77–84. DOI: 10.1109/IV.2001.942042.

[Öne+20] S. Öney, N. Rodrigues, M. Becher, T. Ertl, G. Reina, M. Sedlmair, and D. Weiskopf. “Evaluation of gaze depth estimation from eye tracking in augmented reality.” In: *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*. ACM, 2020, 49:1–49:5. DOI: 10.1145/3379156.3391835.

[Onz+21] C. van Onzenoodt, G. Singh, T. Ropinski, and T. Ritschel. “Blue noise plots.” In: *Computer Graphics Forum* 40.2 (2021), pp. 425–433. DOI: 10.1111/cgf.142644.

[Pad+17] L. M. K. Padilla, P. S. Quinan, M. D. Meyer, and S. H. Creem-Regehr. “Evaluating the impact of binning 2D scalar fields.” In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 431–440. DOI: 10.1109/TVCG.2016.2599106.

[Pal+14] G. Palmas, M. Bachynskyi, A. Oulasvirta, H. P. Seidel, and T. Weinkauf. “An edge-bundling layout for interactive parallel coordinates.” In: *Proceedings of the IEEE Pacific Visualization Symposium*. IEEE, 2014, pp. 57–64. DOI: 10.1109/PacificVis.2014.40.

[Par62] E. Parzen. “On estimation of a probability density function and mode.” In: *The Annals of Mathematical Statistics* 33.3 (1962), pp. 1065–1076. DOI: 10.1214/aoms/1177704472.

[Pat+20] N. Pathmanathan, M. Becher, N. Rodrigues, G. Reina, T. Ertl, D. Weiskopf, and M. Sedlmair. “Eye vs. head: comparing gaze methods for interaction in augmented reality.” In: *Proceedings*

of the ACM Symposium on Eye Tracking Research & Applications. ACM, 2020. DOI: 10.1145/3379156.3391829.

[PD84] T. K. Porter and T. Duff. “Compositing digital images.” In: *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*. 1984, pp. 253–259. DOI: 10.1145/800031.808606.

[Per+18] C. Perin, T. Wun, R. Pusch, and S. Carpendale. “Assessing the Graphical Perception of Time and Speed on 2D+Time Trajectories.” In: *Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 698–708. DOI: 10.1109/tvcg.2017.2743918.

[Pup20] Pupil Labs GmbH. *Pupil Invisible: The world’s first deep learning powered eye tracking glasses*. https://pupil-labs.com/products/invisible/, accessed 2020-02-17. 2020.

[PW01] D. Purves and S. M. Williams. *Neuroscience*. 3rd ed. Sunderland, Massachusetts, USA: Sinauer Associates, 2001.

[PWR04] W. Peng, M. O. Ward, and E. A. Rundensteiner. “Clutter reduction in multi-dimensional data visualization using dimension reordering.” In: *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, 2004, pp. 89–96. DOI: 10.1109/INFVIS.2004.15.

[Qin+17] H. Qin, Y. Chen, J. He, and B. Chen. “Wasserstein blue noise sampling.” In: *ACM Transactions on Graphics* 36.5 (2017), 168:1–168:13. DOI: 10.1145/3119910.

[Qui93] J. R. Quinlan. “Combining instance-based and model-based learning.” In: *Proceedings of the 10th International Conference on Machine Learning*. Morgan Kaufmann, 1993, pp. 236–243. DOI: 10.1016/b978-1-55860-307-3.50037-x.

[RB10] R. A. Rensink and G. Baldridge. “The perception of correlation in scatterplots.” In: *Computer Graphics Forum* 29.3 (2010), pp. 1203–1210. DOI: 10.1111/j.1467-8659.2009.01694.x.

[Rei+20] G. Reina, H. Childs, K. Matković, K. Bühler, M. Waldner, D. Pugmire, B. Kozlíková, T. Ropinski, P. Ljung, T. Itoh, E. Gröller, and M. Krone. “The moving target of visualization software for an increasingly complex world.” In: *Computers & Graphics* 87 (2020), pp. 12–29. DOI: 10.1016/j.cag.2020.01.005.

[RGA95] J. P. Rolland, W. Gibson, and D. Ariely. “Towards quantifying depth and size perception in virtual environments.” In: *Presence: Teleoperators & Virtual Environments* 4.1 (1995), pp. 24–49. DOI: 10.1162/pres.1995.4.1.24.

[Ric+23] G. Richer, A. Pister, M. Abdelaal, J.-D. Fekete, M. Sedlmair, and D. Weiskopf. “Scalability in visualization.” In: *IEEE Transactions on Visualization and Computer Graphics* (2023), 0:1–0:15. DOI: 10.1109/tvcg.2022.3231230.

[Rit86] J. Rit. “Propagating temporal constraints for scheduling.” In: *Proceedings of the 5th National Conference on Artificial Intelligence*. Ed. by T. Kehler. Morgan Kaufmann, 1986, pp. 383–388.

[Rob+08] G. G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. T. Stasko. “Effectiveness of animation in trend visualization.” In: *IEEE Transactions on Visualization and Computer Graphics* 14.6 (2008), pp. 1325–1332. DOI: 10.1109/TVCG.2008.125.

[Rod+17a] N. Rodrigues, M. Burch, L. D. Silvestro, and D. Weiskopf. “A visual analytics approach for word relevances in multiple texts.” In: *Proceedings of the 21st International Conference Information Visualisation*. IEEE, 2017. DOI: 10.1109/iv.2017.62.

[Rod+17b] N. Rodrigues, R. Netzel, K. R. Ullah, M. Burch, A. Schultz, B. Burger, and D. Weiskopf. “Visualization of time series data with spatial context: communicating the energy production of power plants.” In: *Proceedings of the 10th International Symposium on Visual Information Communication and Interaction*. ACM, 2017, pp. 37–44. DOI: 10.1145/3105971.3105982.

[Rod+18] N. Rodrigues, R. Netzel, J. Spalink, and D. Weiskopf. “Multiscale scanpath visualization and filtering.” In: *Proceedings of the 3rd Workshop on Eye Tracking and Visualization*. ACM, 2018. DOI: 10.1145/3205929.3205931.

[Rod+20] N. Rodrigues, C. Schulz, A. Lhuillier, and D. Weiskopf. “Cluster-flow parallel coordinates: Tracing clusters across subspaces.” In: *Proceedings of Graphics Interface*. Canadian Human-Computer Communications Society, 2020, pp. 382–392. DOI: 10.20380/GI2020.38.

[Rod+22a] N. Rodrigues, F. Dennig, V. Brandt, D. Keim, and D. Weiskopf. *Evaluation of animated scatter plot transitions*. 2022. DOI: 10.17605/OSF.IO/3E6G4.

[Rod+22b] N. Rodrigues, C. Schulz, S. Döring, D. Baumgartner, T. Krake, and D. Weiskopf. *Supplemental material for relaxed dot plots*. 2022. DOI: 10.18419/darus-3055.

[Rod+22c] N. Rodrigues, L. Shao, J. J. Yan, T. Schreck, and D. Weiskopf. "Eye gaze on scatterplot: Concept and first results of recommendations for exploration of SPLOMs using implicit data selection." In: *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*. ACM, 2022, 59:1–59:7. DOI: 10.1145/3517031.3531165.

[Rod+23a] N. Rodrigues, F. L. Dennig, V. Brandt, D. A. Keim, and D. Weiskopf. *Supplemental material for comparative evaluation of animated scatter plot transitions*. 2023. DOI: 10.18419/darus-3451.

[Rod+23b] N. Rodrigues, C. Schulz, S. Döring, D. Baumgartner, T. Krake, and D. Weiskopf. "Relaxed dot plots: faithful visualization of samples and their distribution." In: *IEEE Transactions on Visualization and Computer Graphics* 29.1 (2023), pp. 278–287. DOI: 10.1109/TVCG.2022.3209429.

[Rod+24] N. Rodrigues, F. L. Dennig, V. Brandt, D. A. Keim, and D. Weiskopf. "Comparative evaluation of animated scatter plot transitions." In: *IEEE Transactions on Visualization and Computer Graphics* (2024). Early access, p. 11. DOI: 10.1109/TVCG.2024.3388558.

[Rod17] N. Rodrigues. *Sample implementation of nonlinear dot plots*. https://github.com/NilsRodrigues/nonlineardotplots, accessed 2023-05-20. 2017.

[Rod20] N. Rodrigues. *Sample implementation of cluster-flow parallel coordinates*. https://github.com/NilsRodrigues/clusterflow-pcp, accessed 2023-05-14. 2020.

[Rod23] N. Rodrigues. *Interactive demo of study: Evaluation of animated scatter plot transitions*. https://github.com/NilsRodrigues/animated-scatterplot-transitions-for-comparative-study, accessed 2023-05-06. 2023.

[Rol+02] J. P. Rolland, C. Meyer, K. Arthur, and E. Rinalducci. "Method of adjustments versus method of constant stimuli in the quantification of accuracy and precision of rendered depth in head-mounted displays." In: *Presence: Teleoperators & Virtual Environments* 11.6 (2002), pp. 610–625.

[Ros56] M. Rosenblatt. “Remarks on some nonparametric estimates of a density function.” In: *The Annals of Mathematical Statistics* 27.3 (1956), pp. 832–837. DOI: 10.1214/aoms/1177728190.

[Rot+96] K. G. Rottach, A. Z. Zivotofsky, V. E. Das, L. Averbuch-Heller, A. O. Discenna, A. Poonyathalang, and R. J. Leigh. “Comparison of horizontal, vertical and diagonal smooth pursuit eye movements in normal human subjects.” In: *Vision Research* 36.14 (1996), pp. 2189–2195. DOI: 10.1016/0042-6989(95)00302-9.

[Row91] J. M. Rowell. “Tunneling.” In: *Encyclopedia of Physics*. 2nd ed. New York, USA: VCH Publishers, 1991, pp. 1306–1310.

[RP17] P. Renner and T. Pfeiffer. “Attention guiding techniques using peripheral vision and eye tracking for feedback in augmented-reality-based assistance systems.” In: *Proceedings of the IEEE Symposium on 3D User Interfaces*. IEEE, 2017, pp. 186–194. DOI: 10.1109/3DUI.2017.7893338.

[RW18] N. Rodrigues and D. Weiskopf. “Nonlinear dot plots.” In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 616–625. DOI: 10.1109/tvcg.2017.2744018.

[Sch+16] C. Schulz, A. Nocaj, M. El-Assady, S. Frey, M. Hlawatsch, M. Hund, G. K. Karch, R. Netzel, C. Schätzle, M. Butt, D. A. Keim, T. Ertl, U. Brandes, and D. Weiskopf. “Generative data models for validation and evaluation of visualization techniques.” In: *Proceedings of the 6th Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*. Ed. by M. Sedlmair, P. Isenberg, T. Isenberg, N. Mahyar, and H. Lam. ACM, 2016, pp. 112–124. DOI: 10.1145/2993901.2993907.

[Sch+17] C. Schulz, N. Rodrigues, K. Damarla, A. Henicke, and D. Weiskopf. “Visual exploration of mainframe workloads.” In: *Proceedings of the SIGGRAPH Asia Symposium on Visualization*. ACM, 2017, 4:1–4:7. DOI: 10.1145/3139295.3139312.

[Sch+19] C. Schulz, N. Rodrigues, M. Amann, D. Baumgartner, A. Mielke, C. Baumann, M. Sedlmair, and D. Weiskopf. “A framework for pervasive visual deficiency simulation.” In: *Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces*. IEEE, 2019, pp. 1852–1857. DOI: 10.1109/vr44988.2019.9044164.

[Sch+22] P. Schäfer, N. Rodrigues, D. Weiskopf, and S. Storandt. “Group diagrams for simplified representation of scanpaths.” In: *Proceedings of the 15th International Symposium on Visual Information Communication and Interaction*. ACM, 2022, 7:1–7:8. DOI: 10.1145/3554944.3554971.

[Sch+23] P. Schäfer, N. Rodrigues, D. Weiskopf, and S. Storandt. “Group diagrams for simplified representation of scanpaths.” In: *Journal of Visualization* 26.5 (2023), pp. 1173–1187. DOI: 10.1007/s12650-023-00924-4.

[Sch18] C. Schulz. *fuzzy_dbscan*. https://github.com/schulzch/fuzzy_dbscan, accessed 2023-05-14. 2018.

[SD11] T. Schlömer and O. Deussen. “Accurate spectral analysis of two-dimensional point sets.” In: *Journal of Graphics, GPU, and Game Tools* 15.3 (2011), pp. 152–160. DOI: 10.1080/2151237x.2011.609773.

[Sha+17] L. Shao, N. Silva, E. Eggeling, and T. Schreck. “Visual exploration of large scatter plot matrices by pattern recommendation based on eye tracking.” In: *Proceedings of the ACM Workshop on Exploratory Search and Interactive Data Analytics*. New York, NY, USA: ACM, 2017, pp. 9–16. DOI: 10.1145/3038462.3038463.

[Sil+18] N. Silva, T. Schreck, E. Veas, V. Sabol, E. Eggeling, and D. W. Fellner. “Leveraging eye-gaze and time-series features to predict user interests and build a recommendation model for visual analysis.” In: *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*. New York, NY, USA: Association for Computing Machinery, 2018, 13:1–13:9. DOI: 10.1145/3204493.3204546.

[Sil+19] N. Silva, T. Blascheck, R. Jianu, N. Rodrigues, D. Weiskopf, M. Raubal, and T. Schreck. “Eye tracking support for visual analytics systems: Foundations, current applications, and research challenges.” In: *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. ACM, 2019, 11:1–11:10. DOI: 10.1145/3314111.3319919.

[SP09] B. A. Schneider and S. Parker. “Human methods: Psychophysics.” In: *Encyclopedia of Neuroscience*. Elsevier, 2009, pp. 19–27. DOI: 10.1016/b978-008045046-9.00292-8.

[SR96] P. D. Sasieni and P. Royston. “Dotplots.” In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 45.2 (1996), pp. 219–234. DOI: 10.2307/2986156.

[SSE15] J. E. Swan, G. Singh, and S. R. Ellis. “Matching and reaching depth judgments with real and augmented reality targets.” In: *IEEE Transactions on Visualization and Computer Graphics* 21.11 (2015), pp. 1289–1298.

[SW12] H. Sanftmann and D. Weiskopf. “3D scatterplot navigation.” In: *Transactions on Visualization and Computer Graphics* 18.11 (2012), pp. 1969–1978. DOI: 10.1109/TVCG.2012.35.

[SW82] A. Steinbach and K. Y. Wong. “Moiré patterns in scanned halftone pictures.” In: *Journal of the Optical Society of America* 72.9 (1982), pp. 1190–1198. DOI: 10.1364/josa.72.001190.

[Tat+09] A. Tatu, G. Albuquerque, M. Eisemann, J. Schneidewind, H. Theisel, M. Magnork, and D. A. Keim. “Combining automated analysis and visualization techniques for effective exploration of high-dimensional data.” In: *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*. IEEE, 2009, pp. 59–66. DOI: 10.1109/VAST.2009.5332628.

[Tho+18] B. H. Thomas, G. F. Welch, P. Dragicevic, N. Elmqvist, P. Irani, Y. Jansen, D. Schmalstieg, A. Tabard, N. A. M. ElSayed, R. T. Smith, and W. Willett. “Situated Analytics.” In: *Immersive Analytics*. Ed. by K. Marriott, F. Schreiber, T. Dwyer, K. Klein, N. H. Riche, T. Itoh, W. Stuerzlinger, and B. H. Thomas. Cham: Springer International Publishing, 2018. Chap. 7, pp. 185–220. DOI: 10.1007/978-3-030-01388-2_7.

[TK07] T. Tekušová and J. Kohlhammer. “Applying animation to the visual analysis of financial time-dependent data.” In: *Proceedings of the 11th International Conference on Information Visualization*. IEEE Computer Society, 2007, pp. 101–108. DOI: 10.1109/IV.2007.28.

[TLS21] J. R. Thompson, Z. Liu, and J. Stasko. “Data animator: Authoring expressive animated data graphics.” In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2021. DOI: 10.1145/3411764.3445747.

[TM05] M. Tory and T. Möller. “Evaluating visualizations: Do expert reviews work?” In: *IEEE Computer Graphics and Applications* 25.5 (2005), pp. 8–11. DOI: 10.1109/mcg.2005.102.

[Tom+21] C. Tominski, G. Andrienko, N. Andrienko, S. Bleisch, S. I. Fabrikant, E. Mayr, S. Miksch, M. Pohl, and A. Skupin. “Toward flexible visual analytics augmented through smooth display transitions.” In: *Visual Informatics* 5.3 (2021), pp. 28–38. DOI: 10.1016/j.visinf.2021.06.004.

[Tow22] K. Townsend. “Solving the quantum decryption ‘harvest now, decrypt later’problem.” In: *Security Week* (2022). https://www.securityweek.com/solving-quantum-decryption-harvest-now-decrypt-later-problem/, accessed 2023-04-23.

[TSD09] M. Tory, C. Swindells, and R. Dreezer. “Comparing dot and landscape spatializations for visual memory differences.” In: *IEEE Transactions on Visualization and Computer Graphics* 15.6 (2009), pp. 1033–1039. DOI: 10.1109/tvcg.2009.127.

[TT85] J. W. Tukey and P. A. Tukey. “Computer graphics and exploratory data analysis: An introduction.” In: *Proceedings of the 6th Annual Conference and Exposition: Computer Graphics*. Vol. 85. 3. 1985, pp. 773–785.

[Tuf83] E. R. Tufte. *The Visual Display of Quantitative Information*. Cheshire, CT: Graphics Press, 1983. DOI: 10.1097/01445442-198507000-00012.

[Tuk72] J. W. Tukey. “Some graphic and semigraphic displays.” In: *Statistical Papers in Honor of George W. Snedecor*. Ed. by T. Bancroft. Iowa State University Press, 1972, pp. 293–316.

[UK 21] UK National Quantum Technologies Programme. *Quantum-safe secure communications*. https://uknqt.ukri.org/wp-content/uploads/2021/10/Quantum-Safe-Secure-Communications.pdf, accessed 2023-04-23. 2021.

[Utt05] J. M. Utts. *Seeing Through Statistics*. 2nd ed. Pacific Grove, California, USA: Brooks/Cole, 2005.

[Vav+13] G. Vavoulas, M. Pediaditis, E. G. Spanakis, and M. Tsiknakis. “The MobiFall dataset: An initial evaluation of fall detection algorithms using smartphones.” In: *Proceedings of the IEEE International Conference on Bioinformatics and Bioengineering*. IEEE, 2013, pp. 1–4. DOI: 10.1109/bibe.2013.6701629.

[Vav+17] G. Vavoulas, M. Pediaditis, C. Chatzaki, E. G. Spanakis, and M. Tsiknakis. “The MobiFall dataset: Fall detection and classification with a smartphone.” In: *Artificial Intelligence: Concepts, Methodologies, Tools, and Applications*. Hershey, Pennsylva-

nia, USA: IGI Global, 2017, pp. 1218–1231. DOI: 10.4018/978-1-5225-1759-7.ch048.

[vd03] R. van Liere and W. de Leeuw. “GraphSplatting: Visualizing graphs as continuous fields.” In: *IEEE Transactions on Visualization and Computer Graphics* 9.2 (2003), pp. 206–212. DOI: 10.1109/tvcg.2003.1196007.

[WAG06] L. Wilkinson, A. Anand, and R. Grossman. “High-dimensional visual analytics: Interactive exploration guided by pairwise views of point distributions.” In: *IEEE Transactions on Visualization and Computer Graphics* 12.6 (2006), pp. 1363–1372. DOI: 10.1109/tvcg.2006.94.

[Wan+17] J. Wang, X. Liu, H.-W. Shen, and G. Lin. “Multi-resolution climate ensemble parameter analysis with nested parallel coordinates plots.” In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 81–90. DOI: 10.1109/tvcg.2016.2598830.

[War05] M. O. Ward. *Netperf dataset*. https://davis.wpi.edu/~xmdv/datasets/netperf.html, accessed 2023-04-24. 2005.

[Weg90] E. J. Wegman. “Hyperdimensional data analysis using parallel coordinates.” In: *Journal of the American Statistical Association* 85.411 (1990), pp. 664–675. DOI: 10.1080/01621459.1990.10474926.

[Wik21] Wikipedia contributors. *List of countries by renewable electricity production—Wikipedia, the free encyclopedia*. https://en.wikipedia.org/w/index.php?title=List_of_countries_by_renewable_electricity_production&oldid=1015648675, accessed 2021-05-18. 2021.

[Wil99] L. Wilkinson. “Dot plots.” In: *The American Statistician* 53.3 (1999), pp. 276–281. DOI: 10.2307/2686111.

[WW11] L. Y. Wei and R. Wang. “Differential domain analysis for non-uniform sampling.” In: *ACM Transactions on Graphics* 30.4 (2011), 50:1–50:10. DOI: 10.1145/2010324.1964945.

[Yan+15] D. M. Yan, J. W. Guo, B. Wang, X. P. Zhang, and P. Wonka. “A survey of blue-noise sampling and its applications.” In: *Journal of Computer Science and Technology* 30.3 (2015), pp. 439–452. DOI: 10.1007/s11390-015-1535-0.

[Yan+22] J. J. Yan, N. Rodrigues, L. Shao, T. Schreck, and D. Weiskopf. *Sample implementation for the paper 'Eye gaze on Scatterplot: Concept and first results of recommendations for exploration of SPLOMs using implicit data selection'*. 2022. DOI: 10.18419/DARUS-2810.

[Yao+22] L. Yao, A. Bezerianos, R. Vuillemot, and P. Isenberg. "Visualization in motion: A research agenda and two evaluations." In: *Transactions on Visualization and Computer Graphics* 28.10 (2022), pp. 3546–3562. DOI: 10.1109/tvcg.2022.3184993.

[Yua+11] Z. Yuan, G. Rong, X. Guo, and W. Wang. "Generalized Voronoi diagram computation on GPU." In: *Proceedings of the 8th International Symposium on Voronoi Diagrams in Science and Engineering*. Ed. by F. Anton and W. Wang. IEEE Computer Society, 2011, pp. 75–82. DOI: 10.1109/ISVD.2011.18.

[Zha+16] Q. Zhao, S. Chang, F. M. Harper, and J. A. Konstan. "Gaze prediction for recommender systems." In: *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2016, pp. 131–138. DOI: 10.1145/2959100.2959150.

[Zho+08] H. Zhou, X. Yuan, H. Qu, W. Cui, and B. Chen. "Visual clustering in parallel coordinates." In: *Computer Graphics Forum* 27.3 (2008), pp. 1047–1054. DOI: 10.1111/j.1467-8659.2008.01241.x.

[Zho+09] H. Zhou, W. Cui, H. Qu, Y. Wu, X. Yuan, and W. Zhuo. "Splatting the lines in parallel coordinates." In: *Computer Graphics Forum* 28.3 (2009), pp. 759–766. DOI: 10.1111/j.1467-8659.2009.01476.x.

[ZK12] X. Zhao and A. Kaufman. "Structure revealing techniques based on parallel coordinates plot." In: *The Visual Computer* 28.6 (2012), pp. 541–551. DOI: 10.1007/s00371-012-0713-0.

[ZP23] L. Ziaka and A. Protopapas. "Cognitive control beyond single-item tasks: Insights from pupillometry, gaze, and behavioral measures." In: *Journal of Experimental Psychology: Human Perception and Performance* 49.7 (2023), pp. 968–988. DOI: 10.1037/xhp0001127.

[ZPR16] J. Zagermann, U. Pfeil, and H. Reiterer. "Measuring cognitive load using eye tracking technology in visual computing." In: *Proceedings of the 6th Workshop on Beyond Time and Errors*

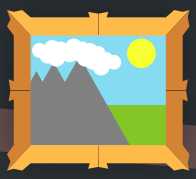
on Novel Evaluation Methods for Visualization. ACM, 2016. DOI: 10.1145/2993901.2993908.

Visualization plays an important role in the lives of various heterogeneous parts of society: from a voter looking for the latest results of an election, to statisticians examining a distribution, to analysts trying to make sense of multidimensional data sets.

This thesis adapts existing point- and line-based visualization methods to improve knowledge gain and targets viewers with varying levels of prior experience. The included contributions address three research questions:

How to scale unit visualization for 1D data?

How to improve navigation between 2D visualizations of multivariate data?



How to combine the advantages of multiple 2D views in a single static visualization for multivariate data?