# Visualization Techniques for Parallel Coordinates

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

Vorgelegt von

Julian Heinrich

aus Nairobi / Kenia

| | |
|---|---|
| Hauptberichter: | Prof. Dr. Daniel Weiskopf |
| Mitberichterin: | Prof. Dr.-Ing. Heidrun Schumann |
| Tag der mündlichen Prüfung: | 26.03.2013 |

Visualisierungsinstitut
der Universität Stuttgart

2013

# Contents

# List of Figures

# List of Tables

# List of Abbreviations and Acronyms

| | |
|---|---|
| **ASA** | American Statistical Association |
| **BRN** | biological reaction networks |
| **C3** | caspase 3 |
| **C3a** | active caspase 3 |
| **C8** | caspase 8 |
| **C8a(0)** | active caspase 8 |
| **CARP** | caspases 8- and 10-associated RING protein |
| **CDC15** | cell division control protein |
| **cDNA** | complementary DNA |
| **CFD** | computational fluid dynamics |
| **CPU** | central processing unit |
| **CT** | computer tomography |
| **EDA** | exploratory data analysis |
| **GLSL** | OpenGL Shading Language |
| **GPU** | graphics processing unit |
| **IAP** | inhibitor of apoptosis protein |
| **KDD** | knowledge discovery in databases |
| **LOD** | level-of-detail |
| **mRNA** | messenger RNA |
| **ODE** | ordinary differential equation |
| **OpenGL** | Open Graphics Library |
| **PC** | personal computer |
| **PCM** | parallel-coordinates matrix |
| **P-SPLOM** | parallel scatterplot matrix |
| **RAM** | random-access memory |
| **SPLOM** | scatterplot matrix |
| **SQL** | structured query language |
| **SVM** | support vector machine |
| **USA** | United States of America |
| **VTK** | visualization toolkit |

### Units

| | |
|---|---|
| **fps** | frames per second |
| **GB** | Gigabyte, $10^9$ bytes |
| **GHz** | Gigahertz, $10^9$ Hertz |

# Acknowledgments

I would like to thank my supervisor Daniel Weiskopf for making this work possible. His advice and commitment had a great impact on my work. I enjoyed many fruitful discussions with him over the years and I am thankful for his support in my research. It was a pleasure to be a PhD student under his guidance.

I am also grateful to my second referee Heidrun Schumann for reviewing this work and to the Deutsche Forschungsgemeinschaft (DFG) for funding parts of my research within the Cluster of Excellence in Simulation Technology at the University of Stuttgart.

I would further like to thank the following collaborators for their contributions to the research that has become part of this thesis: Kay Nieselt (for the long-lasting and ongoing cooperation in bioinformatics and biological data visualization), Dirk Bartz, John Stasko (for having me as a visiting student), Janko Dietzsch, Sven Bachthaler, Jan Hasenauer, Arthur ("Ted") Kirkpatrick, Hao ("Richard") Zhang, and Yuan Luo.

Special thanks go to Alfred Inselberg for his work on parallel coordinates which constitutes the basis for most of my research. I always welcomed his valuable comments on my contributions and enjoyed some fruitful discussions with him.

I conducted the research for this thesis at the Visualization Research Center (VISUS) at the University of Stuttgart, which provided a very pleasant working environment among many colleagues and friends. I had a great time with my roommates Corinna Vehlow, Michael Burch, and Sebastian Grottel. I am indebted to Martin Falk for sharing his LaTeX template and providing support whenever I needed it. Many thanks go to the "Mensa-Boykott" group for providing haut-cuisine alternatives to the Cafeteria and to Marco Ament for the continued and (fundamentally important) coffee supply.

Outside VISUS, I would like to thank Simone Götze, Kevin Kempfer, and Markus Huber for being good friends and for proof-reading.

Finally, I want to thank my wife Sonja for her never-ending support during the past five years. Thank you for your understanding and endurance in bearing with (and without) me when the next deadline was coming up.

# Abstract

Visualization plays a key role in knowledge discovery, visual data exploration, and visual analytics. Static images are an effective tool for visual communication, summarization, and pattern extraction in large and complex datasets. Only together with human-computer-interaction techniques, visual interfaces enable an analyst to explore large information spaces and to drive the whole analytical reasoning process.

Scatterplots and parallel coordinates are well-recognized visualization techniques that are commonly employed for statistics (both explorative and descriptive) and data-mining, but are also gaining importance for scientific visualization. While scatterplots are restricted to the display of at most three dimensions due to the orthogonal layout of coordinate axes, a parallel arrangement allows for the visualization of multiple attributes of a dataset. Although both techniques rely on projections of higher-dimensional geometry and are related by a point–line duality, parallel coordinates enjoy great popularity for the visualization and analysis of multivariate data.

Despite their popularity, parallel coordinates are subject to a number of limitations that remain to be solved. For large datasets, the potentially high amount of overlapping lines may hinder the observer from visually extracting meaningful patterns. Encoding observations with polylines make it difficult to follow lines over all dimensions, as they lose visual continuation across the axes. Clusters cannot be represented by the geometry of lines, and the order of axes has a high impact on the patterns exhibited by parallel coordinates.

This thesis presents visualization techniques for parallel coordinates that address these limitations. As a foundation, an extensive review of the state of the art of parallel coordinates will be given. Based on the point–line duality, the existing model of continuous scatterplots is adapted to parallel coordinates for the visualization of data defined on continuous domains. To speed up computation and obtain interactive frame rates, a scalable and progressive rendering algorithm is introduced that further allows for arbitrary reconstruction and interpolation schemes. A curve-bundling model for parallel coordinates is evaluated with a user study showing that bundling is effective for cluster visualization based on geometric cues while being equally capable of revealing correlations between neighboring axes. To address the axis-order problem, a graph-based approach is presented that allows for the visualization of all pairwise relations in a matrix layout without redundancy. Finally, the use of parallel coordinates is demonstrated for real datasets from computational fluid dynamics, motion capturing, bioinformatics, and systems biology.

# German Abstract
## —Zusammenfassung—

Visualisierung spielt eine zentrale Rolle für die Wissensfindung, in der visuellen Datenexploration und für Visual Analytics. Statische Bilder können effektiv für die visuelle Kommunikation sowie zur Extraktion von Strukturen und Mustern in großen und komplexen Datensätzen eingesetzt werden. Erst im Zusammenspiel mit Techniken der Mensch-Computer-Interaktion jedoch ermöglichen visuelle Schnittstellen dem Analysten große Informationsräume zu erkunden und den gesamten Prozess der Wissensfindung zu steuern.

Streudiagramme und parallele Koordinaten sind verbreitete Visualisierungen, welche insbesondere in der klassischen (explorativen sowie deskriptiven) statistischen Datenanalyse und im Data-Mining verwendet werden, aber auch in der technischen Visualisierung immer mehr Anwendung finden. Während Streudiagramme aufgrund der orthogonalen Anordnung der Koordinatenachsen auf die Visualisierung von höchstens drei Dimensionen beschränkt sind, erlaubt ein paralleler Aufbau der Achsen die Darstellung mehrerer Attribute eines Datensatzes. Obwohl beide Techniken lediglich Projektionen höherdimensionaler Geometrien abbilden können und durch eine Punkt–Linie-Dualität sogar ineinander überführbar sind, erfreuen sich parallele Koordinaten einer wachsenden Beliebtheit für die Visualisierung und Analyse multivariater Daten.

Trotz ihrer Beliebtheit bedarf es weiterer Forschung bezüglich einiger Aspekte von parallelen Koordinaten. So kann es aufgrund der potenziell starken Überlappung einzelner Linien für den Betrachter mitunter schwierig sein, interessante Merkmale in einem Parallelen-Koordinaten-Diagramm zu entdecken. Der Einsatz von Polygonzügen zur Darstellung von Beobachtungspunkten kann weiterhin dazu führen, dass einzelne Linien nicht oder nur schwer visuell über alle Achsen verfolgt werden können. Gruppen von Linien mit ähnlichen Eigenschaften (engl. *cluster*) können nicht geometrisch dargestellt werden und die Reihenfolge der Achsen wirkt sich entscheidend auf die entstehenden visuellen Muster aus.

In dieser Dissertation werden daher Visualisierungstechniken vorgestellt, welche sich mit diesen Problemen befassen. Als Grundlage wird eine ausführliche Übersicht des aktuellen Stands der Forschung bezüglich paralleler Koordinaten gegeben. Basierend auf der Punkt–Linie-Dualität wird das bestehende Modell der kontinuierlichen Streudiagramme für die Visualisierung kontinuierlicher Daten auf parallele Koordinaten erweitert. Um eine interaktive Bildfrequenz zu erhalten, wird ein Beschleunigungsverfahren diskutiert, welches eine schrittweise, progressive Bildgenerierung sowie beliebige Rekonstruktionsfilter erlaubt. Weiterhin wird ein Verfahren zur Bündelung von Kurven in parallelen Koordinaten vorgestellt und evaluiert, welches nachweislich zu einer effizienteren Erkennung

hochdimensionaler Cluster auf einer geometrischen Basis verwendet werden kann, während die Einschätzung linearer Korrelationen zwischen benachbarten Achsen weiterhin möglich bleibt. Schließlich wird eine graphentheoretische Methode vorgestellt, welche es erlaubt, alle paarweisen Relationen eines multivariaten Datensatzes redundanzfrei in parallelen Koordinaten darzustellen. Ein Teil der vorgestellten Visualisierungstechniken wird abschließend beispielhaft für die Analyse realer Datensätze aus der numerischen Strömungsmechanik, der Bewegungs-Erfassung, der Bioinformatik und der Systembiologie demonstriert.

Chapter 1

# Introduction

Visualization is considered a valuable tool to communicate, understand, and interact with complex and large amounts of information. Flowcharts, line diagrams, or pie charts are popular ways to present data in static images and are included in many spreadsheet and presentation applications. Similarly, most statistics software packages provide interactive graphing facilities and visual interfaces to data-mining algorithms in order to support the whole knowledge discovery process.

However, there are many challenges that remain to be solved with respect to visualization and interaction techniques. A particularly difficult task is the visualization of multidimensional data, as human cognition has evolved in a three-dimensional (spatial) world and most rendering devices such as paper or computer screens support only two dimensions. Consequently, multidimensional visualization techniques typically rely on the projection of the data onto lower-dimensional spaces and use Cartesian coordinates for visualization. The orthogonal layout of axes in a Cartesian coordinate system, however, restricts the visualization of data defined therein to a maximum of three dimensions.

Parallel coordinates take a fundamentally different approach to the visualization of multidimensional data. The parallel layout of axes permits an arbitrary number of dimensions to be visualized on screen. The point–line duality between two-dimensional Cartesian and parallel coordinates further guarantees a unique mapping of any point-feature from a scatterplot to a line-feature in parallel coordinates. Remarkably, the duality can be generalized to points and hyperplanes as well. While being well recognized in the visualization and statistical graphics communities, parallel coordinates are subject to a number of problems that partly arise from their specific layout and are partly related to their dual

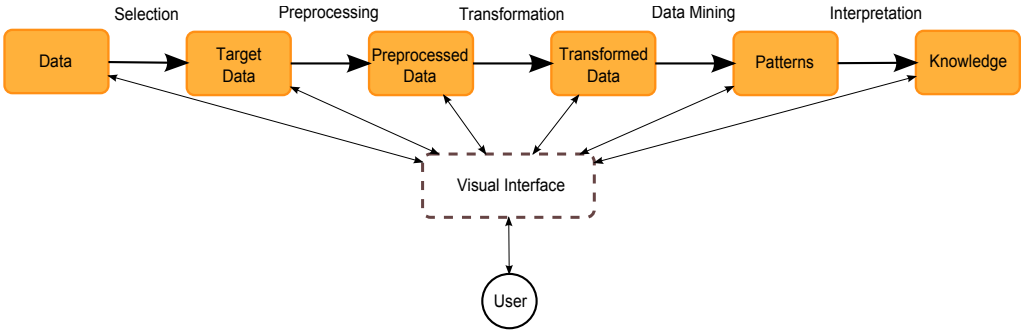



**Figure 1.1:** The knowledge discovery in databases (KDD) pipeline adapted from Fayyad et al. (1996b). While data flow is directed and the individual steps are typically executed from left to right, the whole process is iterative. Interactive visual interfaces are commonly used to control the pipeline and are thus emphasized here.

problems in Cartesian coordinates.

This thesis provides visualization techniques to address those challenges. In particular, a continuous model for the computation of the line density in parallel coordinates is introduced. Density-based visualization naturally solves the overplotting problem and provides a consistent model for continuous data. To allow for interactive frame rates, a progressive refinement technique is discussed which is applicable for both scatterplots and parallel coordinates. Finally, approaches for the geometry-based visualization of clusters as well as for the axis-ordering problem are presented.

In the following sections, some important terms and concepts in the context of visualization, knowledge discovery, and multivariate data are introduced. Parallel coordinates will be described only briefly, as Chapter 2 provides an extensive review of existing techniques for this type of visualization.

## 1.1 Knowledge Discovery

Today, data is being acquired and stored at unprecedented rates (Keim et al., 2006). Scientists collect data from satellites and genomes, retailers store information on customers, financial institutes record credit card transactions, etc. Manually turning these large volumes of data into knowledge is nearly impossible, as humans are not made for high-throughput sequential processing. Instead, automatic tools are required to assist humans in extracting useful information from huge databases, making sense out of the data, and ultimately driving decision-making. Fayyad et al. (1996b,a) define the "nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data" as the knowledge discovery in databases (KDD) process. The KDD

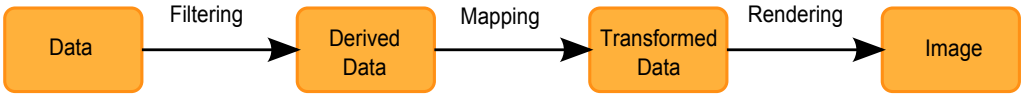



**Figure 1.2:** The visualization pipeline adapted from Haber and McNabb (1990).

process comprises many steps, ranging from data warehousing over preprocessing, transformation, and data mining to the evaluation and interpretation of the extracted patterns. The individual steps are organized in a pipeline (Figure 1.1) illustrating the flow of data in a typical KDD application. Typical data-mining techniques are classification, clustering, regression, density estimation, outlier detection, or *visualization*.

## 1.2 Visualization

Visualization can be effective as a data-mining tool because human sight is capable of perceiving more information than any of the other senses (Ware, 2004). Pre-attentive vision and an effective object-recognition systems are part of the human visual system and make visual communication a very attractive tool to convey and extract information through images. With respect to the KDD pipeline, images are the output of a data-mining algorithm and thereby represent patterns that need to be interpreted by humans in order to create knowledge. Similar to the KDD pipeline, the transformation from data to images can be modeled using another pipeline (Haber and McNabb, 1990) illustrated in Figure 1.2. While filtering is similar to selection and preprocessing in the KDD pipeline and might refer to the same procedures, mapping and rendering are more specific to visualization. Mapping describes the transformation of data to geometric objects or other renderable representations that are turned into an image in the rendering step.

Images have been used for visual communication for a long time in human history, but statisticians in the 18th century were probably the first to map numbers to graphical primitives such as length or area to learn about data (Tufte, 2001). Graphing methods such as scatterplots, histograms, pie charts, etc. have become standard tools for descriptive statistics and were shown to be highly effective (Anscombe, 1973) to "see" patterns that simple statistical models might be unable to capture. A particularly striking example was given by Anscombe (1973). For a set of four bivariate artificial datasets (Table 1.1), Anscombe computed several standard statistics such as the mean, variance, and linear regression coefficient and he visualized each pair of variables in a scatterplot (Figure 1.3). Although the statistics are equal among the four datasets, the visualizations reveal that the relations between the respective variables are different, showing quadratic relations or outliers in the data.

**Table 1.1:** Anscombe's quartet (Anscombe, 1973). All four pairs of variables result in the same descriptive statistics.

| | $x_1$ | $y_1$ | $x_2$ | $y_2$ | $x_3$ | $y_3$ | $x_4$ | $y_4$ |
|---|---|---|---|---|---|---|---|---|
| | 10.0 | 8.04 | 10.0 | 9.14 | 10.0 | 7.46 | 8.0 | 6.58 |
| | 8.0 | 6.95 | 8.0 | 8.14 | 8.0 | 6.77 | 8.0 | 5.76 |
| | 13.0 | 7.58 | 13.0 | 8.74 | 13.0 | 12.74 | 8.0 | 7.71 |
| | 9.0 | 8.81 | 9.0 | 8.77 | 9.0 | 7.11 | 8.0 | 8.84 |
| | 11.0 | 8.33 | 11.0 | 9.26 | 11.0 | 7.81 | 8.0 | 8.47 |
| | 14.0 | 9.96 | 14.0 | 8.10 | 14.0 | 8.84 | 8.0 | 7.04 |
| | 6.0 | 7.24 | 6.0 | 6.13 | 6.0 | 6.08 | 8.0 | 5.25 |
| | 4.0 | 4.26 | 4.0 | 3.10 | 4.0 | 5.39 | 19.0 | 12.50 |
| | 12.0 | 10.84 | 12.0 | 9.13 | 12.0 | 8.15 | 8.0 | 5.56 |
| | 7.0 | 4.82 | 7.0 | 7.26 | 7.0 | 6.42 | 8.0 | 7.91 |
| | 5.0 | 5.68 | 5.0 | 4.74 | 5.0 | 5.73 | 8.0 | 6.89 |
| mean | 9 | 7.5 | 9 | 7.5 | 9 | 7.5 | 9 | 7.5 |
| variance | 11 | 4.12 | 11 | 4.12 | 11 | 4.12 | 11 | 4.12 |
| correlation | 0.82 | | 0.82 | | 0.82 | | 0.82 | |
| regression line | $y = 3 + 0.5x$ | | $y = 3 + 0.5x$ | | $y = 3 + 0.5x$ | | $y = 3 + 0.5x$ | |

Anscombe's quartet can also be used to illustrate the iterative process of knowledge discovery: the initial hypothesis about the data ("there is a linear correlation between $x_i$ and $y_i$") used to build a statistical model (the Pearson correlation) has been invalidated using scatterplots. Now, the visualizations could be used to create new hypotheses and models ("there is a quadratic correlation", "outliers exist") of the data. The iteration between hypothesis generation, data modeling and description, and hypothesis testing was termed exploratory data analysis (EDA) by Tukey (1977). His work is often referred to as a milestone in visualization-assisted data analysis, as he promoted to use visualization explicitly as an analysis tool in addition to the traditional purpose of communicating facts. The developments in computer graphics and human-computer interface technology enabled the implementation of exploration by means of new interaction techniques that allowed one to change parameters of a visualization in real-time and get immediate feedback (Fisherkeller et al., 1975; Eick and Wills, 1995) on a computer screen. The role of interactive visualization is even more evident in *visual analytics*, which was defined as "the analytical reasoning facilitated by interactive visual interfaces" in the book edited by Thomas and Cook (2005). In this context, the *visual data-exploration model* (Keim et al., 2008) finally models knowledge discovery as an iterative process between visualization and data mining.

While the different models were created at different times, contexts, and communities, they all emphasize the importance of visualization and interactivity for knowledge discovery. However, there are some challenges to be solved in

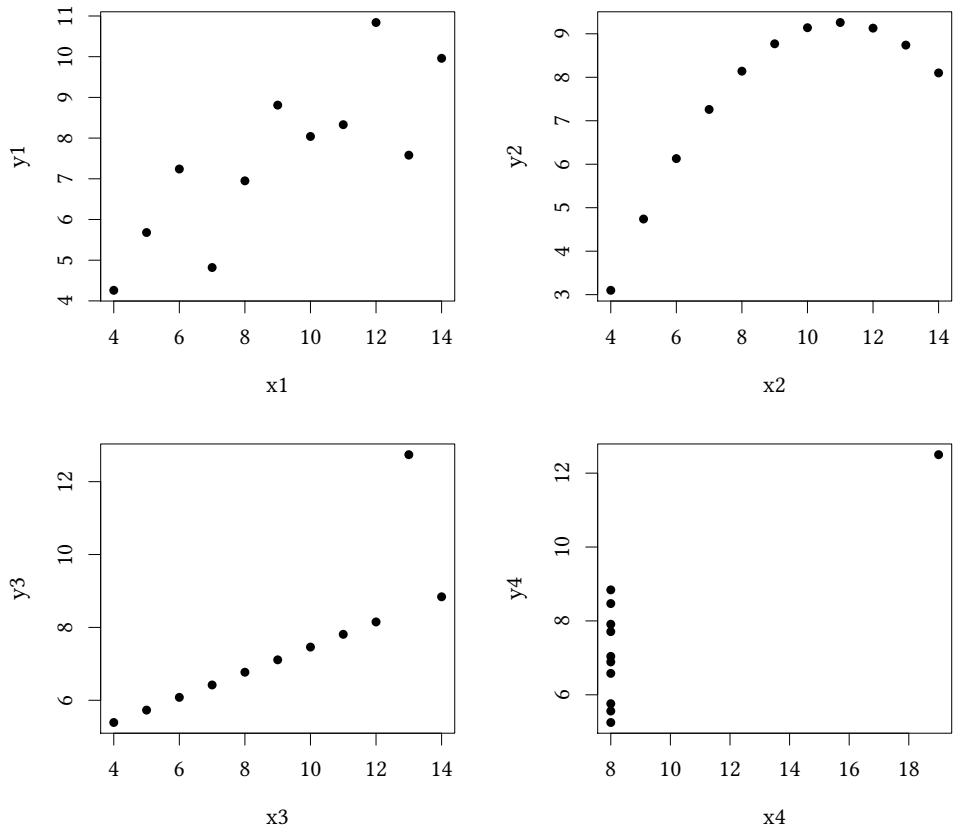



**Figure 1.3:** Visualizing the four datasets of Table 1.1 in one scatterplot each reveals the differences between pairs of variables.

order to cope with the rising amount of data to be visualized (Hibbard, 1999; Johnson, 2004; Chen, 2005; Keim et al., 2006). Efficient algorithms for the computation of a single image are required in order to achieve interactive frame rates. Visualizations need to be scalable (Chen, 2005; Keim et al., 2006) with respect to dataset size, which is particularly important for streaming applications (Chen, 2005). Further challenges are heterogeneity and dimensionality of the data. Heterogeneous data may be comprised of different data types, such as one-, two-, or multidimensional data, text, or networks (Keim, 2002). The data can further be from different sources, may be composed of different scale types (Stevens, 1946), contain missing or uncertain data, or may be defined on multiple domains. Common data types are one-, two-, or multidimensional data, text, or graphs (Keim, 2002). Data scales can be categorized into nominal, ordinal, interval, and ratio (Stevens, 1946).

This thesis addresses some of these challenges for multidimensional data of scale types ratio and interval. In addition, the data used in this work may be defined in multiple domains and describe attributes, dimensions, or variables. To

disambiguate the terms "dimension" and "variable", the terminology of Wong and Bergeron (1994) will be used where applicable. They define **multidimensional objects** as entities in Euclidean $N$-space where its geometry is of interest (such as images or volumes). In contrast, **multidimensional data** typically comprises a set of **observations** or **measurements** that are somehow related to each other. In mathematics and statistics, this relation is usually expressed with a function

$$f\colon \mathbb{R}^N \longrightarrow \mathbb{R}^M,\ \boldsymbol{x} \mapsto f(\boldsymbol{x});\ N, M \in \mathbb{N}$$

assigning values from the **independent variable** $\boldsymbol{x}$ to the **dependent variable** $\boldsymbol{y} = f(\boldsymbol{x})$. I will use the term "multidimensional" for the dimensionality $N$ of the independent variable and "multivariate" for the dimensionality $M$ of the dependent variable. With this convention, a set of five series of stock-market return values observed over time is a one-dimensional, five-variate dataset, while the temperature and pressure measured at a single time point in three-space is a three-dimensional, bivariate dataset. Both domains are further allowed to be 0-dimensional. A typical example is table-based "abstract" data, such as a set of $K$ variables describing the attributes of a car with $N = 0$ and $M = K$ or the bivariate datasets given in Table 1.1 with $N = 0$ and $K = 2$. Note that multivariate data can be represented by multidimensional Euclidean geometry. In this case, and in particular if $N = 0$, both "multidimensional" and "multivariate" can be used to describe the data.

There are many known multidimensional multivariate visualization techniques (Wong and Bergeron, 1994; Keim and Kriegel, 1996; Cleveland, 1993; Chen et al., 2008). Each of them reveals a different aspect of the data, but none succeeds in showing all of the hidden information. Keim and Kriegel (1996) classify the different approaches into pixel-based, icon-based[1], and geometric projection techniques. Mapping variables to pixels allows one to visualize large amounts of data, as a pixel is the smallest addressable entity on a computer screen. An early example of an icon-based approach are Chernoff faces (Chernoff, 1973), where variables are mapped to different attributes of a face. Projection-based techniques include the Andrews plot (Andrews, 1972), multidimensional scaling, dimensional stacking, or the grand tour. However, most of these techniques require a deep understanding of the underlying projection method and are therefore rather difficult to read for non-experts. Hence, a popular method to visualize multivariate data is to use a series of simple one-dimensional or two-dimensional projections such as histograms, boxplots, bagplots, or scatterplots for the visualization of higher-dimensional data[2]. The benefits of these arrangements are easy interpretation and high availability in many software packages. Using independent plots further allows one to apply any improvement or new technique that becomes available for the single plot also to the whole arrangement. A popular example is the scatterplot matrix (Hartigan, 1975), where

[1] or glyph-based

[2] Tufte (2001) would name such arrangements *small multiples*

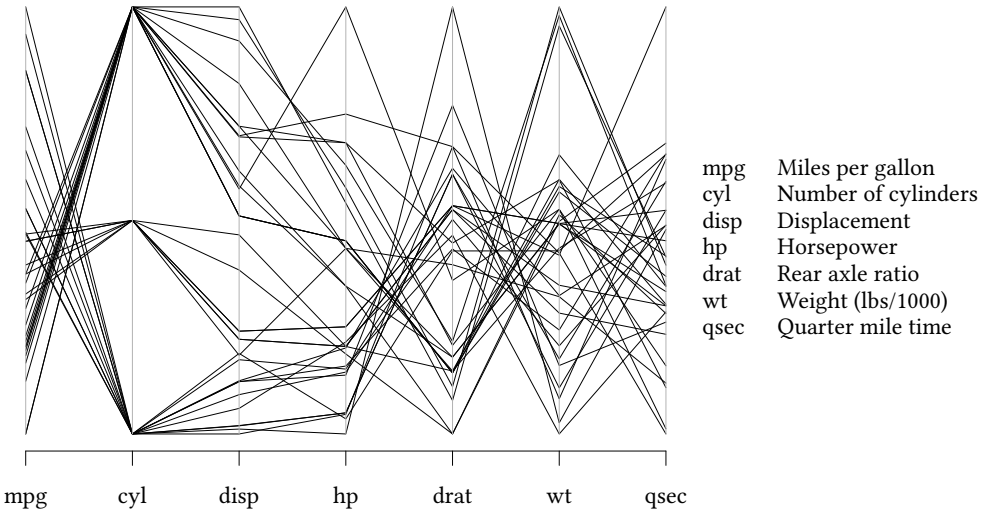



**Figure 1.4:** Parallel-coordinates visualization of 32 cars (Henderson and Velleman, 1981) with 7 attributes each. Every attribute is represented by a vertical axis, while a data point (i.e. a car) is mapped to a polygonal line that intersects every axis at the respective coordinate value, ordered from low (bottom) to high (top) values.

2D axis-aligned projections are arranged in a matrix of scatterplots. Another is parallel coordinates, which is the main subject of this thesis.

## 1.3 Parallel Coordinates

Parallel coordinates is a visualization technique for high-dimensional geometry based on projective geometry (Inselberg, 1981, 1985; Wegman, 1990). In contrast to Cartesian coordinates, axes are placed in parallel such that more than three dimensions can be visualized in a single plot. In the most common setting, $N$-dimensional points are represented as polygonal lines intersecting the axes at the respective coordinate values, as illustrated in Figure 1.4.

Polygonal lines are a common visual mapping and have been used for a long time, in particular for the visualization of time-dependent data. Well-known and widely used examples are stock-market diagrams or temperature forecasts. The similarity to such familiar diagramming techniques is certainly one of the reasons of the rising popularity of parallel coordinates both in science[3] and for the casual user (Riehmann et al., 2012). The ability to visually trace an object's

[3] The number of publications with the term "parallel coordinates" in the title has been steadily rising from 14 in the year 1991 to approximately 543 in 2011, with a total of 5.620 publications as reported by Google scholar on 15.12.2012

properties over multiple dimensions is an important building block for higher-level tasks such as finding extrema (Amar et al., 2005) and has just recently been shown to be effective in parallel coordinates (Kuang et al., 2012). As an example, it is easy to see that the cars in Figure 1.4 with a high mileage (mpg) are those with a low number of cylinders (cyl) and small displacement (disp). Knowing the geometric properties of parallel coordinates further allows one to "see" linear and non-linear relations between variables (Inselberg, 1985). Adding interaction techniques makes parallel coordinates a powerful querying tool for databases and has been shown to outperform traditional querying languages (Siirtola and Raiha, 2006). Accordingly, parallel coordinates have become a widely adopted method in various communities for exploratory data analysis (Wegman, 1990; Inselberg, 1997), visual data mining (Lee et al., 1995; Keim, 2002), and statistical graphics (Wegman and Luo, 1997; Unwin et al., 2003; Hurley, 2004; Moustafa, 2011), to mention just a few. Despite their popularity, parallel coordinates are subject to some of the visualization challenges stated above. Scalable rendering techniques and visual aggregation algorithms are required to support interactive frame rates. Other issues with respect to the visual encoding of data samples and dimensions, the rendering process, and the interpretation of the resulting image, remain to be solved. The goal of this thesis is to investigate solutions for these research questions.

## 1.4 Research Questions

There is a large body of scientific literature about parallel coordinates, including journal articles, conference papers, PhD theses, reviews, etc. While the textbook by Inselberg (2009) covers the geometric foundations of parallel coordinates and provides a reference for the mathematical background as well as many analytic proofs, a more recent survey (Moustafa, 2011) about alternative geometric modeling approaches and rendering techniques was published in the computational statistics community in 2011. Both take different approaches to parallel coordinates, but some aspects are not covered in either. These aspects include alternative geometric mappings, rendering algorithms, interaction methods, user studies, and more. A survey and classification scheme for this wide range of topics would allow for white spot analysis, identification of new research branches, and searching techniques applicable for specific problems.

Many multidimensional multivariate datasets are defined on a continuous domain within a spatial embedding. Typical examples are weather simulations or computer tomography (CT) scans, where multiple parameters such as temperature or pressure are measured in time and space. Given samples of such continuous data on some type of grid allows continuous reconstruction of in-between data values if an appropriate reconstruction or interpolation scheme is applied. Although the spatial visualization of continuous data with underlying continuous models is well known in scientific visualization, continuous

density models for histograms (Carr et al., 2006; Scheidegger et al., 2008) and scatterplots (Bachthaler and Weiskopf, 2008) have only recently been published. Similarly, the visualization of continuous data in parallel coordinates requires a generic line-density model.

Interactivity is one of the major benefits of computer-aided visual data analysis (Eick and Wills, 1995). The possibility to change parameters of a model with immediate feedback greatly facilitates the exploration of large search spaces (Fekete et al., 2008) and is essential for the implementation of the information seeking mantra (Shneiderman, 1996) or the direct manipulation of a visualization. Interactive visualization is important to achieve effective human-computer interaction, which has been on the list of the top research challenges for years (Hibbard, 1999; Johnson, 2004). In order to achieve interactivity, the computation of a visualization has to be efficient with respect to the visual complexity and the time required to render a single frame. Since the point-wise reconstruction of densities for statistical plots is very time-consuming (Heinrich and Weiskopf, 2009; Bachthaler and Weiskopf, 2009), more sophisticated techniques for rendering continuous scatterplots and parallel coordinates are required to achieve interactive frame rates even for large datasets.

Another issue with traditional parallel coordinates is that polygonal lines used for data representation are only $C^0$ continuous. They generally lose visual continuation across the parallel-coordinates axes, making it difficult to follow lines that share a common point along an axis. Also, when two or more data points have the same or similar values for a subset of the attributes, the corresponding polylines may overlap and clutter the visualization. This artifact may occur even for medium-sized datasets with a few thousand points. Also, clusters and related internal structure of the data are not represented in the geometry of the plot, except for implicit visual clustering based on the density of polylines at the axes.

As a consequence of the parallel layout of dimensions, patterns exhibited by a parallel-coordinates plot greatly depend on the order of axes (Wegman, 1990). Since there are $N!$ possibilities to order $N$ axes, selecting the "right" one is infeasible and many heuristics have been proposed to address this issue (see Section 2.3.2 for an overview). While most approaches use some sort of measure to score the set of possible orderings and select only the highest ranked for the final visualization, few methods have been presented to layout multiple parallel-coordinates plots with different orderings. In contrast to the scatterplot matrix (SPLOM), which allows one to visualize all pairwise relations of a multivariate dataset, no such layout exists for parallel coordinates.

These research questions will be addressed in the different chapters of this thesis, as outlined in the next section.

## 1.5 Outline and Contributions

This section provides a short abstract for each chapter and states the authors' contributions to the related publications. Note that all publications were co-authored by my advisor Daniel Weiskopf.

Chapter 2 presents a survey of the current state of the art of parallel coordinates. It covers the fundamental geometric models for constructing parallel coordinates and reviews methods for creating and understanding visual representations of parallel coordinates. The classification of these methods is based on a taxonomy that was established by surveying the literature. It is aimed at guiding researchers and practitioners to find existing techniques and to identify white spots that require further research. The methods covered in this chapter are further put in perspective to an established taxonomy of knowledge-discovery tasks to support users of parallel coordinates in choosing a technique for their problem at hand. The major challenges in constructing and understanding parallel-coordinates plots are discussed and state-of-the-art approaches to address these are presented. This chapter is based on (Heinrich and Weiskopf, 2013).

In Chapter 3, the concept of continuous scatterplots for the visualization of spatially continuous input data is adopted to derive a density model for parallel coordinates. Based on the point–line duality between scatterplots and parallel coordinates, a mathematical model that maps density from a continuous scatterplot to parallel coordinates and different algorithms for both numerical and analytical computation of the resulting density field are presented. In addition, it will be shown how the 2D model can be used to successively construct continuous parallel coordinates with an arbitrary number of dimensions. Since continuous parallel coordinates interpolate data values within grid cells, a scalable and dense visualization is achieved. Chapter 3 is based on (Heinrich and Weiskopf, 2009).

With the existing techniques for rendering continuous scatterplots and parallel coordinates, rendering such plots becomes prohibitively slow, especially for large datasets. Chapter 4 presents a scalable and progressive rendering algorithm for continuous data plots that allows exploratory analysis even for large datasets at interactive frame rates. The algorithm employs splatting to produce a series of plots that are combined using alpha blending to achieve a progressively improving image. For each individual frame, splats are obtained by transforming Gaussian density kernels from the 3D domain of the input dataset to the respective data domain. A closed-form analytic description of the resulting splat footprints is derived to allow precomputation of splat textures for efficient GPU rendering. The plotting method is versatile because it supports arbitrary reconstruction or interpolation schemes for the input data and the splatting technique is scalable because it chooses splat samples independently from the size of the input dataset. The effectiveness of the method is compared to existing techniques with respect to rendering performance and quality.

This work is based on (Heinrich et al., 2011a), which was co-authored by Sven Bachthaler, who contributed to the model and implementation of the algorithm for continuous scatterplots.

Chapter 5 describes a technique for the visualization of clusters using bundled curve representations in parallel-coordinates plots and presents a controlled user study evaluating their effectiveness. Replacing the traditional $C^0$ polygonal lines by $C^1$ continuous piecewise Bézier curves makes it easier to visually trace data points through each coordinate axis. The resulting Bézier curves can then be bundled to visualize data with given cluster structures. A controlled user study with 14 participants confirmed the effectiveness of curve bundling for parallel-coordinates visualization: 1) compared to polygonal lines, it is equally capable of revealing correlations between neighboring data attributes; 2) its geometric cues can be effective in displaying cluster information. For some datasets, curve bundling allows the color perceptual channel to be applied to other data attributes, while for complex cluster patterns, bundling and color can represent clustering far more clearly than either alone. Chapter 5 is based on joint work with Yuan Luo, Arthur E. Kirkpatrick, Hao Zhang, and Daniel Weiskopf (Heinrich et al., 2011b) and has been reworked by myself for publication (Heinrich et al., 2012a). Yuan Luo implemented the initial model, produced parts of the figures and conducted the user study together with Arthur E. Kirkpatrick. I re-implemented a refined version of the curve and bundling models, conducted the case study, and prepared the final version of the paper together with Arthur E. Kirkpatrick.

Chapter 6 introduces the parallel-coordinates matrix (PCM) as the counterpart to the SPLOM for parallel coordinates. Using a graph-theoretic approach, a list of axis orderings is determined such that all pairwise relations can be displayed without redundancy, while each parallel-coordinates plot can be used independently to visualize all variables of a dataset. Therefore, existing axis-ordering algorithms, rendering techniques, and interaction methods can easily be applied to the individual parallel-coordinates plots. The value of the PCM is demonstrated with several case studies and it is shown how it can serve as an overview visualization for parallel coordinates. Existing focus-and-context techniques are applied in an interactive setup to support a more detailed analysis of multivariate data. Chapter 6 is based on an extended version of (Heinrich et al., 2012b) which was co-authored by John Stasko, who helped writing the paper.

The next chapter presents applications of parallel coordinates and the methods developed in the previous chapters from different domains. Density-based parallel coordinates are used to guide the selection of parameters for a network model of proapoptotic signaling in cancer cells. Interactive computation of statistics and cluster visualization methods are used for the visualization of gene expression data in parallel coordinates. The application of continuous parallel coordinates is demonstrated for computational fluid dynamics (CFD) and pose estimation with motion-capture data. This chapter is based on (Heinrich and

Weiskopf, 2009; Dietzsch et al., 2009; Hasenauer et al., 2012). The paper (Dietzsch et al., 2009) was reworked from my diploma thesis and was co-authored by Janko Dietzsch, who supervised my diploma thesis together with the other co-authors Kay Nieselt and Dirk Bartz. In this work, I developed the software, produced the figures and parts of the case studies, and helped writing the paper. Janko Dietzsch initiated the diploma thesis together with Kay Nieselt and Dirk Bartz and helped conducting the case studies. Kay Nieselt and Dirk Bartz supervised the diploma thesis and helped to write the paper. The first author of the paper (Hasenauer et al., 2012) is Jan Hasenauer, who developed the model for heterogeneous cell populations and conducted the case study. I produced the density-based parallel coordinates figures and helped writing the manuscript. The other co-authors are Daniel Weiskopf, Frank Allgöwer, Peter Scheurich, and Malgorzata Doszczak, who contributed in writing and proof-reading the final manuscript. Finally, the chapter contains results of joint work with Sebastian Grottel and Stefan Gumhold, who provided software and conducted analysis on motion-capture data, whereas I contributed the mathematical model of continuous parallel coordinates for time series.

Original material from the publications mentioned above (Dietzsch et al., 2009; Heinrich and Weiskopf, 2009; Heinrich et al., 2011a,b, 2012a,b; Hasenauer et al., 2012) has been partly reused in this thesis.

Chapter 2

# State of the Art of Parallel Coordinates

This chapter presents a survey of recent developments of parallel coordinates focused on technical aspects to complement the mathematical background and analytic proofs provided in the textbook by Inselberg (2009).

The first section of this chapter presents a taxonomy to classify parallel coordinates techniques which will then be used to structure the following sections.

## 2.1 Taxonomy

The taxonomy given in Figure 2.1 was established from the scientific literature about various topics regarding parallel coordinates. It is aimed at identifying research directions and providing a classification scheme at different levels of abstraction. This is helpful as a guide for (i) scientists to identify areas that require further research and for (ii) users of parallel coordinates providing an overview of available techniques and possible challenges. At the top level, we distinguish *geometric models* as the theoretical foundation of parallel coordinates from the technical parts dedicated to *image generation* and *image analysis*. In addition to the taxonomy, we identified a set of challenges a user might be faced with when working with parallel coordinates. We summarize these challenges and provide links to the respective sections of this chapter and to the literature in order to address them. Finally, we present selected applications by domain to give examples of the wide range of data types visualized with parallel coordinates.

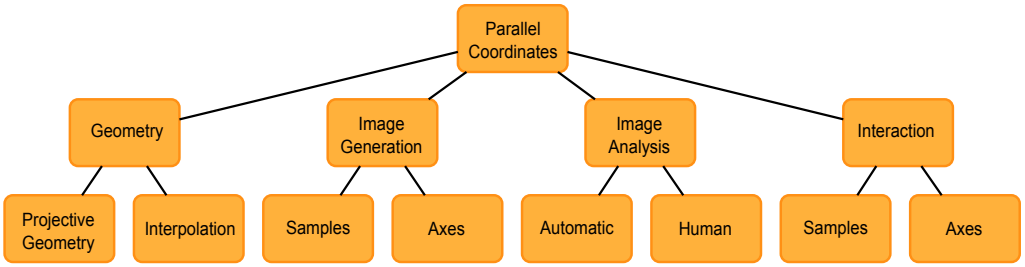



**Figure 2.1:** Taxonomy of topics for parallel coordinates in the scientific literature. The first-level nodes each represent a section in this chapter, where the scope and definition of the respective topic will be explained in detail.

We use an established taxonomy (Fayyad et al., 1996a) to relate the techniques covered in the following sections to a set of high-level tasks that support knowledge discovery in databases:

**Classification** is the task of mapping data samples to a set of predefined classes. A typical technique in interactive visualization environments that supports the classification of samples is brushing (Section 2.5.1). Brushing is typically used to select data points that are then subject to further processing, such as learning a classifier (Avidan and Avidan, 1999; Tam et al., 2011).

**Regression** is a common task for predicting the values of a dependent variable with respect to one or more independent variables. Parallel coordinates can be used for visual regression (Wegman and Luo, 1997) or to visualize statistical properties of regression models (Unwin et al., 2003; Dietzsch et al., 2009; Steed et al., 2009).

**Clustering** is the identification of sets of data items exhibiting similar characteristics. There is a wide range of automatic clustering techniques that typically depend on the similarity measure being used. Parallel coordinates can be used for "visual clustering", i.e. to find groups of similar points based on visual features such as the proximity of lines or line density. Another application for which parallel coordinates are frequently used is the visualization of precomputed clusters and their characteristics, typically using color or geometry-based visual cues. Chapter 5 presents a geometry-based approach for the visualization of clusters using curves and bundles.

**Summarization** refers to the computation of aggregated data and usually involves loss of information. Visualization is considered to be a summarization technique in KDD because it requires multivariate data to be projected to two dimensions. From a visualization viewpoint, the presentation of an overview is what best describes the summarization task. This is an important task and the starting point of the information-seeking mantra (Shneiderman, 1996). There are many approaches to show aggregated information in parallel coordinates, either as additional visual items, or by representing sets of items using alternative

visual encodings such as envelopes of lines (Inselberg, 2009) or density (Miller and Wegman, 1991; Heinrich and Weiskopf, 2009).

**Dependency modeling** is the process of establishing qualitative or quantitative dependencies between variables. Linear correlation between two variables is the most common dependency that can be visualized in parallel coordinates as a result of the point–line duality. The quantification of dependencies is an important measure for determining the relative importance of dimensions that can be used to order axes in parallel coordinates. The axis-ordering problem is discussed in more detail in Sections 2.3.2, 2.5.2, and 2.6.2. Chapter 6 further presents an alternative approach to ordering axes of a single parallel-coordinates plot.

**Change and deviation detection** includes the detection and visualization of outliers or other anomalies of the data with respect to some previously known measure. For example, data samples can be classified as outliers using a density estimation (Novotny and Hauser, 2006) based on parallel coordinates of the raw data. The detection of abnormal behavior using parallel coordinates is also an important task in process control applications (Dunia et al., 2012).

## 2.2 Geometry

A coordinate system provides a scheme for locating points given its coordinates and vice versa. The choice of coordinate system is therefore an important step in visualizing data, as it transforms the geometry representing the data that is being visualized. With coordinate transformations, straight lines (e.g. in Cartesian coordinates) can be mapped to curves (e.g. in polar coordinates) or points (e.g. in parallel coordinates). The choice of coordinate system determines the patterns exhibited by a visualization to a large part and therefore it is important to know how to "read" it. After introducing the notation used in this work, the construction of parallel coordinates is briefly described and two models that can be used for the transformation of data points from Cartesian coordinates to parallel coordinates are discussed. Parallel coordinates can be used to visualize geometry that represents data in multiple domains. Here, the term "domain" is used as a synonym for the domain of a function, i.e. the set of values for which a function is defined. Some domains will be used frequently and are thus assigned a meaningful name as well as consistent labels to help the reader connect a symbol used in an equation to the respective domain. The notation of Inselberg (2009) is adopted to distinguish between Cartesian and parallel coordinates with respect to the following domains (see also Figure 2.2):

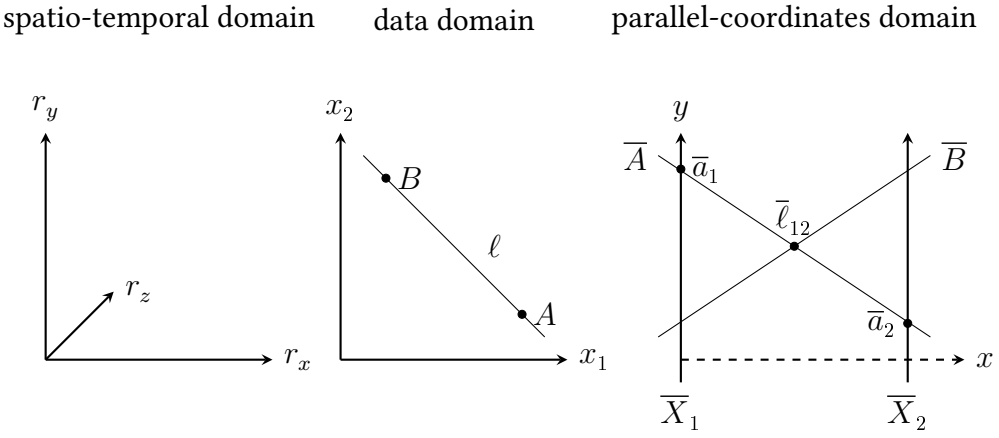



**Figure 2.2:** The notation used for different domains. The spatio-temporal domain (left) describes events in space and time with up to 4 dimensions. Many datasets describing abstract data are defined in the data domain (center) with a finite number of dimensions. The parallel-coordinates domain (right) refers to the $xy$-plane in Cartesian coordinates that is used to construct a parallel-coordinate system. See Section 2.2 for an explanation of the point–line duality illustrated above.

- The **spatio-temporal domain** represents the set of four-dimensional real values $\mathbb{R}^4$ describing events in space and time as well as any projection thereof to lower-dimensional subspaces (such as time only). Events are represented by data points referred to as *spatial*, *temporal*, or *spatio-temporal* data. A point $P = (r_x, r_y, r_z, t) \in \mathbb{R}^4$ is denoted using $xyz$-coordinates plus $t$ for the time dimension. Lines and curves are denoted with lowercase letters. The vector $\boldsymbol{p} = (r_x, r_y, r_z, t)^{\mathrm{T}}$ is also lowercase with bold typeface.
- The **data domain** represents the set of $N$-dimensional real values $\mathbb{R}^N$, $N \in \mathbb{N}^+$. Data defined in the data domain usually depicts *non-spatial* or *abstract* data such as *observations* drawn from random variables. The position of points $X = (x_1, x_2, \ldots, x_N)$ in the data domain is determined using indexed coordinates, as $N$ may take any natural number greater than zero. Unless stated otherwise, indexed lowercase letters denote the respective *dimension*, such that $x_1$ refers to the first dimension of the data domain. Vectors are lowercase with bold typeface.
- The **parallel-coordinates domain** is represented by the $xy$-plane in $\mathbb{R}^2$. It is of special interest as its representation in Cartesian coordinates enables the construction of parallel coordinates, for which it forms the *embedding coordinate system*. A point $\overline{\ell} = (x, y)$ is represented by a lowercase letter with a bar while capital letters with a bar denote lines. This notation was proposed by Inselberg (2009) to emphasize the *dualities* between data domain and parallel-coordinates domain.

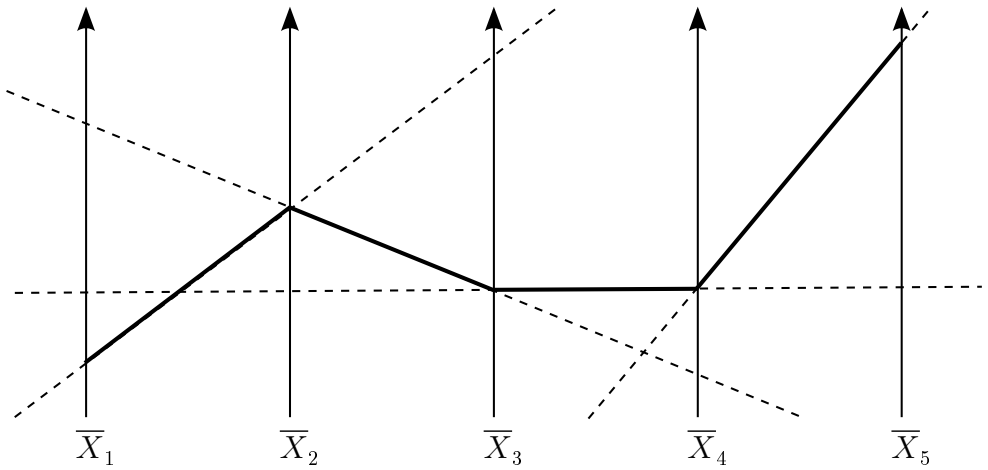



**Figure 2.3:** Constructing parallel coordinates with five dimensions represented by $N = 5$ vertical lines. Points in the plane are represented by lines joining the corresponding coordinates at the respective axes. Typically, only the line segments between the axes are drawn (represented by the bold polyline).

Note that many datasets in data mining and statistics are (zero-dimensional) multivariate data, as they are described exclusively by points in the data domain and have no spatial or temporal embedding.

### 2.2.1 Constructing Parallel Coordinates

Parallel coordinates are constructed by placing axes in parallel with respect to the embedding 2D Cartesian coordinate system in the plane (the parallel-coordinates domain). While the orientation of axes can be chosen freely, the most common implementations use horizontal (parallel to the $x$-axis) or vertical (parallel to the $y$-axis) layouts. The choice of layout depends on the number of axes, the range of the data, the dimensions of the screen, and the personal preference. For reasons of simplicity and consistency, vertical axes will be used throughout this document unless stated otherwise. For $N$-dimensional geometry, this results in $N$ copies of the $y$-axis

$$\overline{X}_i : x = d_i,\ i = 1, 2, \ldots, N$$

where the $N$-vector $\boldsymbol{d_N} = (d_1, d_2, \ldots, d_i, \ldots, d_N)^{\mathrm{T}}$ is used to denote the axis spacing as the distance of the $i$-th axis to the $y$-axis at $x = 0$. With this setting, $\frac{N(N-1)}{2}$ *pairs of axes* are obtained that will also be referred to as *segments*. Note that for a given $\boldsymbol{d_N}$, there are $N - 1$ *adjacent* pairs of axes, as illustrated in Figure 2.3. For a discussion of the order to choose for the axes, please refer to Sections 2.3.2 and 2.6.2.

### 2.2.2 Projective Plane Model

The *point–line duality* in the plane (Inselberg, 1985) is only briefly summarized here. A more detailed description including analytic proofs and the representation of hyperplanes and $p$-flats in $\mathbb{R}^N$ are given elsewhere (Inselberg, 1985; Inselberg and Dimsdale, 1990; Wegman, 1990; Inselberg, 2009).

For $N = 2$, let $\boldsymbol{d}_2 = (0, d)$ describe a two-dimensional parallel-coordinate system as in Figure 2.2. Then, a point $A = (a_1, a_2) \in \mathbb{R}^2$ of the corresponding data domain is represented in parallel coordinates by the line joining $(0, a_1)$ and $(d, a_2)$

$$\overline{A}: y = \frac{a_2 - a_1}{d} x + a_1, \; d \neq 0. \tag{2.1}$$

A set of points all located on the line

$$\ell: x_2 = m x_1 + b$$

is represented by a set of lines in parallel coordinates that intersect at the *indexed point*

$$\overline{\ell}_{12} = \left( \frac{d}{1-m}, \frac{b}{1-m} \right), \; m \neq 1.$$

Here, indexes denote axes or dimensions, and $\overline{\ell}_{ij}$ is a point in the $\overline{X}_i \overline{X}_j$ coordinate system. Similarly, points $\overline{p}_i$ with a single index are always located *on* the corresponding axis $\overline{X}_i$. For the sake of clarity, indexes will be omitted if the corresponding dimensions are obvious from the context, in particular for discussions of two-dimensional parallel-coordinate systems.

Note that the horizontal position of $\overline{\ell}$ only depends on the axis spacing and the slope of $\ell$. For the common case $d > 0$, $\overline{\ell}$ is located

- left of $\overline{X}_1$ if $m > 1$,
- right of $\overline{X}_2$ if $1 > m > 0$, and
- between $\overline{X}_1$ and $\overline{X}_2$ if $0 > m$.

So far, this formulation provides a mapping of points to lines and vice versa for all lines in the data domain with $m \neq 1$ and for all lines in the parallel-coordinates domain that are not vertical, such as the axes. In order to resolve those special cases and complete the duality, both the data domain and the parallel-coordinates domain are considered *projective planes* $\mathbb{P}^2$ that allow us to map the line $\ell: x_2 = x_1 + b$ with $m = 1$ in the data domain to the *ideal point* $\overline{\ell}_\infty$ in parallel coordinates where the set of parallel lines with slope $b/d$ intersect. Likewise, the vertical line $\overline{P}_m^\infty: x = \frac{d}{1-m}$ in parallel coordinates maps to the set of parallel lines (or the ideal point) $P_m^\infty$ with slope $m$ in the data domain. Figure 2.4 illustrates ideal points in both domains.

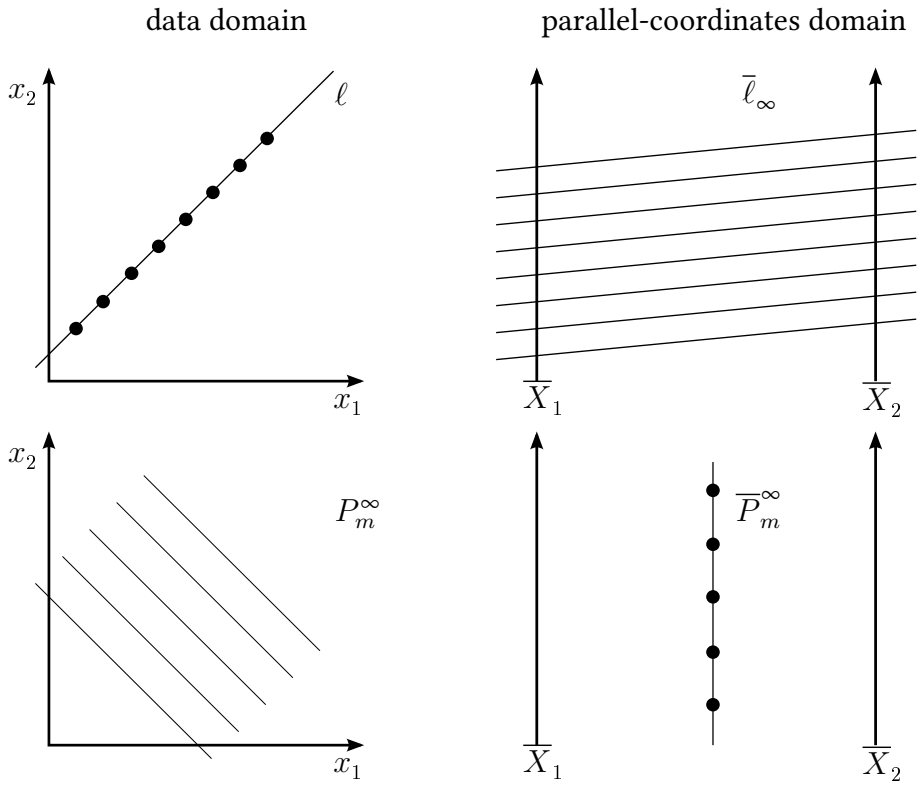



**Figure 2.4:** The line with slope $m = 1$ in the data domain is mapped to the ideal point $\overline{\ell}_\infty$ in parallel coordinates (top). The vertical line $\overline{P}_m^\infty : x = \frac{d}{1-m}$ in parallel coordinates is represented by the ideal point $P_m^\infty$ with slope $m$ in the data domain. Both domains are considered projective planes.

Based on the point–line duality, other mappings can be expressed using the *envelope* of lines in parallel coordinates. For example, $\overline{\ell}$ is the envelope of all intersecting lines and is dual to the line $\ell$ as shown above. Inselberg further uses envelopes to establish a *curve–curve duality* between Cartesian and parallel coordinates. Here, a curve $c$ is mapped point-wise from the data domain to lines in the parallel-coordinates domain resulting in the *line-curve* $\overline{c}$. The envelope of the line-curve now describes a point-curve in parallel coordinates. As an example, ellipses in Cartesian coordinates are mapped to hyperbolas in parallel coordinates, as can be seen in Figure 2.5. The ellipse–hyperbola duality has implications for the visualization of Gaussian distributions (Miller and Wegman, 1991; Feng et al., 2010; Heinrich et al., 2011a) in parallel coordinates. Another duality that has implications for brushing (see Figure 2.9, page 32) is the *rotation–translation* duality. Translating a point in parallel coordinates along the $x$-axis changes the slope of its dual line in the data domain, and vice versa. Similarly, rotating a line in parallel coordinates about a point results in the dual point to move along the line dual to the point of rotation. Please refer to Inselberg (2009) for details.

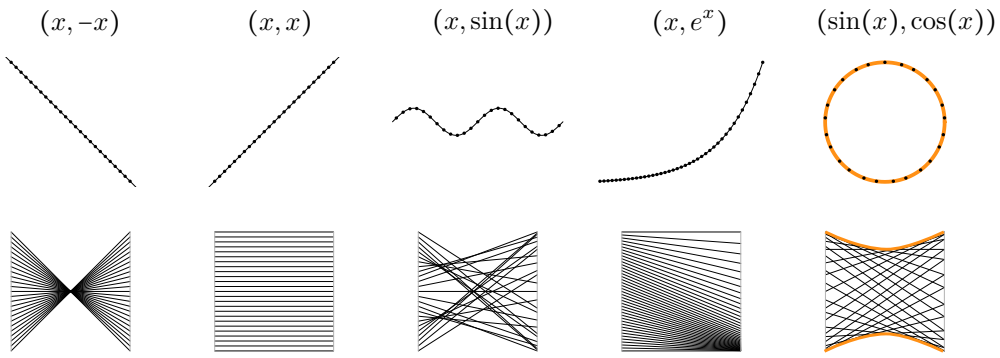



**Figure 2.5:** Common patterns in Cartesian coordinates (top) and their dual representation in parallel coordinates (bottom). The envelope of lines is highlighted for the ellipse–hyperbola duality.

### 2.2.3 Interpolation Model

Given $N$ parallel axes, the polyline that is typically used to represent a point $A \in \mathbb{R}^N$ can also be obtained using a piecewise linear interpolation of the respective indexed points $\overline{a}_i,\ i = 1, 2, \ldots, N$ located on the axes. For example, the line $\overline{A}$ in Figure 2.2 can be computed by linearly interpolating the points $\overline{a}_1$ and $\overline{a}_2$.

In analogy to Section 2.2.2, let $N = 2$ and $\boldsymbol{d}_2 = (0, d)^{\mathrm{T}}$. Then, Equation (2.1) for the representation of a point $A = (a_1, a_2)$ in parallel coordinates can also be written as

$$\overline{A}\colon y = \frac{1 - x}{d} a_1 + \frac{x}{d} a_2,\ x \in [0, d]. \tag{2.2}$$

The interpolation model allows for a wide range of different visual mappings from points in Cartesian coordinates to lines and curves in parallel coordinates, as any scheme that interpolates the indexed points $\overline{p}_i$ at the axes can be employed (see Section 2.3.1 for an overview on curves). For example, the interpolation model with linear interpolation can be used to produce the same patterns as in Figure 2.5 and it can be shown that a line in Cartesian coordinates is always mapped to a point in parallel coordinates, regardless of the interpolation model applied (Moustafa, 2009). See (Moustafa, 2011) and references therein for a more detailed discussion of the interpolation model and its properties.

## 2.3 Image Generation

For multivariate data with $N > 2$, $N$ axes are placed in parallel as described in Section 2.2.1. Applying the point–line duality to an $N$-dimensional point for every adjacent pair of axes results in $N - 1$ lines (dashed in Figure 2.3), each

representing a projection of the point to the corresponding plane. Restricting the mapping to segments results in a *polyline* intersecting all axes at the respective coordinates (bold in Figure 2.3) and constitutes the most common visualization for $N$-dimensional points in parallel coordinates. In terms of the visualization pipeline (Haber and McNabb, 1990), the dashed-line representation and the polyline representation constitute different geometric mappings. Further mapping and rendering techniques for *image generation* are presented in this section.

Many parallel-coordinates visualizations are composed of several *layers*, each of which may be computed independently. While we could consider using one layer for every line or geometric object, we will distinguish only two main layers here: one layer for the *data points* (which are typically mapped to polylines) and one for the axes. Other frequently used layers are:

- brushes or any other object used for interaction with the plot,
- axis overlays such as boxplots or ellipses,
- any other geometry that is mapped to the final image.

A *parallel-coordinate system* is usually visualized using the axis layer only. A *parallel-coordinates plot* is a visualization of the sample layer with optional axis layer. A *composite parallel-coordinates plot* is a parallel-coordinates plot with any additional layer as described above.

In the following, different mapping and rendering approaches for the two main layers are described.

### 2.3.1 Samples

This section discusses various visual encodings in the parallel-coordinates domain for $N$-dimensional data points (defined in the data domain). It is important to note that the geometric mappings presented in the following are the objects used for visualization in the final parallel-coordinates plot and do not refer to objects in the $N$-dimensional data domain[1]. Also note that, with some exceptions, most of the mappings are constructed using one of the models described in Section 2.2.

The following subsections describe two fundamentally different approaches to the visualization of a set of data points. **Geometry-based** approaches use geometric objects such as points, lines, curves, or polygons as a mapping for individual data samples or groups of samples. The analysis task thereby varies from the visualization of correlation over the detection of outliers to the characterization of clusters over multiple dimensions, among others.

[1] For a discussion of the representation of multidimensional lines, planes, $p$-flats, curves, etc. in parallel coordinates, please refer to the respective chapters in Inselberg (2009)

Density or density estimates of the input data can be visualized implicitly or explicitly. Implicit density visualizations are based on the proximity of geometric objects. Depending on the sample size and the shape of the (true, but typically unknown) distribution, geometry-based visualizations represent both the raw data and the respective density or density estimate. Due to the potential overlap of visual items, however, these approaches may fail to convey useful information, in particular if the data is very large. In contrast, **density-based** approaches explicitly visualize a continuous density function of the underlying data instead of discrete samples. Figure 2.6 illustrates examples of explicit density visualizations for univariate, bivariate, and multivariate data.

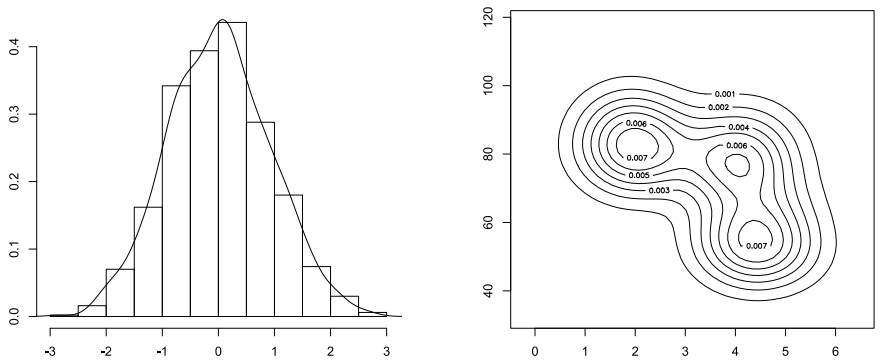
Computing and visualizing densities is a typical summarization task, as it is used to show aggregated information about the raw data. In addition, the estimation of a probability density is closely related to the clustering task (Fayyad et al., 1996a).

### Points

Points in the parallel-coordinates domain may represent points, lines, planes, hyperplanes, or $p$-flats with $p \in \mathbb{N}^+$ of the data domain. In order to distinguish different point-representations, Inselberg (2009) uses the notation of *indexed points*. Points with one index represent one-dimensional projections of the data domain. An $N$-dimensional point $P$ in the data domain is mapped to $N$ indexed points $\bar{\ell}_i = (d_i, p_i)$ in the parallel-coordinate system. This can be used to represent marginal distributions on the axes, similar to a set of $N$ one-dimensional scatterplots (also referred to as dot plot). Points with two indices $\bar{\ell}_{ij}$ represent lines of the respective $x_i x_j$-plane in the data domain, as described by the point–line duality in Section 2.2.2. For the generalization of this scheme to $p$-flats, see Inselberg (2009, Chapter 5).

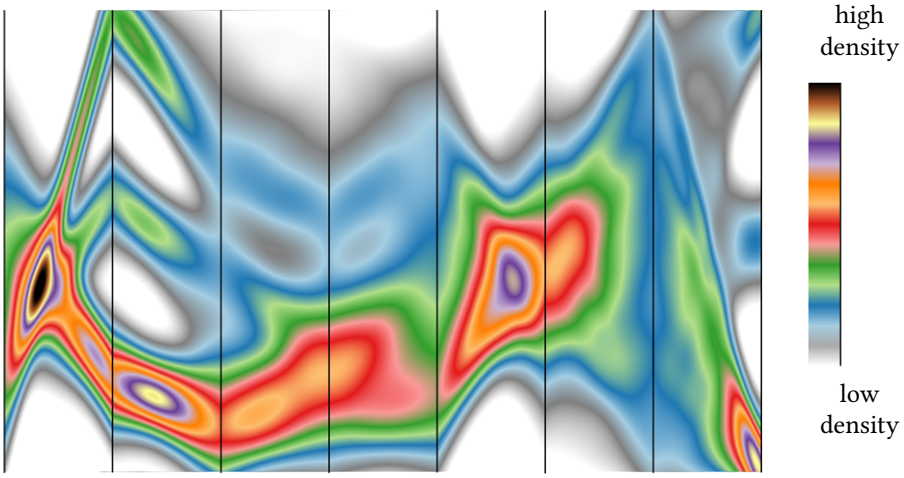
The density of points with two indices can be used to detect lines in images (Inselberg et al., 1997; Dubska et al., 2011). Here, the data domain represents a grayscale image composed of pixels that are mapped to lines in a parallel-coordinate system with two axes for the horizontal and vertical pixel coordinates. Then, the density of intersecting points is evaluated, where high density regions or clusters are used as an indication of a line in the corresponding image. To capture lines with positive slopes, the first axis (e.g. for the horizontal position of pixels) is negated and appended to the parallel-coordinate system.

To combine the advantages of scatterplots and parallel coordinates, points have also been used to embed scatterplots between adjacent axes (Yuan et al., 2009; Holten and Wijk, 2010). The respective point coordinates are determined by rotating either one of the axes by $90$ degrees (Yuan et al., 2009) or both axes by $45$ degrees (Holten and Wijk, 2010) to obtain the corresponding Cartesian coordinate system.




(a) histogram

(b) contour plot




(c) density-based parallel coordinates

**Figure 2.6:** Density visualizations for (a) univariate, (b) bivariate, and (c) multivariate data. The histogram shows a density estimate with discrete *bins* and an overlayed continuous density estimate reconstructed using a Gaussian kernel. The contour plot for bivariate data shows isolines for a density estimated from point data using a 2D Gaussian kernel. Density-based parallel coordinates are computed from pairs of 2D density fields. A colormap was applied to the density in parallel coordinates.

### Lines

Due to the point–line duality, lines are the most common visual mapping for parallel coordinates. As described in Section 2.2, $N$-dimensional points are represented with a polygonal line intersecting each of the $N$ axes at the respective coordinates (Figure 2.3 illustrates this scheme).

### Curves

Using the interpolation model introduced in Section 2.2.3, the polyline resulting from connecting lines at the axes can also be described as a non-smooth, $C^0$ continuous curve that is not differentiable at the axes. Several authors proposed using smooth, $C^n$ continuous curves with $n > 0$ to

1. visualize multiple, and higher-order correlations (Theisel, 2000; Moustafa and Wegman, 2002),
2. facilitate line tracing (Moustafa and Wegman, 2002; Graham and Kennedy, 2003; Yuan et al., 2009; Holten and Wijk, 2010; Heinrich et al., 2012a),
3. enable the detection of overplotted line segments (Graham and Kennedy, 2003), and
4. visualize clusters using bundling (Zhou et al., 2008; McDonnell and Mueller, 2008; Heinrich et al., 2012a).

Theisel (2000) use piecewise cubic B-splines to visualize multiple pairwise correlations by choosing two "main axes" with an arbitrary number of additional axes placed in-between. Andrews plots (1972) can be obtained using an interpolation model with Fourier bases (Moustafa and Wegman, 2002). Other functions forming an orthonormal basis can be used to emphasize quantization effects on the axes and to detect second-order structures (Moustafa and Wegman, 2002). Piecewise quadratic and piecewise cubic interpolation models were proposed (Graham and Kennedy, 2003; McDonnell and Mueller, 2008; Holten and Wijk, 2010) to enforce tangents at a point $\overline{p}_i$ to be parallel to the line $\overline{\overline{p}_{i-1}\overline{p}_{i+1}}$. These models also resolve ambiguities if curves intersect axes orthogonally (Holten and Wijk, 2010). Many interactive implementations further add a parameter (Holten and Wijk, 2010; Heinrich et al., 2012a) to control the amount of smoothing. All these techniques guarantee curve smoothness and mitigate the line-tracing problem (see Section 2.6.3) by assigning different trajectories to curves that intersect at an axis.

### Bundling

Curves can also be used for edge bundling (Holten, 2006) to visualize clusters in parallel coordinates (Zhou et al., 2008; McDonnell and Mueller, 2008; Heinrich

et al., 2012a). Here, a *bundle* represents all data samples belonging to a cluster defined a-priori (McDonnell and Mueller, 2008) or emerging from the bundling algorithm (Zhou et al., 2008). Bundles can be visualized *implicitly* as a set of curves (McDonnell and Mueller, 2008; Zhou et al., 2008; Heinrich et al., 2012a) or *explicitly* using polygons (McDonnell and Mueller, 2008). In both cases, the visual signature of a bundle is constructed by "attracting" one (McDonnell and Mueller, 2008) or more (Zhou et al., 2008) control points from individual curves toward a point that represents the respective cluster, such as the cluster centroid (McDonnell and Mueller, 2008; Heinrich et al., 2012a). Chapter 5 describes one of the curve-models (Heinrich et al., 2011b) and presents an evaluation of a bundling method for parallel coordinates.

### Polygons

Another mapping that readily supports the *summarization task* is from sets of points in the data domain to envelopes and quadrilaterals in the parallel-coordinates domain. This is also an example of the *explicit visualization* of sets or clusters, where the visual mapping for a group of data points is chosen prior to the rendering step and usually involves one or more filtering steps from the raw data (such as clustering the data). Given a set of data samples in the data domain contained in an $N$-dimensional convex hypersurface, Inselberg (1985) suggests drawing the envelope of the respective polygonal lines in parallel coordinates. Then, any point interior to the hypersurface in the data domain is represented by a polyline that is also interior to the envelope in parallel coordinates. Fua et al. (1999b) render convex quadrilaterals resembling the axis-aligned bounding box of a cluster in the data domain. The same geometric mapping can be used with different shadings for classification rules (Han and Cercone, 2000), fuzzy points (Berthold and Hall, 2003), sets and subsets (Andrienko and Andrienko, 2004), contingency tables (Bendix et al., 2005; Kosara et al., 2006), binned data (Novotny and Hauser, 2006), multivariate time series (Johansson et al., 2007), and quartiles (Moustafa, 2011). Non-convex quadrilaterals can also be used to indicate negative correlations (Johansson et al., 2007; Zhang et al., 2012). Other mappings, in shape similar to envelopes, evolved from bundling (McDonnell and Mueller, 2008) and the visualization of line densities (see next section).

### Density

In many cases, the density function

$$\sigma: \mathbb{R}^N \longrightarrow \mathbb{R}$$

describing the distribution of a (possibly multivariate) data sample cannot be reconstructed, but has to be estimated from data. A well-known probability density estimate for a univariate dataset $X = (x_1, x_2, \ldots, x_n)$ is the histogram (the term "histogram" is used both for a function representing a density estimate

as well as for the visualization using rectangular "bins" (Figure 2.6), as proposed by Pearson (1895)) that Scott and Sain (2005) define as

$$\sigma(x) = \frac{v_k}{nh},\ x \in B_k, \tag{2.3}$$

where $h$ is the (uniform) bin width for all bins $B_k, k \in \mathbb{N}$ and $v_k$ is the number of observations falling in bin $B_k$. The histogram illustrated in Figure 2.6 (left) was computed using Equation (2.3). For the bivariate case, $\sigma$ is defined on a two-dimensional domain $\sigma\colon \mathbb{R}^2 \longrightarrow \mathbb{R}$ and the bins $B_k$ represent areas (usually rectangular) instead of intervals. The process of constructing such a 2D histogram is sometimes also referred to as *binning*. For visualization in the data domain, binned data is usually mapped to color. Hence, the model of a histogram is based on counting the number of samples per line segment in 1D or per area in 2D. The density as computed in Equation (2.3) can be thought of as the probability of observing a data point in $B_k$, and the total probability of observing a point in any bin equals one. A more general density estimate (Wegman and Luo, 2002) for multivariate data and arbitrary *kernels* reads:

$$\sigma(\boldsymbol{x}) = \frac{1}{nh^N} \sum_{i=1}^{n} K\left(\frac{\boldsymbol{x} - \boldsymbol{x_i}}{h}\right), \tag{2.4}$$

where $K$ is the respective kernel with *bandwidth parameter* $h$. Figure 2.6 illustrates the histogram with discrete bins and a continuous density estimate using Equation (2.4) with Gaussian kernels.

Similar to the implicit point-density model for Cartesian coordinates, a *line density* is implicitly encoded in parallel coordinates by the proximity of lines. A common approach to compute the density

$$\varphi\colon \mathbb{R}^2 \longrightarrow \mathbb{R},\ (x, y) \mapsto \varphi(x, y)$$

explicitly in parallel coordinates at any given *point* $\bar{\ell} = (x, y)$ is to employ the same binning strategy (Novotny, 2004; Johansson et al., 2005b; Holten and Wijk, 2010; Dasgupta and Kosara, 2010) as for the point-wise density computations in scatterplots. Here, the number of lines intersecting a 2D bin is evaluated instead of the points contained in the bin. Note that rectangular bins should not be confused with *pixels* (Smith, 1995).

Binned densities can also be transformed to parallel coordinates using a scattering approach: quadrilaterals are rendered instead of lines, each representing a rectangular bin mapped from the data domain (Artero et al., 2004; Novotny and Hauser, 2006). Here, the shading of quadrilaterals either reflects the density of the respective 2D bin (constant shading) or can be interpolated between one-dimensional density estimates corresponding to the respective axes (Rodrigues et al., 2003). The final density at a point in parallel coordinates is then computed as the sum over the sample contributions. This is typically implemented using additive blending.

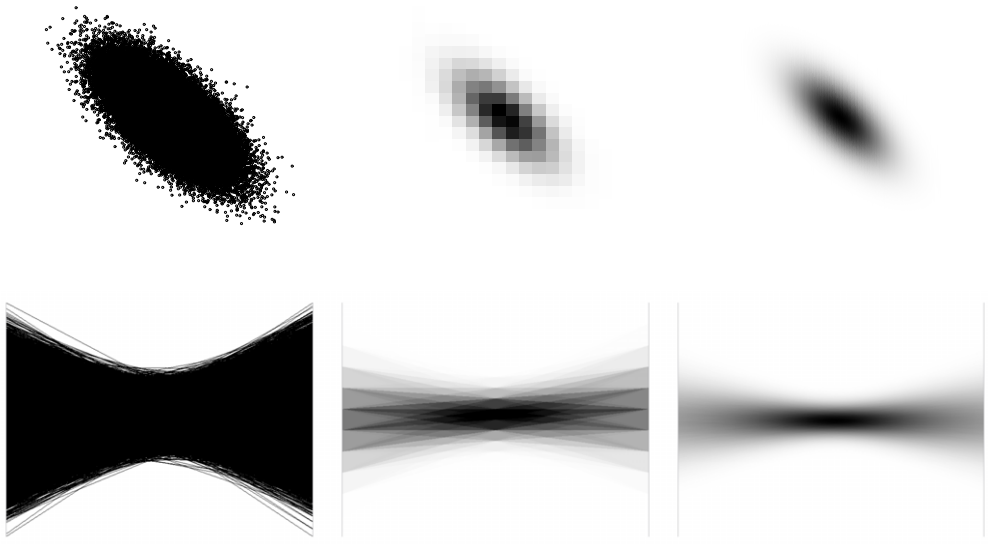


**Figure 2.7:** A sample of 100 000 observations drawn from a bivariate normal distribution (top, left) rendered using traditional parallel coordinates (bottom, left), binned parallel coordinates (bottom, center), and the line-density model proposed by Miller and Wegman (1991) (bottom, right). The dual pattern for each approach in Cartesian coordinates is shown in the top row.

Alpha-blending is used by many authors (assuming alpha-blending whenever the term *opacity* is used and the exact blending mode is not mentioned). Here, the contribution of the density of a line to $\varphi$ at a point in parallel coordinates decreases exponentially with increasing number of contributions and thus does not converge to the line-density model. The advantage of this technique is that normalization is not required, as $\sigma$ is bounded and converges asymptotically to a maximum value (typically $1$ or $256$). It is further important to note that alpha-blending is non-associative, i.e. the value of $\varphi$ depends on the order of lines being rendered, for non-uniform distributions of $\alpha$.

As Miller and Wegman (1991) point out, 2D binning in the parallel-coordinates domain might result in lines being counted in multiple bins, which violates the requirements of a probability density function to integrate to one. Instead, the probability of observing a line should be equal for any horizontal position, such that line density should be based on counting lines on vertical intervals instead of two-dimensional areas. A closed-form solution for bivariate normal and uniform distributions was given by Miller and Wegman (1991). Figure 2.7 compares the traditional, constant-density, line-based rendering with binning in the data domain and the approach proposed by Miller and Wegman (1991). A density plot of the cars dataset obtained with Gaussian kernels in the data domain and the transformation to line density is illustrated in Figure 2.6. In

addition, a colormap has been applied to the density field.

The model of continuous scatterplots (Bachthaler and Weiskopf, 2008) for the mass-conserving transformation of density from the spatio-temporal domain to the data domain was also extended to parallel coordinates and is presented in detail in Chapters 3 and 4.

Using independent bivariate density estimates in the data domain for each pair of axes produces footprints with a potentially different density for every segment in parallel coordinates. As a result, the rendered primitives might not be visually traced over all axes, losing visual coherence. To accommodate for this, Moustafa (2009) quantizes densities in the parallel-coordinates domain between adjacent axes and accumulates the binned frequencies for each data point. After normalization, polylines are rendered in order of ascending cumulated frequencies, i.e. the data point with the highest overall frequency (over all 2D projections) is drawn last. Distance-based weighting schemes (Zhou et al., 2009; Holten and Wijk, 2010) were also used to construct a multidimensional density function that is then used to assign a density to polylines. Anisotropic diffusion of noise textures (Muigg et al., 2011) was employed to visualize line orientations for density-based parallel coordinates computed from discrete samples.

Finally, density can also be used to resolve ambiguities, as illustrated in Figure 2.8.

### 2.3.2 Axes

Axes are an important part of a parallel-coordinates plot that fulfill many purposes: they implicitly visualize the axis spacing $\boldsymbol{d}_N$, help an observer read off the value of a sample, and serve as a visual anchor for labels, ticks, and other overlays. Axes are usually mapped to straight lines and rendered solid and fully opaque. Labels are typically attached either at the top or at the bottom of an axis and can be rotated in order to save space. As with axes in other coordinate systems, arrows can be used to indicate the direction of increasing values. Axes are often enriched, or composited, with additional information about the respective dimension. Common examples of such overlays are histograms (Hauser et al., 2002; Geng et al., 2011; Walker et al., 2012) or boxplots (Siirtola, 2000; Theus, 2003). Other mappings for axes are curves (Qu et al., 2007; Walker et al., 2012) for the representation of polar coordinates and tag clouds (Collins et al., 2009) for the visualization of word frequencies.

### 2.3.3 3D Plots

Several approaches to rendering axes (Johansson et al., 2005a; Lind et al., 2009) and samples (Wegenkittl et al., 1997; Fanea et al., 2005; Rübel et al., 2006; Dang et al., 2010; Moustafa, 2011; Walker et al., 2012) in 3D are known for parallel coordinates. The placement of axes on a plane in a 3D world allows one to

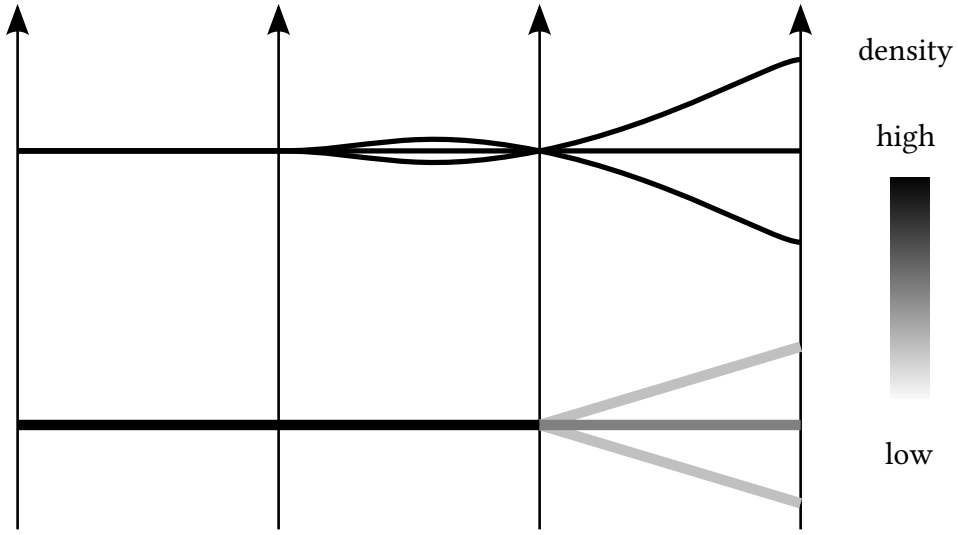



**Figure 2.8:** Some ambiguities cannot be resolved entirely using curves (as in Figure 2.12), because tangents only depend on adjacent axes. The lines appear as one in the leftmost segment using curves (top). The density representation (bottom) reveals two different densities, where the horizontal line appears darker than the other two. Assuming equal and constant densities for each sample, this means that at least two samples are contributing to the density of the horizontal line. In conclusion, the plot must be showing at least four samples instead of three as the top plot suggests.

visualize multiple 2D parallel-coordinates plots without duplicating axes. For the visualization of sets of parallel-coordinates plots, such as for time points of dynamical systems (Wegenkittl et al., 1997) or the expression of genes at different spatial positions (Rübel et al., 2006), stacking the single plots along the third axis (Wegenkittl et al., 1997; Rübel et al., 2006; Dang et al., 2010) or rotating the parallel-coordinates domain around a shifted $x$-axis (Fanea et al., 2005) was proposed. While 3D representations allow more flexibility by adding one degree of freedom to the visualization, they also introduce occlusion and distortion by projection.

## 2.4 Image Analysis

This section presents work related to parallel coordinates in an image analysis context. Here, image analysis refers to any process that uses parallel coordinates or a parallel-coordinates plot as input. Examples are the visual perception of a parallel-coordinates plot by a human observer, e.g. in a data-analysis task, or the processing by a computer algorithm, e.g. for automatic feature detection.

Some formal evaluations compare traditional parallel coordinates with other

visualizations, such as scatterplots (Li et al., 2010; Holten and Wijk, 2010; Kuang et al., 2012) or stardinates (Lanzenberger et al., 2005). It was shown that humans perform better using scatterplots than parallel coordinates in visual correlation analysis (Li et al., 2010) and cluster identification (Holten and Wijk, 2010) tasks. The former study investigated the participants' ability to estimate the Pearson correlation of two random variables in scatterplots and parallel coordinates, while the task in the latter study was to estimate the number of clusters in a dataset. The same task was shown to be effective using bundled parallel coordinates (Heinrich et al., 2012a). The performance of estimating the coordinate value of a given $N$-dimensional point at a given dimension was found to be better using parallel coordinates than scatterplots for small datasets (Kuang et al., 2012). The perception of patterns in the presence of different levels of noise was investigated by Johansson et al. (2008). They found out that patterns in parallel coordinates can be identified with a probability of $70.7\%$ if approximately $13\%$ noise was added to the signal. The patterns were created using a sample of $300$ points from five different signals, including linear and sinusoidal functions. Other studies showed that parallel coordinates are effective in querying databases (Siirtola and Raiha, 2006) and alarm filtering (Azhar and Rissanen, 2011). Finally, there is evidence that understanding patterns in parallel coordinates can be learned quickly (Siirtola et al., 2009).

Parallel coordinates have also been used for the automatic detection of lines (Inselberg et al., 1997; Dubska et al., 2011) and other features (Lehmann and Theisel, 2011) of the data domain as well as for the computation of metrics for visual abstraction (Johansson and Cooper, 2008) and for the ranking of 2D plots (Dasgupta and Kosara, 2010) (see also Section 2.3.2). Line detection in images can be realized using the density-based mapping approaches presented in Section 2.3.1. Rendering a line for every sample of a grayscale input image with the respective density results in a parallel-coordinates plot similar to the example in Figure 2.7. The density at a point in parallel coordinates now reflects the density of the dual line of the image. Note that, in order to detect lines with positive slopes (with points in the parallel-coordinates domain located to the left or right of the axes), one of the spatial axes has to be inverted and added to the plot (Dubska et al., 2011).

## 2.5 Interaction

Interaction plays an important role to enhance perception for dataset exploration and visual data mining (Ferreira de Oliveira and Levkowitz, 2003). It enables the user of a software to change parameters interactively and get immediate feedback from the system. In the KDD process, interaction allows the user to modify each step of the pipeline individually, from the acquisition of a new dataset over changing normalization parameters to defining new visualizations. According to the information-seeking mantra (Shneiderman, 1996), the user

of a data-analysis system should gain an *overview first*, with the option to get *details on demand*. The previous sections illustrate how static images of parallel-coordinates plots are used for tasks such as summarization, dependency modeling, or cluster detection. *Interactive parallel coordinates* further support these tasks and enable the exploration of a dataset.

There are many interactions possible with parallel coordinates, as any free parameter of any technique presented in the previous sections could be changed interactively. For this reason, only interactions compatible with the traditional parallel-coordinates plot are considered here, based on the geometric framework of Section 2.2. While others classified interactions with parallel coordinates by task (Andrienko and Andrienko, 2001; Siirtola and Raiha, 2006), the same taxonomy as in Section 2.3 is used here to distinguish between interactions with samples and axes.

### 2.5.1 Interacting with Samples

#### Brushing

A common interaction technique used in statistical graphics is the *brushing* of samples, which was introduced for the *masking* and *isolation* of data points in scatterplots (Fisherkeller et al., 1975). Brushing is an operation that allows the user to select a subset of samples by means of a *brush*, which originally referred to an axis-aligned rectangle for selections in scatterplots (Becker and Cleveland, 1987). The selected set of points is then used as input for subsequent operations, such as highlighting, labeling, replacing, deleting, and many more (Becker and Cleveland, 1987; Becker et al., 1987). A particularly important task supported by highlighting brushed samples is the visual linking of data samples between multiple graphical representations (*brushing and linking*), as in the scatterplot matrix (Hartigan, 1975; Becker and Cleveland, 1987). Brushing can further be direct and indirect (Martin and Ward, 1995), be composed of logical operations (Martin and Ward, 1995) or graphs (Chen, 2003), and be applied to dimensions instead of samples (Turkay et al., 2011). As most of those concepts are applicable to parallel coordinates as well, the discussion will be restricted to the geometry of brushes and methods specifically designed for parallel coordinates.

An axis in the parallel-coordinates domain represents a set of parallel lines (or the ideal point) in the data domain (Inselberg, 2009). Brushing a point on an axis is thus equivalent to the selection of a line (i.e. all points on a line for discrete data) in the data domain. In addition, these lines are perpendicular to the respective axis in the data domain, such that the brush depends only on one dimension. Accordingly, a *range* on an axis in parallel coordinates results in an interval on the respective dimension in the data domain (such as the blue and green intervals on the axes in the topmost illustration in Figure 2.9). Extending such

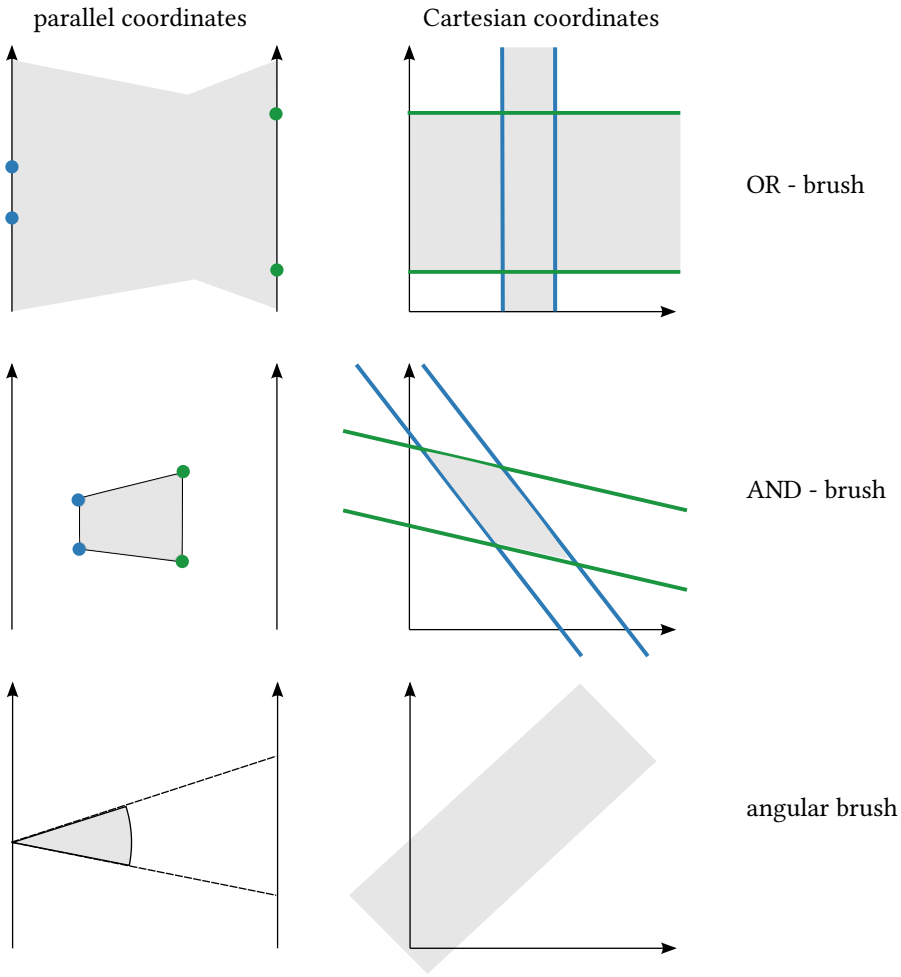



**Figure 2.9:** Brushing in parallel coordinates. Top: Axis-aligned brushing. A brush on one axis (e.g. left axis, bounded by the blue points) corresponds to a one-dimensional interval brush in the data domain (bounded by the dual lines). The union (OR) with another interval (green) results in the orange brush. The intersection (AND) of two intervals is shown in the center. Translating the blue and green points in parallel coordinates results in a rotation of the dual region in Cartesian coordinates. The bottom row illustrates the dual of an angular brush to the data domain, which corresponds to a set of ideal points $\overline{\ell}_\infty$ as in Figure 2.4.

an one-dimensional brush to multiple axes enables the construction of higher-dimensional brushes (Martin and Ward, 1995) using logical operators (Ward, 1994; Avidan and Avidan, 1999; Hauser et al., 2002) or graphs (Chen, 2003). For instance, the AND-operation can be used to subsequently build a convex polygon in parallel coordinates that is dual to a hypercube in the data domain.

By exploiting the rotation–translation duality, line-based and polygon-based brushes can also be employed in the space between axes. As indicated in Figure 2.9, translating the blue and green points in parallel coordinates results in a rotation of the corresponding area in Cartesian coordinates.

Another brush that can be used to select samples in parallel coordinates is based on the slope of lines between adjacent axes. With angular brushing (Avidan and Avidan, 1999; Hauser et al., 2002), a range of angles in parallel coordinates (e.g. relative to the horizontal) can be used to define a set of ideal points $\overline{\ell}_\infty$ as a brush. In contrast to axis-aligned brushing, angular brushing enables a line-based brush in the data domain and thus further allows for the selection of lines with positive slopes in the data domain without the need to flip axes (see also Section 2.5.2).

For large datasets, hierarchical brushes using wavelets (Wong and Bergeron, 1996) and hierarchical clustering (Fua et al., 1999b,a, 2000) were proposed. Here, brushed samples are aggregated in a balanced (Wong and Bergeron, 1996) or unbalanced (Fua et al., 1999b) tree that can be navigated in discrete steps by defining the current depth (Wong and Bergeron, 1996) or continuously with arbitrary cuts (Fua et al., 2000). Both techniques give the user control over the current level-of-detail (LOD).

Traditional brushing can be expressed as binary function assigning either $0$ or $1$ to every sample in the dataset. Smooth brushing (Martin and Ward, 1995; Hauser et al., 2002; Feng et al., 2010) uses a continuous function instead and allows one to express a certain degree-of-interest to any point (line) in the data (in parallel coordinates). However, composites are more difficult to compute using smooth brushes (Martin and Ward, 1995; Hauser et al., 2002).

Brushing in parallel coordinates can be supported by haptic feedback, e.g. by projecting a parallel-coordinate system on a mixing-board interface (Crider et al., 2007). Bimanual interaction was found to be helpful for exploration and can also be used for angular brushing with touch interfaces.

### 2.5.2 Interacting with Axes

The position of axes in a parallel-coordinates plot has a high impact on the patterns emerging from the visualization of samples, as they define the scheme for locating an individual sample in the parallel-coordinate system. Translating axes changes the order of variables and the spacing in-between. The scaling determines the range of values that intersect an axis and provides a mechanism

for flipping axes. Both operations, translation and scaling, cover a wide range of interactions that have been proposed for parallel coordinates.

### Translation

The absolute horizontal position of axes $\boldsymbol{d}_N$ in parallel coordinates is a free parameter of the visualization and does not affect the validity of the point–line duality. The relative distance between adjacent axes is usually chosen to be uniform, as this configuration puts equal emphasis on all pairwise variable relations. However, in some cases it is beneficial to move axes horizontally, e.g. to investigate a particular pattern in detail (by exploiting the additional space gained for one pair of axes if another axis is translated horizontally), or to manually rearrange the axis order. Axis translation is often implemented as a *drag-and-drop* operation, where a uniform axis spacing is reconstructed after releasing an axis.

Translating axes and associated sample coordinates in the vertical direction can be useful to align a set of axes to a common scale or a common value (Andrienko and Andrienko, 2001).

### Scaling

As with most statistical plots, patterns emerging in parallel coordinates depend on the scale of variables and axes. The default range of values represented on an axis is bounded by the minimum and maximum values of the corresponding variable, i.e. the smallest value will always intersect the axis at the bottom and the largest value at the top. While this setting allows us to see patterns in data of different units, it is not suited to compare values of equal units if the range of measurements differ between axes. Here, a uniform scale on all axes might be a better solution. Axis scaling is equivalent to applying a function to all values of the respective variable and has also been referred to as *dimension zooming* (Fua et al., 1999b). Scaling can be used to align axes to a common base (Andrienko and Andrienko, 2001), such that one sample is represented as a horizontal line. This allows the user to visually estimate the similarity of other samples with respect to a reference.

A special case of scaling is the *flipping* of axes. Flipping negates all values of the respective dimension, which has the effect of reversing the relation of positive values at the top and negative values at the bottom. As a result, the slopes of lines are also negated as well as the patterns for negative and positive relations. Hence, a set of parallel lines indicating a positive correlation is transformed to a negative correlation, which can be represented as a point in parallel coordinates. This is particularly useful for systems searching for points in a parallel-coordinates plot, e.g. for the automatic detection of lines in the data domain (Inselberg et al., 1997; Dubska et al., 2011). Here, a two-dimensional data domain is represented using three axes, say $\overline{X}_1$,$\overline{X}_2$, and $\overline{X}'_1$ in parallel coordinates, where $\overline{X}'_1$ denotes

the flipped $\overline{X}_1$. Now, the intersection of two lines will always occur within one of the segments.

## 2.6 Challenges

As we have seen in the previous sections, many decisions have to be made in order to find the "right" way to visualize (Section 2.3), analyze (Section 2.4), or interact with (Section 2.5) parallel-coordinates plots. Similarly, the research conducted in the area of parallel coordinates may be categorized by visualization or interaction techniques, analysis tasks, applications, or *challenges*. While the analysis challenge is clearly defined by a particular question or task (e.g. "find outliers in the data"), many authors motivate their work implicitly or explicitly by addressing some sort of "drawback", "limitation", or "problem" of a particular visualization. A good example of such a deficiency is "the clutter" in parallel coordinates, and the corresponding challenge is to reduce it. While there are objective measures for clutter (Ellis and Dix, 2006), a subjective quantification of clutter in practice usually depends on the context and individual experience of the observer with the respective visualization. In many cases, no particular analytical task is addressed explicitly by reducing the clutter, although diverse findings such as clusters, outliers, or other patterns can be revealed by doing so. As a consequence, many researchers were faced with the following challenges when visualizing data with parallel coordinates:

- **Overplotting** occurs in parallel coordinates if lines potentially occlude patterns in the data.
- The **order of axes** implicitly defines which patterns emerge between adjacent axes.
- The **line-tracing** problem occurs if two or more lines intersect an axis at the same position.
- **Nominal and ordinal data** such as sets and clusters have to be mapped to a metric scale before it can be visualized in parallel coordinates.
- **Time series** are special in that time points, if interpreted as dimensions, have a fixed order.
- **Uncertain** data is another challenge for visualization, and there are approaches for the visualization of uncertainty in parallel coordinates.

### 2.6.1 Overplotting

The most prominent challenge in parallel coordinates is the clutter produced by a large number of lines, which potentially hide the patterns contained in the data. Lines need more ink than points such that the total mass of data appears larger in parallel coordinates than in scatterplots.

While many authors use the term "clutter" as a synonym for "density" (Ellis and Dix, 2006, 2007), it is important to note that a dense display can reveal important information as well, even without any modification to the traditional parallel-coordinates plot (Inselberg, 2009). Here, we loosely define "clutter" as a parallel-coordinates plot that does not reveal any pattern useful to the observer.

The clutter reduction techniques for parallel coordinates can be categorized into data-driven and screen-based approaches. The former refers to algorithms that operate on the data *before* mapping and rendering in terms of the visualization pipeline and do not affect the visualization. The latter are methods that modify parameters of those two stages. Hence, clustering the data and visualizing only the cluster centroids in traditional parallel coordinates is an example of a data-driven clutter-reduction approach, while zooming into the image is a screen-based approach that might have different effects for different visualizations.

Some approaches to clutter reduction in parallel coordinates are discussed using a slight modification of an established taxonomy (Ellis and Dix, 2007). The methods are grouped in filtering, aggregation, and spatial distortion techniques.

**Filtering** is an operation that removes signals from its input. A filter reduces the number of lines to be rendered. In this sense, dynamic querying (Shneiderman, 1994) is a filter, if implemented with brushing (Section 2.5.1), which reduces clutter by putting the filtered lines *in focus* using some highlighting mechanism. Combining simple brushes using logical operators (Martin and Ward, 1995; Avidan and Avidan, 1999) further allows the user to formulate rather complex queries that might even achieve faster and more accurate results using parallel coordinates than using a structured query language (SQL) (Siirtola and Raiha, 2006). Another type of filter uses sampling at lower rates than for the input data and has been suggested to reduce the actual number of lines to be rendered (Ellis and Dix, 2006) depending on the density (Section 2.3.1). This approach assumes that subsets of the data may represent the dominant features if sampled appropriately. Clearly, it depends on the sampling strategy and the density estimation technique (Ellis and Dix, 2006).

**Aggregation** refers to the computation of the sum or integral of a subset of data and can be performed in the data domain and in the parallel-coordinates domain. There are many different ways to aggregate data and to render the resulting *aggregate items* (Elmqvist and Fekete, 2010). To reduce clutter aggregates are rendered instead of individual samples. Typical aggregate items computed in the data domain are the mean (Siirtola, 2000; Heinrich et al., 2012a; Hasenauer et al., 2012), median (Rosenbaum et al., 2012), or cluster centroid (Fua et al., 1999b) of a subset of samples. The range of visual mappings for aggregate items covers those discussed in Section 2.3. Traditional polylines (Siirtola, 2000) and curves (McDonnell and Mueller, 2008; Heinrich et al., 2012a) can be used either alone (Siirtola, 2000) or as an overlay (Hasenauer et al., 2012) if no information about the distribution of the subset is available. Polygons (Fua et al., 1999b;

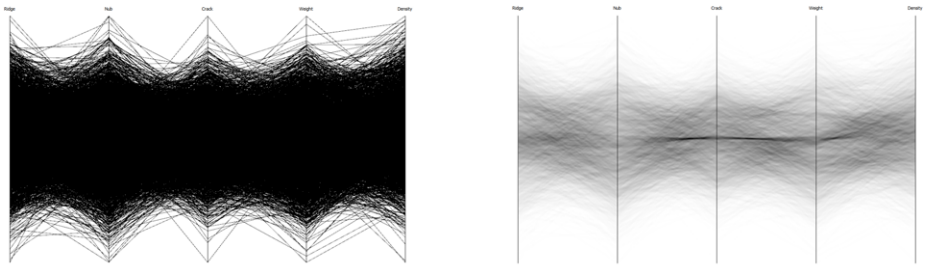


**Figure 2.10:** A frequently used synthetic dataset for parallel-coordinates plots about the geometric features of pollen grains. The large number of lines (3 848 observations, left) may hinder the perception of patterns in the data. Note, however, that the hyperbolic shape of the envelope hints at normally distributed data. Density-based approaches (right) may reveal patterns that are not visible otherwise. The cluster of samples that appears in the center forms the word "EUREKA" if viewed in a scatterplot.

Andrienko and Andrienko, 2004; Rosenbaum et al., 2012), histograms, or boxplots on the axes provide means to visualize the extent and distribution of subsets. Clusters can also be visualized using bundles. Hierarchical data structures (Fua et al., 1999b; Rosenbaum et al., 2012) can further be exploited to render lines or aggregate items at different levels of detail or to progressively refine the final visualization. The computation of a density (Section 2.3.1) is often referred to as a clutter-reduction technique as it is particularly useful to reveal dense areas and clusters in the data (Figure 2.10).

**Spatial distortion** techniques apply a transformation to the viewport. The most common representatives are fisheye views and the traditional, linear zoom. Distortion can help resolve uncertainty about line crossings, clarify dense areas, and brush individual lines with a pointing device. In addition, horizontal distortion (changing the axis-spacing vector) affects angles and slopes of lines, which can have an impact on the accuracy of judging angles (Cleveland and McGill, 1984, 1987; Gehlenborg and Wong, 2012).

With axis scaling (Section 2.5.2), the same effect as for spatial distortion can be achieved by rescaling the data at adjacent axes with the same function. Axis scaling is performed in the data domain and thus allows one to use different scales for each axis. Axis scaling thus belongs to the class of *line-displacement* techniques for clutter reduction.

**Dimensional reordering** in parallel coordinates is the same as axis translation (Section 2.5.2). Reordering the axes in a parallel-coordinates plot may reduce clutter by revealing patterns (e.g. of correlation) that might have been hidden before. An overview of axis-reordering techniques is given in Sections 2.3.2 and 2.5.2.

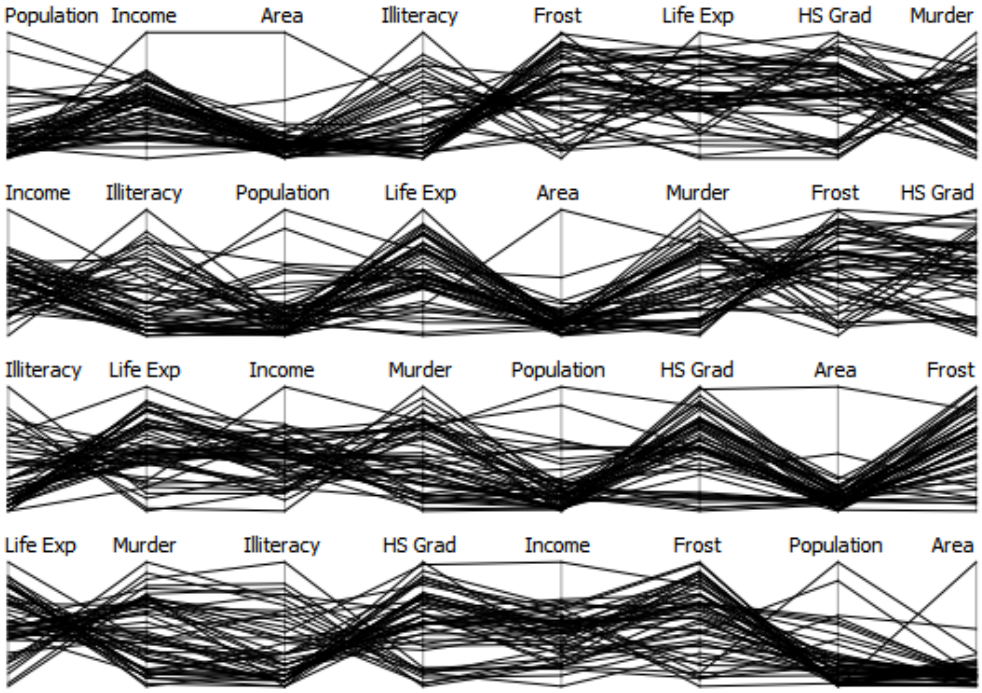



**Figure 2.11:** Different axis orders exhibit different patterns of correlation. The 8-dimensional census dataset (Becker et al., 1988) shows several statistics of the 50 states of the USA and is layed out in the parallel-coordinates matrix (Chapter 6) such that every pair of axes appears exactly once. The topmost plot shows a negative correlation between "Illiteracy" and "Frost", while the bottom plot indicates that "Life Exp" is negatively correlated with "Murder". Taking a close look at the "HS Grad" axes, we find that there is a cluster of states having a low rate of high-school graduates. Also, the bottom row indicates a negative correlation between "Illiteracy" and "HS Grad". A moderately negative correlation seems to be between "Life Exp" and "Illiteracy" in the third row, as well as a positive correlation with "Income".

### 2.6.2 Axis Order

Since parallel coordinates were introduced (Maurice d'Ocagne, 1885), axes are placed in parallel with different preferences for a horizontal (Inselberg, 1985) or vertical (Wegman, 1990) layout. Independent of the orientation, the order of axes affects the patterns revealed by a parallel-coordinates plot (Wegman, 1990) (see Figure 2.11). As there are $N!$ possible orderings for $N$ axes, many researchers addressed the *axis order problem* in their work. While most of the papers deal with using some measure to score an ordering of axes, others build on that and discuss how to visualize multiple orderings in a single display.

Considering two-dimensional relations, where the order of $N$ axes defines

the pairwise plots of the full parallel-coordinates plot independently of the orientation, it is useful to model these relations in a graph-theoretic framework (Wegman, 1990; Qu et al., 2007; Hurley and Oldford, 2010; Zhang et al., 2012) where vertices

$$V = \{x_i \mid i = 1, ..., N\}$$

represent axes and edges

$$E = \big\{\{x_i, x_j\} \mid i, j = 1, ..., N\big\}$$

represent pairwise plots of axes. Now, the *complete graph* $K_N$ models the set of all pairwise relations between $N$ dimensions and the number of edges is

$$|E| = \frac{N(N-1)}{2}.$$

Note that a parallel-coordinates plot can be constructed by following a *path* in $K_N$ and laying out axes in parallel according to the order of nodes in the path. In particular, the traditional parallel-coordinates plot corresponds to a *Hamiltonian path* in $K_N$, i.e. a path that visits every node exactly once. See Hurley and Oldford (2010) for an excellent treatment of graph-theoretic approaches to the pairwise display of variables.

There are different ways to visualize *all* pairwise relations in parallel coordinates using the previously described graph model. In general, it suffices to find an Eulerian trail (Hurley and Oldford, 2010) visiting all edges in $K_N$ and laying out the axes in parallel coordinates accordingly. For $N = 2m + 1, m \in \mathbb{N}$, no such trail exists, and some redundancy has to be tolerated by visiting some edges twice. For some applications, it is necessary to add another constraint to the problem of visualizing all pairwise relations by requiring sub-paths to be Hamiltonian and of length $N$. In other words, all pairwise relations should be visualized in sets of $N$-dimensional parallel-coordinates plots, where every plot contains all $N$ axes of the input dataset. Such a *Hamiltonian decomposition* of the complete graph $K_N$ into $m$ Hamiltonian paths for $N = 2m$ and $m$ Hamiltonian cycles for $N = 2m + 1$ can be used to visualize all pairwise relations in a single parallel-coordinates plot (Hurley and Oldford, 2010) (with some edges visited twice for $N = 2m$) or in a matrix layout (Heinrich et al., 2012b) as in Figure 2.11 (with some vertices visited twice for $N = 2m + 1$). Other matrix-based visualizations of multiple parallel-coordinates plots use Latin-squares (Viau et al., 2010), ranked displays (Tatu et al., 2009; Albuquerque et al., 2009), and manual orderings (Claessen and van Wijk, 2011). The PCM will be presented in Chapter 6.

With increasing $N$, all approaches to enumerate and visualize multiple paths will become impractical at some point, either due to the computational complexity or the limited screen real-estate. Then, a choice has to be made to decide which axis order to prefer. This problem can be translated to the graph model by weighing edges with a distance measure $d\colon (x_i, x_j) \longrightarrow \mathbb{R}$ and order paths by

their total edge weight. The metrics for ordering axes in parallel coordinates can be grouped into data-space measures (Ankerst et al., 1998; Yang et al., 2003; Guo, 2003; Zhao et al., 2003; Peng et al., 2004; Hurley, 2004; Wilkinson et al., 2006; Johansson et al., 2009; Hurley and Oldford, 2010; Ferdosi and Roerdink, 2011; Zhao and Kaufman, 2012) defined in the data domain and image-space measures (Tatu et al., 2009; Albuquerque et al., 2009; Dasgupta and Kosara, 2010; Tatu et al., 2011) defined in the parallel-coordinates domain. Data-space metrics are well-known from statistics and data mining and include the Euclidean distance, Pearson correlation, Kendall's $\tau$, etc. In contrast, image-based metrics measure the slope of lines, their overlap (density), the number of line crossings and -angles, convergence, etc. Screen-based metrics (Novotny and Hauser, 2006; Dasgupta and Kosara, 2010) operate on the rasterized image of a parallel-coordinates plot and further incorporate the current screen resolution when computing a measure. The most common tasks being supported by both types of measures are correlation analysis (Hurley, 2004; Johansson et al., 2009; Hurley and Oldford, 2010; Ferdosi and Roerdink, 2011; Zhao and Kaufman, 2012), clustering of data points (Guo, 2003; Tatu et al., 2009; Albuquerque et al., 2009; Johansson et al., 2009; Tatu et al., 2011; Ferdosi and Roerdink, 2011; Zhao and Kaufman, 2012), clustering of dimensions (Ankerst et al., 1998; Hurley, 2004), clutter reduction (Peng et al., 2004), dimensionality reduction (Yang et al., 2003; Johansson et al., 2009), and outlier detection (Wilkinson et al., 2006; Johansson et al., 2009). Note that all measures can be applied before or after rasterization in the respective domain, which allows one to include the current resolution into the computation of a metric. As even finding the single Hamiltonian path/cycle with the smallest edge weight is NP-hard (Hurley and Oldford, 2010), heuristics (Ankerst et al., 1998; Hurley, 2004; Hurley and Oldford, 2010) or manual path selection (Qu et al., 2007; Zhang et al., 2012) can be used instead.

Other approaches have been proposed to order axes according to higher-order measures (Theisel, 2000; Johansson et al., 2009; Ferdosi and Roerdink, 2011), clustering (Inselberg and Avidan, 1999; Yang et al., 2003), or 3D parallel coordinates (Lind et al., 2009). Without changing the order of axes, a grand tour can be used with parallel coordinates to traverse different projections of the data.

### 2.6.3 Line Tracing

The line-tracing problem in parallel coordinates is a special case of the *linking problem* in statistical graphics (Becker et al., 1987). Assume two data points $A = (a_1, a_2, a_3) \in \mathbb{R}^3$ and $B = (b_1, b_2, b_3) \in \mathbb{R}^3$ and two 2D scatterplots that relate dimensions $x_1$ with $x_2$ and $x_2$ with $x_3$. Then, linking $A$ with $B$ refers to the task of relating the lower-dimensional projections of $A$ and $B$ with each other by some visual means. For a single polygonal line, parallel coordinates inherently solve the linking problem. However, if $A$ and $B$ coincide on one dimension, e.g. $a_2 = b_2$, it is impossible to visually link the points. This is demonstrated in

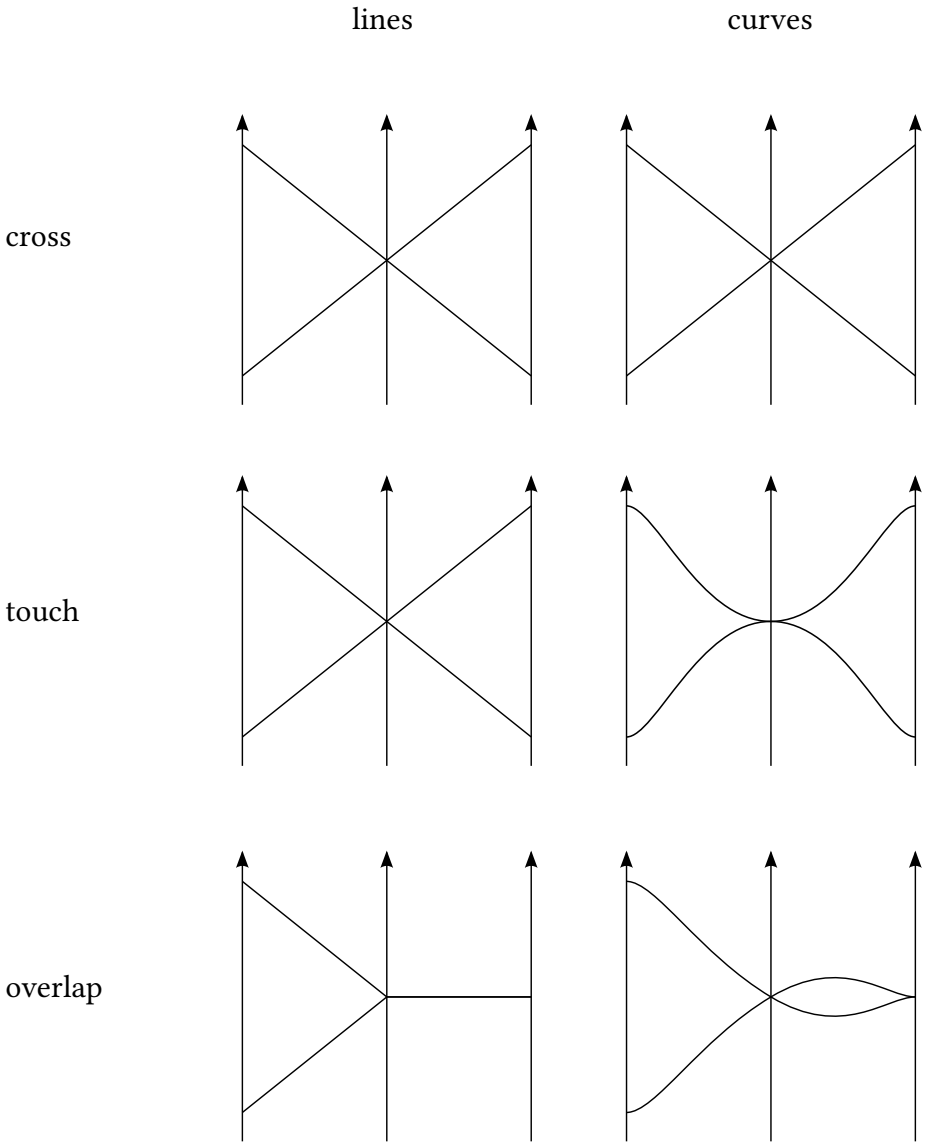



**Figure 2.12:** Ambiguities for tracing lines can partly be solved using curves instead of lines. For a pair of lines crossing at the midway axis (top), the curve model exhibits the same pattern. If lines touch instead (middle), a different pattern emerges. Note that without knowing the underlying model, it is still not possible to visually trace the lines. For lines that coincide (bottom), a smooth representation succeeds in disambiguating the samples.

Figure 2.12, where it is not possible to assign all line segments unambiguously to a data point. There are basically two approaches to mitigate the linking problem for parallel coordinates. Using different colors to distinguish different points is a popular solution. However, this approach does not scale well with the number of points as it is difficult for the human visual system to reliably distinguish

more than twelve colors (Ware, 2004). The other technique is to use curves instead of lines (see Section 2.3.1 for a review of the different implementations using curves). In contrast to lines, curves provide at least $C^1$ continuity and thus support the Gestalt principle of continuity. The disadvantage of using curves is the distortion of values between axes, such that some of the geometric properties as presented in Section 2.2 are not valid. However, other statistical properties of curve-based parallel coordinates were shown to be useful for pattern recognition (Moustafa, 2011).

### 2.6.4 Sets and Clusters

The previous section presented clustering as a clutter-reduction technique. The focus of this section is the visualization of pre-clustered data with parallel coordinates. Here, the motivation for clustering is not to reduce clutter but to visualize patterns or anomalies within or between *sets of data*. For metric data, some of the techniques presented in the previous section about aggregation are applicable, i.e. the representation of a cluster by its mean value (or *centroid*). However, sets are not necessarily metric data and are often used to categorize a data*set*. A simple but effective method to distinguish a small set of categories is by using color. If the color channel cannot be used, bundling has been shown to work well for the identification of clusters while having a low impact on the effectiveness of the estimation of correlations (Heinrich et al., 2012a). Other approaches based on geometry are to map clusters to envelopes (Moustafa, 2011) or bounding-boxes (Fua et al., 1999b).

### 2.6.5 Time Series

Time series are frequently visualized using line plots, where a single line or curve represents the progression or change of a data point over time. These plots can be constructed with the linear interpolation model of Section 2.2.3, simply by labeling the dimensions of the data domain as the time points of a time series. Using this model, time-series plots are a special case of parallel-coordinates plots, with the restriction to a common scale on every axis and a fixed ordering of dimensions. This has implications in both directions—from time-series plot to parallel-coordinates plot and vice versa. On the one hand, some of the results that were presented for parallel coordinates might also be valid for the interpretation of time-series plots. On the other hand, one of the reasons of the popularity of parallel coordinates might be the familiar visual pattern of a line interpolating a set of points that is long known from time-series plots such as stock market diagrams or the temperature forecast. While both types of visualization are expressed using similar visual mappings, the underlying model is different, as time points are samples of a one-dimensional continuous domain, whereas the axes in parallel coordinates represent one dimension each.

Several authors combined the visualization of time series and parallel coordinates. A simple but effective technique is to append data dimensions as axes to a time series plot (Dietzsch et al., 2009), which enables the brushing of data samples with respect to the additional variables. Interchanging axes with "profiles" in a time-series plot allows for a truly multivariate interpretation of time-series data: here, an axis represents a measurement, dimension, or variable, while a data point is mapped to a polyline. Inselberg (2001) maps time to an axis in parallel coordinates and visualizes aircraft trajectories with indexed points in parallel coordinates. Although samples from time series are intrinsically ordered, the order in which data samples are rendered in parallel coordinates has no effect on the final visualization unless alpha-blending is applied. *Temporal parallel coordinates* (Johansson et al., 2007) respect the order of time points and render a constant-density polygon for two consecutive samples. This corresponds to a nearest-neighbor interpolation of values in the data domain. The density approach is scalable and allows one to put more emphasis on large gradients, i.e. for which data dimensions the total amount of change is highest.

Another technique for the visualization of time series in parallel coordinates employs animation (Barlow and Stuart, 2004; Theron, 2006). Here, a single parallel-coordinates plot is computed for every time step. A series of frames can then either be animated automatically or explored by the user in a stepwise fashion.

### 2.6.6 Uncertainty

Uncertainty is a term that is difficult to define, and it is not the purpose of this section to do so. For the upcoming discussion, uncertainty may refer to variance, error, precision, or noise. We will shortly review how uncertainty may be introduced by visualizing data with parallel coordinates, how it can be addressed, and how a given, quantitative measure of uncertainty may be visualized along with the primary data in parallel coordinates.

According to the taxonomy of Dasgupta et al. (2012), uncertainty occurs in a parallel-coordinates plot in the process of encoding the image of the parallel-coordinates plot as well as in the decoding steps involved when processed by humans or machines. For humans, additional uncertainty may arise individually due to different aspects of cognition. For example, an experienced user might be less uncertain about the recognition of patterns in parallel coordinates than a novice. Encoding and decoding directly relate to what we termed *image generation* and *image analysis*, for which we briefly discuss uncertainty.

The encoding of a parallel-coordinates plot further introduces uncertainty in different stages (Dasgupta et al., 2012) of the transformation from the data domain to the parallel-coordinates domain. Despite the loss of information due to the projection of a high-dimensional dataset to a set of 2D spaces, *visual uncertainty* may have a variety of sources in parallel coordinates. The user-driven

filtering of dimensions and the application of algorithms in the data-mapping stage introduces uncertainty regarding the *completeness* (sample size) and the *configuration* (axis ordering) of the plot. Note that, in the KDD pipeline (Fayyad et al., 1996b), data mapping refers to the transformation step. The rendering of a parallel-coordinates plot causes further uncertainty as it involves sampling lines or line densities on a regular grid (the pixels). This step depends on the resolution of the screen (the sampling frequency), the sampling kernel (usually a rectangular function), and the reconstruction kernel (rectangular for opaque lines). These parameters influence the *precision* in the visual mapping of data samples to lines. Visualizing aggregated information such as clusters instead of individual samples decreases the *granularity* and with that increases the uncertainty of the visual representation. Granularity is a common parameter subject to interaction and is often controlled by detail-on-demand operations (see also Section 2.5).

The analysis of an image of parallel coordinates consists of decoding the information contained in the sampled representation of the plot. In order to discuss the theoretical aspects of uncertainty associated with perception and cognitive processing of a parallel-coordinates plot, however, a perfect reconstruction of lines has to be assumed. Then, the human visual system introduces uncertainty when sampling the image, for the same reasons as above. The traceability of lines at the intersection with axes is yet another source of uncertainty that occurs for many visualizations where overlap is possible (Elmqvist and Fekete, 2010). This type of uncertainty can be reduced with brushing (Section 2.5.1) and curves (Sections 2.2.3 and 2.3.1). A related problem is the identification of lines in heavily cluttered displays with the special case of overlapping lines. While the former can be resolved geometrically with increasing resolution or by scaling, the latter can be detected using density or transparency (Section 2.3.1).

The representation of uncertainty in parallel coordinates has been addressed by few researchers. One approach is to model data points with a normal distribution and to map these to parallel coordinates with respect to the point–line duality (see Section 2.3.1). The resulting image resembles a probability distribution of lines in parallel coordinates, such that uncertain points are de-emphasized while certain values appear more salient.

## 2.7 Applications

This section provides references to some of the many applications of parallel coordinates in the life sciences and engineering domains. Due to the large amount of publications using parallel coordinates, we can only provide a short and non-exhaustive list of selected applications per domain.

### 2.7.1 Life Sciences

Parallel coordinates were used in a number of applications in different fields of the life sciences, including biology (Keefe et al., 2009; Geng et al., 2011), bioinformatics (Dietzsch et al., 2009), systems biology (Barsky et al., 2008; Hasenauer et al., 2012), genetics (Rübel et al., 2006), functional genomics (Meyer et al., 2010b), neurophysiology (ten Caat et al., 2005; ten Caat and Maurits, 2007), or computational chemistry (Becker, 1997). For gene expression data, parallel coordinates can be used to visualize the "profile" of genes (lines) over a set of conditions (axes). This is the natural counterpart to the heatmap (Eisen et al., 1998), where rows represent genes and columns represent conditions. Note that in many occasions, conditions represent time steps, which makes the corresponding parallel-coordinates plot a time-series visualization, where the order of axes is fixed. However, appending statistical dimensions to "profile plots" is a useful technique for model verification and querying statistical properties (Dietzsch et al., 2009). Gene profiles can also be visualized in parallel coordinates to help experts establish functional relationships of the expression of genes to their spatial location (Rübel et al., 2006; Meyer et al., 2010a). In conjunction with metabolic networks, parallel coordinates were used for the visualization of network parameters for single cells (Barsky et al., 2008) and cell populations (Hasenauer et al., 2012).

### 2.7.2 Engineering

Data in engineering applications often consists of multi-attribute samples given in the spatio-temporal domain, that need to be analyzed with respect to the time and place they were taken. Hence, linking parallel-coordinates plots (representing the multi-attribute data) to maps and 3D-visualizations (representing the spatial domain) was shown to be useful, e.g. to compare census data of different countries (Andrienko and Andrienko, 2001), visualize health statistics (Edsall, 2003), analyze traffic (Guo et al., 2011a), or explore CFD data such as weather simulations (Doleisch et al., 2004; Blaas et al., 2008) or nasal air flow (Zachow et al., 2009). Continuous parallel coordinates (Heinrich and Weiskopf, 2009) were used to detect features in CFD data (Lehmann and Theisel, 2011). Parallel coordinates can also be used to navigate high-dimensional parameter spaces in volume-rendering applications (Pradhan et al., 2005; Tory et al., 2005; Crider et al., 2007) and to design multidimensional transfer functions (Guo et al., 2011b). The exploration of high-dimensional parameter spaces is another frequent application of parallel coordinates, e.g. for aircraft- and car design (Goel et al., 1999; Berger et al., 2011) or diesel injection systems (Matkovic et al., 2005; Muigg et al., 2011). Parallel coordinates were further applied for process control in chemical plants (Albazzaz and Wang, 2006; Comfort et al., 2011; Dunia et al., 2012), for alarm filtering for industrial systems (Azhar and Rissanen, 2011), and for air traffic control (Inselberg, 2001). Another security-related application of parallel

coordinates is the detection of network attacks (Choi et al., 2009; Kim et al., 2009; Tricaud et al., 2011; Promrit et al., 2011).

CHAPTER 3

# Continuous Parallel Coordinates

## 3.1 Introduction

Although statistical data plots including parallel coordinates and scatterplots were originally introduced using discrete (point-based) rendering, they have been proven to be useful tools for the analysis of multidimensional multivariate data (Doleisch et al., 2003; Blaas et al., 2008). Here, the data is typically given on a grid within a spatial embedding, and the data values at the grid points are interpreted as discrete samples for the construction of the statistical plots, which can then be applied to analyze trends, correlations, clusters, or distributions, etc. While the data domain may be continuous, the plotting process typically relies on discrete samples that are then rendered using either points or density estimates (such as kernel density estimation). Hence, a discrete model is involved at one stage of the rendering process.

While the representation of discrete data points as lines in parallel coordinates may reveal trends and patterns latently contained in the data, it also tends to clutter the view due to potentially heavy overplotting. In consequence, traditional parallel coordinates do not scale well with sample size, making it difficult to use with large datasets. This challenge is commonly referred to as the overplotting problem and was discussed in more detail in the previous chapter. However, the clutter-reduction techniques reviewed in Section 2.6.1 ignore the continuous nature of the data in multidimensional multivariate applications.

Bachthaler and Weiskopf (2008) introduced the concept of *continuous scatterplots* for data defined on a continuous domain. Based on their generic density model,

this chapter introduces *continuous parallel coordinates* as an extension of the mathematical model of line density in parallel coordinates (Miller and Wegman, 1991) for generic continuous density distributions given in the data domain. The construction of continuous parallel coordinates requires substantial modifications and extensions compared to scatterplots and histograms because the duality of points and lines needs to be considered. As proposed by Miller and Wegman (1991), the definition of point density is based on "counting" discrete lines: the point density is derived by examining the limit process of lines intersecting an interval with indefinitely small vertical extent. Using this model, a relation of point densities from 2D continuous scatterplots (Bachthaler and Weiskopf, 2008) and multivariate time series to continuous parallel coordinates is derived.[1]

Furthermore, different numerical and analytical solutions for the computation of the model are examined. Based on the point–line duality of scatterplots and parallel coordinates, the algorithms can be divided in two classes. In the *scattering* approach, a density description in parallel coordinates is obtained implicitly by sampling points from the input field. In contrast, the *gathering* approach computes the density by integration within the scatterplot.

Continuous parallel coordinates exhibit several benefits: (i) The visualization does not depend on the resolution of the data, as the available interpolation schemes are used to compute the continuous representation in parallel coordinates. (ii) In contrast to other frequency plot construction algorithms, our method is parameter-free: it does not rely on bucket size, binning, or texture resolution, which are commonly used for the approximation of density. (iii) A continuous density model scales well with sample size and resolution, providing the basis for a visualization for which overplotting cannot occur. This makes continuous parallel coordinates interesting for the analysis of large data.

## 3.2 Related Work

This section covers work related to the application of statistical plots for multidimensional multivariate data, overplotting, and density-based clutter-reduction techniques for parallel coordinates. A more general survey of density-based representations and clutter reduction techniques is given in Sections 2.3.1 and 2.6.1.

The application of statistical graphics to multidimensional multivariate data has gained importance in recent years, both in academic and industrial applications. The SimVis system (Doleisch et al., 2003) uses histograms and scatterplots in a coordinate-view setup with linking and brushing to explore simulation data.

[1] **Parts of this chapter have been published in:** J. Heinrich and D. Weiskopf. Continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1531–1538, 2009.

Similarly, the visualization toolkit (VTK) (Schroeder et al., 2006) contains a multitude of visualization techniques for multidimensional multivariate data. In other examples, 2D transfer functions are specified with data frequencies visualized in scatterplots (Kniss et al., 2002), and parallel coordinates are used for multidimensional transfer function design (Pradhan et al., 2005; Tory et al., 2005; Zhao and Kaufman, 2010; Guo et al., 2011b) and the analysis of simulation data (Blaas et al., 2008). However, none of these examples considers the continuous domain of the input data field; instead they rely on discrete samples.

To resolve the inconsistency between continuous data model and discrete plotting, Bachthaler and Weiskopf (2008) introduced a mathematical model that describes the mapping of densities from a continuous input field with known interpolation scheme to the continuous data domain. The model of continuous scatterplots is generic and versatile, with various applications and further analysis methods that build upon the construction of those scatterplots. For example, critical line structures can be extracted from continuous scatterplots to obtain further insight in the structure of the data (Lehmann and Theisel, 2010). As another example, 2D joint histograms could, in the form of continuous scatterplots, support information-theoretic flow visualization (Xu et al., 2010). Continuous scatterplots may also be interpreted as joint probability density functions that facilitate the identification of mutual information in image registration problems or between multiple 3D scalar fields in multi-field data analysis (Nagaraj and Natarajan, 2011). Continuous scatterplots also play an important role wherever kernel estimates have to be computed for feature analysis. For example, they could serve as an alternative for the continuous density distribution functions that are used for volumetric transfer function generation (Maciejewski et al., 2009). Finally, the 1D version of continuous scatterplots—continuous histograms—can be used to examine isosurface statistics for analyzing 3D scalar fields (Carr et al., 2006; Scheidegger et al., 2008).

## 3.3 Mathematical Model

The model of continuous parallel coordinates is based on the scalar density fields defined on the $N$-dimensional data domain. A nice feature of parallel coordinates is that the construction of the overall plot can be split into the construction of several independent parallel-coordinate systems for 2D data, each emerging from a 2D scatterplot (cf. Section 2.2.1). The final plot is then formed by placing the parallel axes consecutively on the plane. For $N$-dimensional data, this results in the computation of $N - 1$ independent parallel-coordinate systems.

Therefore, we will focus on 2D data for the derivation of the mathematical model for continuous parallel coordinates and assume $\boldsymbol{d}_2 = (0, 1)^{\mathrm{T}}$ if not stated otherwise.

As described in Section 2.2, a point $A = (a_1, a_2)$ in the data domain is mapped to the line $\overline{A}$ between adjacent axes $\overline{X}_1$ and $\overline{X}_2$ in the parallel-coordinates domain. Assuming $\boldsymbol{d_N} = (0, 1)^{\mathrm{T}}$, Equation (2.2) can be written as:

$$\overline{A}\colon y = (a_2 - a_1)x + a_1;\ x \in [0, 1]. \tag{3.1}$$

In the data domain, Equation (3.1) implicitly represents the line $\ell$ corresponding to the point $\overline{\ell} = (x, y)$ of the parallel-coordinate system. For this purpose, it may be interpreted as the projection of the vector $\boldsymbol{A} = (a_1, a_2)^{\mathrm{T}}$ onto $\tilde{\boldsymbol{n}}$, which can be expressed by the dot product:

$$\ell\colon y = \tilde{\boldsymbol{n}} \cdot \boldsymbol{A}. \tag{3.2}$$

Note that $\tilde{\boldsymbol{n}} = (1 - x, x)^{\mathrm{T}}$ is perpendicular to $\ell$ and only depends on $x$.

The distance $D$ of $\ell$ to the origin is inherently contained in (3.2), but its computation assumes normalization of $\tilde{\boldsymbol{n}}$ to unit length, such that:

$$D(\ell) = \frac{y}{\|\tilde{\boldsymbol{n}}\|} = \frac{\tilde{\boldsymbol{n}}}{\|\tilde{\boldsymbol{n}}\|} \cdot \boldsymbol{A}. \tag{3.3}$$

Hence, the main conclusions of this section are two-fold: (i) the distance $D(\ell)$ linearly correlates with $y$, the vertical position of the corresponding point in the parallel-coordinates domain and (ii) the slope $m$ in the data domain only depends on $x$, the horizontal position of the corresponding point $\overline{\ell}$ in the parallel-coordinates domain.

### 3.3.1 Generic Density Model

The proposed density model is based on mass conservation, assuming that (i) points in the data domain are given according to some density description, and (ii) the mapping of points from the data domain to lines in the parallel-coordinates domain does not change the number of points (lines), i.e. a point in the data domain corresponds to exactly one line in the parallel-coordinates domain and vice versa. As a consequence, a vertical line (or an interval) in the parallel-coordinates domain is mapped to a set of indefinitely dense parallel lines[2] (or an area) in the data domain (see Figure 3.1).

This can be used to derive a density description for points in parallel coordinates by examining the limit process at the transition of areas to lines in the data domain. With the assumptions (i) and (ii) stated above, the mass $M$ covering an area $\Phi \subset \mathbb{R}^2$ in the data domain with density $\sigma\colon \mathbb{R}^2 \longrightarrow \mathbb{R},\ \boldsymbol{x} \mapsto \sigma(\boldsymbol{x})$ is $M = \int_\Phi \sigma(\boldsymbol{x}) \mathrm{d}^2 x$. Considering the duality of points and lines, the density $\varphi\colon \mathbb{R}^2 \longrightarrow \mathbb{R},\ (x, y) \mapsto \varphi(x, y)$ of a point $\overline{\ell} = (x, y)$ in parallel coordinates is based on "counting" lines within an interval along the vertical axis (cf. Section 2.3.1). It

[2] The ideal point assuming projective planes (see Figure 2.4, page 19)

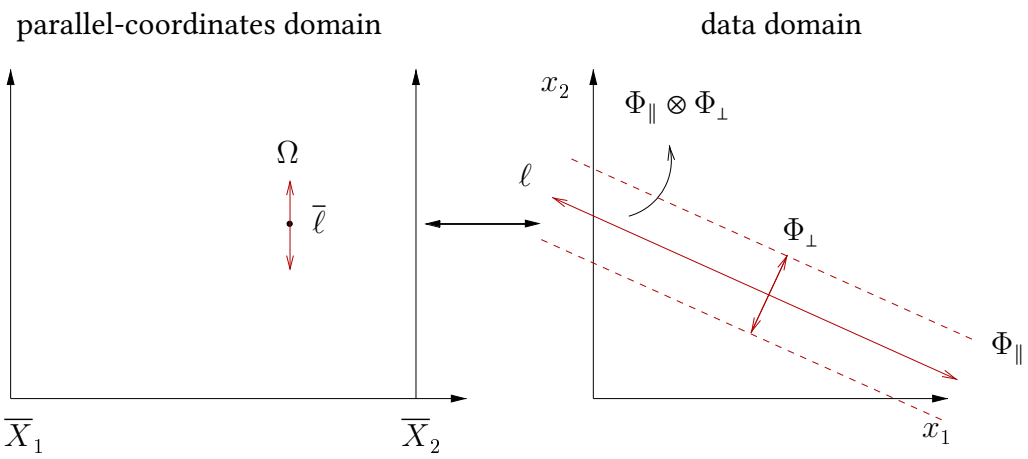



**Figure 3.1:** The interval $\Omega$ containing $\bar{\ell}$ in the parallel-coordinates domain is mapped to the area stripe $\Phi$ containing $\ell$ in the data domain. Since the slope of $\ell$ is independent from $x$, the stripe has parallel border lines.

can then be integrated to compute the mass of the covered interval $\Omega$ according to $\int_\Omega \varphi(x,y)\mathrm{d}y$. Assuming mass conservation, the mass of points (lines) does not change under the transformation from data domain to parallel-coordinates domain:

$$M = \int_\Omega \varphi(x,y)\mathrm{d}y = \int_\Phi \sigma(\boldsymbol{x})\mathrm{d}^2x. \tag{3.4}$$

In contrast to the discussion in Section 2.3.1, we now assume the density $\sigma(\boldsymbol{x})$ to be known for any $\boldsymbol{x}$ (see (Bachthaler and Weiskopf, 2008) for a derivation of densities in the data domain). Applying the fundamental theorem of calculus to (3.4) can be used to express the density in the parallel-coordinates domain in terms of $\sigma$:

$$\varphi(x,y) = \frac{\mathrm{d}M}{\mathrm{d}y} = \frac{\mathrm{d}}{\mathrm{d}y}\int_\Phi \sigma(\boldsymbol{x})\mathrm{d}^2x. \tag{3.5}$$

To compute the integral, the integration domain $\Phi$ is split in two parts: one integration along the line $\Phi_\parallel = \ell$ corresponding to $(x,y)$ and another integration along the perpendicular direction $\Phi_\perp$ (see Figure 3.1). For this purpose, the $x_1x_2$-coordinate system is rotated such that the rotated $x_2$-axis can be identified with the normal of the line $\ell$.

According to the detailed derivation in Appendix A, the line density in a point $\bar{\ell} = (x,y)$ of the parallel-coordinate system is obtained by integrating over the dual line in the data domain:

$$\varphi(x,y) = \int_\ell \frac{\sigma(\boldsymbol{\ell}(\lambda))}{\|\tilde{\boldsymbol{n}}\|}\mathrm{d}\lambda \tag{3.6}$$

with $\boldsymbol{\ell}(\lambda)$ being the arc-length parametrized line $\ell$. Note that $\sigma$ typically has finite support, although $\ell$ is defined on an indefinite domain.

### 3.3.2 Numerical Integration

Equation (3.6) describes the line density at any point in the parallel-coordinates domain as a line integral along its dual line in the data domain, where the function to be integrated is the respective point density $\sigma$ of the scalar input field. In this section, two substantially different approaches to the numerical integration of (3.6) are briefly discussed.

A typical *gathering* technique is to sample $\varphi$ in the parallel-coordinates domain followed by an evaluation of (3.6) in the data domain. Here, each sample $\overline{\ell}$ has a dual line $\ell$ constituting the integration domain for the computation of $\varphi(\overline{\ell})$. Numerical integration now implies further sampling of $\sigma$ over $\ell$ and can be implemented using known techniques such as Monte Carlo integration or Riemann sums.

By exploiting the point–line duality, another approach to numerical integration of (3.6) is possible. Here, points are sampled from the data domain and the respective densities are *scattered* to line densities in the parallel-coordinates domain. The generic scattering algorithm using additive blending is

1: sample points $P_i,\ i = 1, 2, \ldots, n$
2: **for all** $P_i$ **do**
3: setRGBAdrawColor$(1, 1, 1, \alpha)$
4: drawLine$(\overline{P}_i)$
5: **end for**

A possible application of the scattering algorithm is to sample points on a regular grid on the data domain (step 1) and set $\alpha \leftarrow \sigma(P_i)$, effectively resulting in a uniform sampling of the density function $\sigma$. Note that point densities $\varphi$ are then constructed implicitly by the superposition of lines with different density. Due to the linear model of (3.6), this leads to the same result as the gathering approach. Instead of sampling uniformly on a regular grid in the data domain, a random sampling strategy (with a uniform probability distribution) could be used to achieve an "implicit" Monte Carlo integration for the computation of density in parallel coordinates. Similarly, low-discrepancy sequences (Niederreiter, 1992) could be used for sampling to obtain quasi Monte Carlo integration. Using $\sigma$ in an importance sampling approach further improves performance compared with the standard or quasi Monte Carlo methods. In this case, a constant density $\alpha$ must be used for $\overline{P}_i$, i.e. $\alpha \leftarrow const.$ in step 3 of the generic scattering algorithm. Sample points are drawn from a probability density function given by $\sigma$, up to a constant scaling factor. Now, the computation of $\varphi(\overline{\ell})$ at the sampling points $\overline{\ell}$ remains only a matter of counting the (weighted) lines intersecting with $\overline{\ell}$, which also is the basis of our mathematical model of continuous parallel coordinates. Note that $\varphi$ depends on the number of samples and thus has to be normalized in order to properly compare the results.

In practice, many 2D density fields are derived from higher dimensional input

fields defined in the spatio-temporal domain with known (sampling) densities, such as 3D scalar fields, 3D vector fields, or multi-attribute fields. Bachthaler and Weiskopf (2008) describe the transformation of density from the spatio-temporal domain to the data domain under the assumption of mass conservation.

In consequence, the computation of continuous parallel coordinates using scattering may also be conducted on the spatio-temporal domain. Here, multidimensional points are sampled and mapped to polylines in parallel coordinates with $\alpha \leftarrow const.$ This approach affects step 1 of the generic scattering algorithm, as points are now sampled according to the given density in the spatio-temporal domain (typically, constant density). This method and previous density-based methods (cf. Section 2.3.1) converge to the same basic computation with increasing grid resolution of the input field. Therefore, in the limit of infinitely high resolution of input data, continuous parallel coordinates and previous density-based representations yield the same result.

### 3.3.3 Triangulated Data

In this section, an analytic solution to (3.6) for data given on tetrahedral grids in the spatio-temporal domain is provided. Tetrahedral grids play an important role as simulation grids or as common ground for data exchange using the approximation of other grid structures by triangulation. Continuous scatterplots also support tetrahedral grids by exploiting the projected tetrahedra algorithm (Shirley and Tuchman, 1990). Under the assumption of mass conservation, spatial tetrahedra are projected to a set of triangles in the data domain, resulting in a triangulation of the density distribution with piecewise linear interpolation. Therefore, a piecewise computation of $\varphi(\bar{\ell})$ can be achieved by linear superposition of the contribution of all triangles intersecting the dual line $\ell$. This approach is similar to the previously described scattering of densities, although in this case, triangles instead of points are mapped to parallel coordinates.

Figure 3.2 shows a possible footprint of a triangle $\Delta_{ABC}$ from the data domain to the parallel-coordinates domain.

The points $A$, $B$, and $C$ are mapped to lines $\overline{A}$, $\overline{B}$, and $\overline{C}$ in the parallel-coordinates domain, as described in Equation (3.1). For any vertical line $x = const.$, the intersections with $\overline{A}$, $\overline{B}$, and $\overline{C}$ are $\bar{a}$, $\bar{b}$, and $\bar{c}$, respectively. Without loss of generality, let

$$\bar{a}_y \leq \bar{c}_y \leq \bar{b}_y. \tag{3.7}$$

This means that for each triangle, we label its vertices such that (3.7) is true. Then, $\Delta_{ABC}$ is divided in two subtriangles $\Delta_{AEC}$ and $\Delta_{EBC}$. Here, a case differentiation is necessary depending on the choice of $\overline{w}_y$. First, let $\bar{a}_y \leq \overline{w}_y \leq \bar{c}_y$,

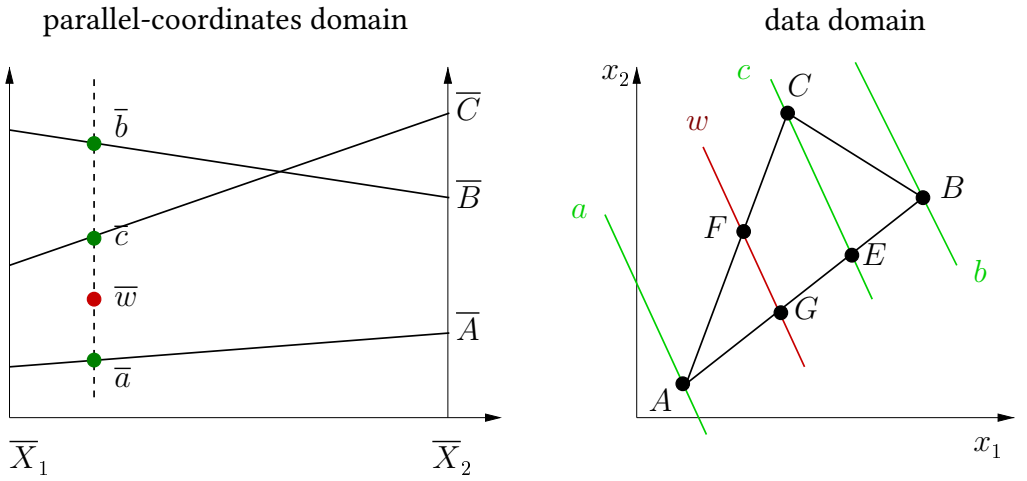



**Figure 3.2:** Footprint of a triangle in the data domain (right) and after transformation to parallel coordinates (left). The point $\overline{w}$ and its dual line $w$ are highlighted in red. Assuming $\overline{w}_y < \overline{c}_y < \overline{b}_y$, the triangle $\Delta_{ABC}$ is divided in two subtriangles $\Delta_{AEC}$ and $\Delta_{EBC}$ by the lines $a$, $b$, and $c$ in the data domain.

as highlighted red in Figure 3.2. The corresponding line $w$ in the data domain intersects $\Delta_{AEC}$ at $F$ and $G$. Using vector notation, $w$ can also be written as:

$$\boldsymbol{w}(\lambda) = \boldsymbol{f} + \frac{\lambda}{t}(\boldsymbol{g} - \boldsymbol{f}) \tag{3.8}$$

with $t = \|\boldsymbol{g} - \boldsymbol{f}\|$ and $\lambda \in [0, t]$ for the segment contained in $\Delta_{ABC}$. Due to the piecewise linear density distribution obtained from the projected tetrahedra algorithm the density in the data domain can also be determined from the respective point densities:

$$\sigma(\boldsymbol{w}(\lambda)) = \sigma(\boldsymbol{f}) + \frac{\lambda}{t}\left(\sigma(\boldsymbol{g}) - \sigma(\boldsymbol{f})\right) \tag{3.9}$$

such that the contribution $\varphi_f^g$ to $\varphi(\overline{w})$ as computed according to (3.6) is:

$$\varphi_f^g = \int_0^t \frac{\sigma(\boldsymbol{w}(\lambda))}{\|\tilde{\boldsymbol{n}}\|} \mathrm{d}\lambda = \frac{t}{2\|\tilde{\boldsymbol{n}}\|}\left(\sigma(\boldsymbol{f}) + \sigma(\boldsymbol{g})\right). \tag{3.10}$$

With barycentric interpolation in subtriangle $\Delta_{AEC}$ the density at the intersection points is obtained as:

$$\sigma(\boldsymbol{g}) = \sigma(\boldsymbol{a}) + u(\sigma(\boldsymbol{e}) - \sigma(\boldsymbol{a})) \tag{3.11}$$

and

$$\sigma(\boldsymbol{f}) = \sigma(\boldsymbol{a}) + u(\sigma(\boldsymbol{c}) - \sigma(\boldsymbol{a})). \tag{3.12}$$

For the computation of $u$, distances of lines as derived in (3.3) can be used. With $\Delta y_w = \overline{w}_y - \overline{a}_y$ and $\Delta D_w = D(w) - D(a)$, $u$ can be derived using the intercept theorem in the data domain:

$$u = \frac{\Delta D_w}{\Delta D_c} = \frac{\Delta y_w}{\Delta y_c}. \tag{3.13}$$

Similarly, for the computation of $\sigma(\boldsymbol{e})$, barycentric interpolation within $\Delta_{ABC}$ yields:

$$\sigma(\boldsymbol{e}) = \sigma(\boldsymbol{a}) + v(\sigma(\boldsymbol{b}) - \sigma(\boldsymbol{a})) \tag{3.14}$$

where

$$v = \frac{\Delta y_c}{\Delta y_b} \tag{3.15}$$

and $\Delta y_b > 0$ (the special case $\Delta y_b = 0$ will be treated later). Now, the final parameter to determine in order to solve Equation (3.10) is $t = \|\boldsymbol{g} - \boldsymbol{f}\|$, which can be obtained using the intercept theorem:

$$\frac{\|\boldsymbol{g} - \boldsymbol{f}\|}{\|\boldsymbol{e} - \boldsymbol{c}\|} = \frac{\|\boldsymbol{g} - \boldsymbol{a}\|}{\|\boldsymbol{e} - \boldsymbol{a}\|} \tag{3.16}$$

and thus:

$$t = u\|\boldsymbol{e} - \boldsymbol{c}\| \tag{3.17}$$

where $\boldsymbol{e}$ is linearly interpolated similarly to (3.14).

Altogether, Equation (3.10) resolves to a single expression depending only on the point coordinates $\overline{w}$ in parallel coordinates and the densities at the respective triangle vertices:

$$\varphi_{\boldsymbol{f}}^{\boldsymbol{g}} = \frac{t}{2\|\tilde{\boldsymbol{n}}\|}\left((2 - u - uv)\sigma(A) + uv\sigma(B) + u\sigma(C)\right). \tag{3.18}$$

The second case $\overline{c}_y \leq \overline{w}_y \leq \overline{b}_y$ is derived analogously by swapping indices $a$ and $b$ in Equations (3.12), (3.11), and (3.13).

Note that both subtriangles $\Delta_{EBC}$ and $\Delta_{AEC}$ may degenerate to a line if either $\overline{c}_y = \overline{b}_y$ or $\overline{a}_y = \overline{c}_y$. As these cases are covered by (3.13) and (3.15), there only remains the special case $\overline{a}_y = \overline{c}_y = \overline{b}_y$, where $v$ is no longer defined. Here, $\Delta_{ABC}$ degenerates to a line with three density values at the corresponding vertices, such that linear interpolation is not valid anymore. In this case, the density at $\overline{w}$ according to the triangle model can no longer be represented by a function in the parallel-coordinates domain. Instead, the degenerate triangle from the data domain maps to a single point in parallel coordinates. The associated density is represented by a delta distribution: $\varphi(\overline{\ell}) = M\delta(\overline{\ell} - \overline{w})$, where $M$ is the mass of the degenerate triangle, which is conveniently determined by integration in the spatio-temporal domain.

### 3.3.4 Time Series

This section provides an analytic solution to (3.6) for time-dependent data in the spatio-temporal domain. Examples are multivariate trajectories, i.e. one-dimensional multivariate data in the notation of Wong and Bergeron (1994).

Let $\boldsymbol{\tau}(t) = (\tau_1(t), \tau_2(t), \ldots, \tau_N(t))^{\mathrm{T}}$ denote an $N$-dimensional trajectory. The traditional (discrete) representation of $\boldsymbol{\tau}(t)$ in parallel coordinates is shown in Figure 3.3 (top) with discrete time steps $t_j$.

As illustrated in Figure 3.4, employing the scattering algorithm with constant density to the samples given by the trajectory may lead to ambiguous visual encodings. Assuming piecewise linear interpolation between samples in time, the mass-conservation model disambiguates those cases.

Given the density function $s(t)$ of the trajectory in the spatio-temporal domain, mass conservation can be stated as

$$\int_{y_j}^{y_{j+1}} \varphi(x,y)\mathrm{d}|y| = M = \int_{t_j}^{t_{j+1}} s(t)\mathrm{d}t \tag{3.19}$$

with $y_j$ being the $y$-coordinate of the polyline generated by $\boldsymbol{\tau}(t_j)$ at horizontal location $x$. Taking the absolute value of $\mathrm{d}y$ in Equation (3.19) ensures a positive mass for polylines with negative slope.

As was mentioned before, an N-dimensional parallel-coordinates plot can be constructed by computing the $N-1$ segments independently. We therefore consider $N = 2$ and $\boldsymbol{d}_2 = (0,1)^{\mathrm{T}}$ for the upcoming derivation. Using the interpolation model from Section 2.2.3, we compute the $y$-coordinate of the line representing $\boldsymbol{\tau}(t) = (\tau_1(t), \tau_2(t))^{\mathrm{T}}$ in parallel coordinates as

$$y(t,x) = (1-x)\tau_1(t) + x\tau_2(t), \;\; x \in [0,1]. \tag{3.20}$$

To obtain the contribution of a single sample $\varphi_j(x,y)$ from $y(t,x)$, the derivative of $y$ with respect to $t$ is required:

$$\frac{\partial}{\partial t} y(t,x)|_{[t_j,t_{j+1}]} = \frac{\Delta y_j(x)}{\Delta t_j} = \frac{y(t_{j+1},x) - y(t_j,x)}{t_{j+1} - t_j}. \tag{3.21}$$

Using this derivative for variable substitution in Equation (3.19)

$$\int_{t_j}^{t_{j+1}} \varphi(x,y)\frac{|\Delta y_j(x)|}{\Delta t_j} dt = \int_{t_j}^{t_{j+1}} s(t)\mathrm{d}t \tag{3.22}$$

finally yields the equation for the density in the parallel-coordinates domain:

$$\varphi_j(x,y) = \frac{\Delta t_j}{\Delta y_j(x)} s(t). \tag{3.23}$$

Note that the sampling density is typically set to $s(t) = 1$.

Figure 3.6 illustrates the result for a dataset with 2 700 samples using the traditional scattering approach with constant line density, the model presented by Johansson et al. (2007), and the model based on mass-conservation.

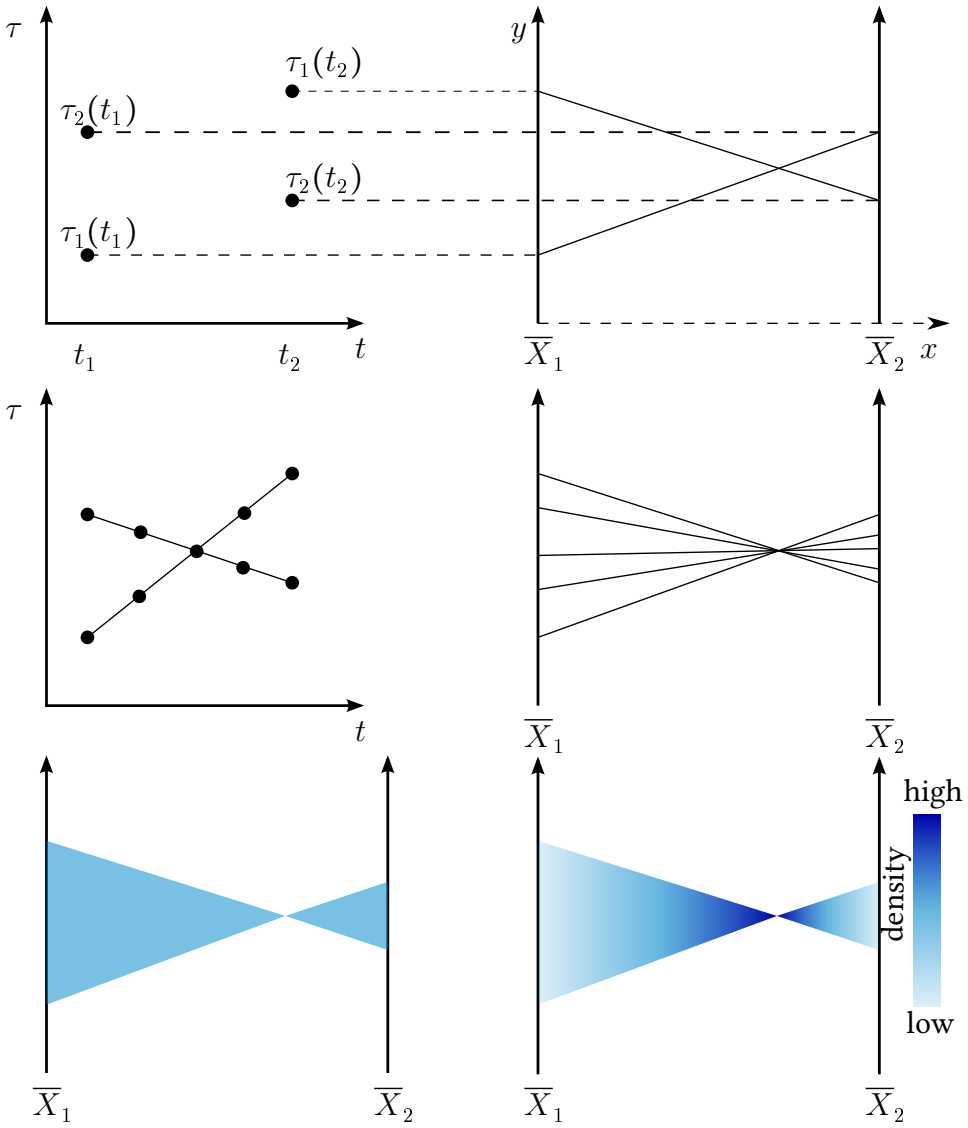



**Figure 3.3:** Footprints of samples from time series mapped from the spatio-temporal domain (left) to the parallel-coordinates domain (right). Time-dependent variables $\tau_i$ are represented as axes $\overline{X}_i$, and for each point in time $t_j$ a line is obtained in parallel coordinates (top). Resampling between $t_j$ and $t_{j+1}$ with linear interpolation results in lines with constant density (middle). Note how the vertical density of lines depends on the horizontal position $x$. The model using mass conservation (bottom, right) reflects this dependency, while previous models (Johansson et al., 2007) (bottom, left) assume constant density in parallel coordinates.

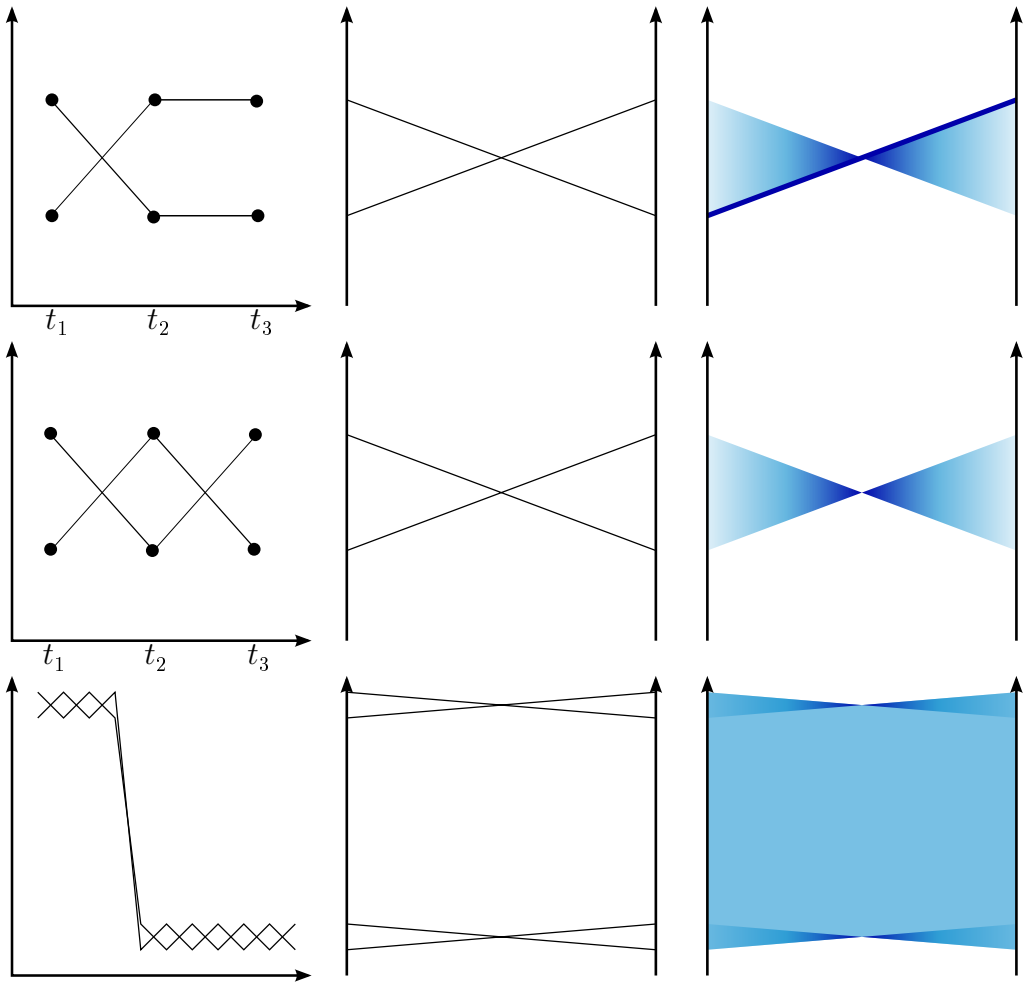



**Figure 3.4:** Ambiguous cases for discrete parallel coordinates. The top and middle time series are mapped to the same visual encoding in discrete parallel coordinates (center column). With a density representation, these cases are disambiguated (right column). The positive correlation of the two time series between time points $t_2$ and $t_3$ (top) contributes to the dense line in continuous parallel coordinates, whereas a negative correlation over all time points (center) results in a different pattern. Note that densities are normalized independently for each plot. Continuous parallel coordinates further succeed in visualizing local (i.e. within short periods of time) and global (i.e. within the whole time series) patterns simultaneously, as the bottom example illustrates. Here, two local negative correlations and one global positive correlation are contained between the two time series. Note that the same colormap as in Figure 3.3 was applied.

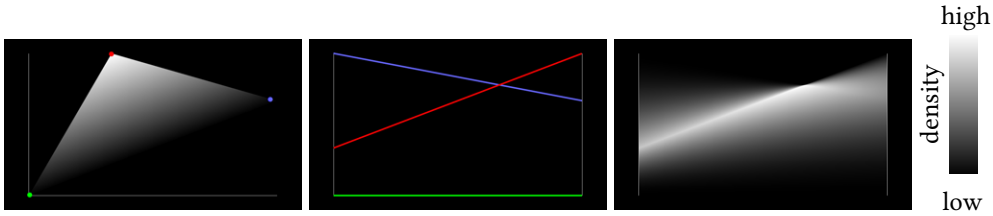



**Figure 3.5:** The reference triangle with continuous density in the data domain (left), its footprint in the parallel-coordinates domain (middle), and the density plot using an analytic solution for triangulated data (right). The triangle vertices and respective lines in the footprint are marked red, green, and blue. The density plots computed with numerical integration (gathering and scattering) are indistinguishable from the analytic solution. The respective $l^2$ distances are given in the main text.

## 3.4 Implementation

This section presents implementations of the different computational models introduced in Sections 3.3.2, 3.3.3, and 3.3.4. For triangulated data, each method will shortly be explained and applied to a test dataset comprising a single triangle with known density distribution in the data domain (see Figure 3.5) in order to evaluate the numerical quality of the different methods. For time series, the footprint is compared to discrete parallel coordinates and the model of Johansson et al. (2007).

The implementations are based on C++ and OpenGL with GLSL. All calculations were performed using a 2048×2048 floating-point render target.

### 3.4.1 Triangulated Data

In Section 3.3.3, the contribution of the piecewise linear density given on a triangle to $\varphi(\overline{\ell})$ was reduced to a single equation depending only on $\overline{\ell}$ and the densities at the triangle vertices. This can be used to implement a rasterization of line densities in parallel coordinates. After projecting a tetrahedral mesh from the spatial domain to the data domain, the density distribution of each triangle is mapped to parallel coordinates according to (3.18). Due to the linear density model, the total density $\varphi(\overline{\ell})$ can then be computed using additive blending.

Using a floating-precision buffer as render target, the density is computed for each texel individually, such that the algorithm can easily be adapted for a GPU implementation. In particular, fast interpolation can be exploited for the computation of parameters to (3.18). Hence, the primitives have to be generated, such that the necessary parameters can be attached as texture coordinates. As can easily be seen in Figure 3.5, the footprint of a triangle in parallel coordinates consists of three lines, each representing one vertex of the triangle. In turn, each

line may intersect any other line, such that a minimum of zero and a maximum of three intersections may occur. Dividing the horizontal axis at each intersection yields up to four line segments, each consisting of two quadrilaterals. Rendering each quadrilateral with attached texture coordinates representing $u$, $\Delta y_b$, and $\Delta y_c$ then allows for the evaluation of Equations (3.18) and (3.15) in a graphics processing unit (GPU) fragment program. For the special case $\overline{a}_y = \overline{c}_y = \overline{b}_y$, a constant value can be stored in a separate channel of the render target in order to mark the corresponding pixel. In future, this may be considered for the final display, e.g. to indicate degenerate cases by visual means. For a triangle $\Delta_{ABC}$, the algorithm consists of the following steps:

1. Determine all intersections of lines $\overline{A}$, $\overline{B}$, and $\overline{C}$ and divide the horizontal axis into line segments accordingly.

2. Determine upper and lower quadrilaterals (treat triangles as degenerate quadrilaterals) and attach parameters $u$, $\Delta y_b$, $\Delta y_c$ as texture coordinates to the corresponding vertices.

3. Render quadrilaterals with fragment program enabled.

Figure 3.5 shows the result of the implementation for the reference triangle, after density normalization to $[0, 1]$. As this approach represents the analytic solution to the mathematical model of continuous parallel coordinates, it may also be considered as ground truth for comparison purposes.

### 3.4.2 Numerical Integration

Given a 2D scalar density field, the gathering approach presented earlier accumulates densities for each $\overline{\ell}$ along the dual line $\ell$ in the data domain. For implementation, the continuous reference triangle can be used with densities stored in a floating-point render target to compute line integrals according to the gathering approach. Density values for parallel coordinates are stored in a floating-point render target of the same resolution. Then, for each texel in the parallel-coordinates domain, the dual line is sampled from the input field. In order to properly reconstruct $\sigma$, the sampling rate should be set to the respective Nyquist rate. Due to the texel-based computation, the algorithm is perfectly suited for hardware-accelerated computation. As there is no visible difference to ground truth, the $l^2$ norm of the difference vector of the respective render targets was computed to obtain a quantitative distance measure. After normalization, the relative distance, i.e. $\frac{l^2}{N}$ with $N = 2048^2$, of the gathering approach to ground truth is approximately $1.2 \cdot 10^{-7}$. The error is negligible and, therefore, the gathering approach is an appropriate alternative to the analytic solution. The sources of the small difference between the numerical and the analytic solution include the sampled representation of the scatterplot, the numerical integration,

and the interpolation when accessing the data domain. All these error sources depend on the resolution of the data-domain representation. Therefore, the quality of the numerical solution can be controlled by adapting the resolution of the intermediate scatterplot texture. In contrast to the analytic solution using triangulated data, the gathering approach does not depend on the size of the dataset, such that it may be used in a fast, although less accurate, implementation for the computation of continuous parallel coordinates. Note that, for the efficient rendering of continuous scatterplots, Bachthaler and Weiskopf (2009) proposed adaptive techniques supporting a wide class of reconstruction filters, including trilinear interpolation.

A scattering approach was implemented according to the generic scattering algorithm presented in Section 3.3.2. Samples are drawn randomly on a triangle in the data domain using rejection sampling, i.e. observations are sampled from the surrounding rectangle, rejecting samples outside the triangle and linearly interpolating those accepted. Then, for each sample $P_i$, the dual line $\overline{P}_i$ in parallel coordinates is rendered as a white polyline with density being represented by the respective alpha value (i.e. $\alpha \leftarrow \sigma(P_i)$). The overall density $\varphi(\overline{\ell})$ according to (3.6) is obtained by accumulating alpha values of each line intersecting $\overline{\ell}$, which is conveniently implemented using additive blending. After normalizing, the resulting image is finally low-pass filtered using a Gaussian $5 \times 5$ kernel in order to compensate for aliasing artifacts. Again, there is no visible difference to ground truth. The relative $l^2$ difference to ground truth is approximately $2.75 \cdot 10^{-6}$, i.e. about one order of magnitude higher than for the gathering approach. This could be further improved by increasing the number of samples.

### 3.4.3 Time Series

The footprints for time-series data are rendered in a similar fashion to the triangular footprints of the previous section. For each footprint, two successive points $t_j$ and $t_{j+1}$ are sampled in the spatio-temporal domain, as illustrated in Figure 3.3. The geometry used for the final footprint depends on the location of the intersection point of the two lines:

- If the intersection occurs between the axes, two triangles are rendered as illustrated in Figure 3.3.
- If the intersection occurs to the left of the axes or the lines are parallel, a single quadrilateral is rendered.

Again, the density is computed for every texel independently using a GLSL shader. The total density at a pixel is then computed with additive blending.

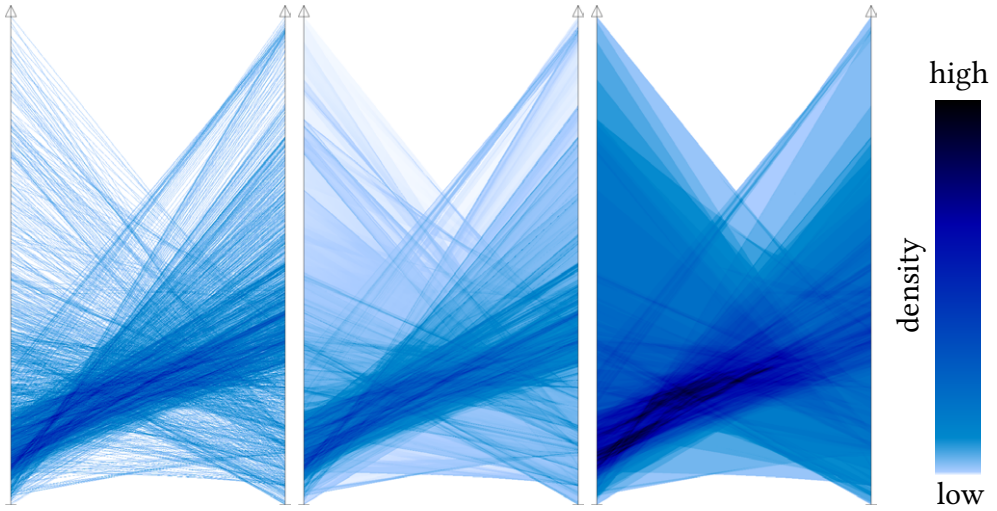



**Figure 3.6:** Comparing different implementations for rendering time series in parallel coordinates. Left: traditional scattering approach with constant density for every line. Center: continuous parallel coordinates using mass conservation. Right: the approach of Johansson et al. (2007).

Figure 3.6 compares three implementations for continuous parallel coordinates for a sample of 2 700 time points. Again, the mass-conservation model is almost identical to the line-based scattering approach using traditional parallel coordinates.

## 3.5 Results

In this section, discrete density-based and continuous parallel coordinates are compared for an artificial dataset. A case study employing continuous parallel coordinates for computational fluid dynamics is given in Section 7.1. In all examples, discrete parallel coordinates are created by drawing one polyline for each sample in the spatial domain. For continuous parallel coordinates, a 2D density field is computed using the projected tetrahedra algorithm as in (Bachthaler and Weiskopf, 2008). The resulting triangles in the data domain are then mapped to parallel coordinates as described in Section 3.4.1. For the gathering approach, the density field is loaded into a texture and the density in parallel coordinates is computed for every pixel as described in Section 3.4.2. In all three approaches, a render-target texture is used to obtain floating-point precision for the computation of densities. For discrete parallel coordinates, the density of a pixel is computed with additive blending. Before the content of the texture is written to the framebuffer, densities are normalized to the same density range. Furthermore, a logarithmic colormap is applied to the normalized densities, such that low densities are shown in black/dark-blue, mid-density

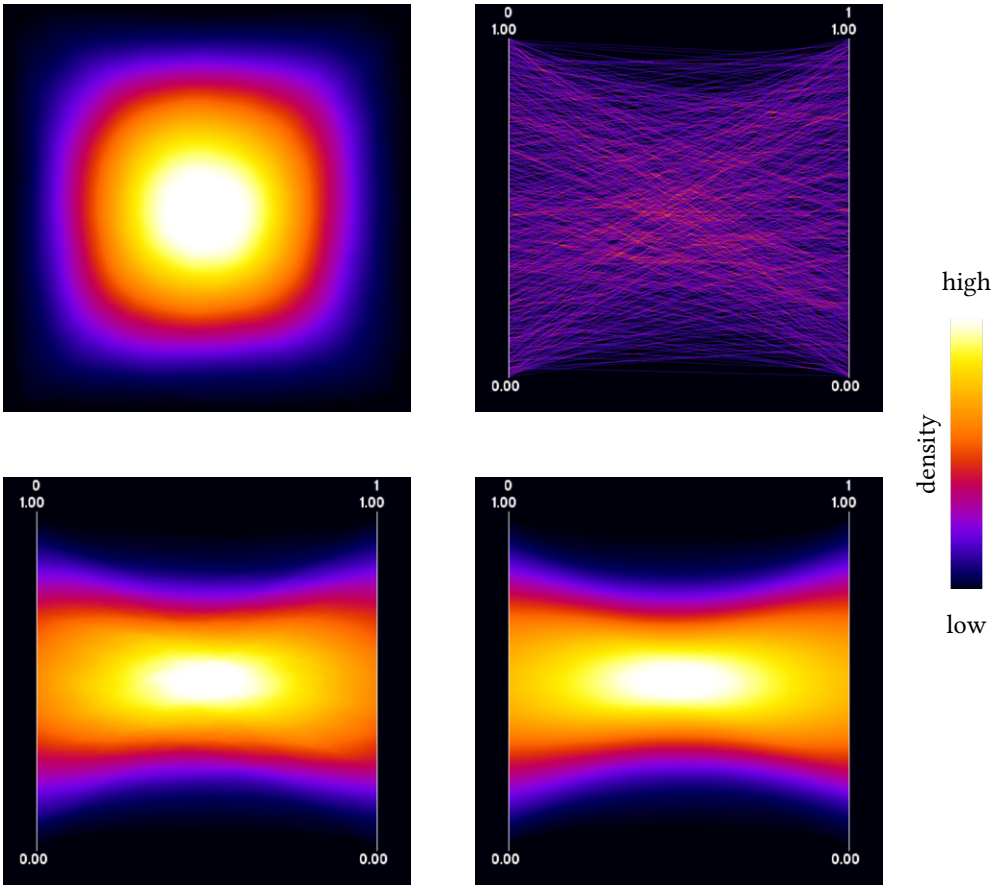



**Figure 3.7:** Samples from an artificial dataset of two Gaussian distributions on a $10 \times 10 \times 10$ grid in the spatio-temporal domain, mapped to a continuous scatterplot (top, left), discrete density-based parallel coordinates (top, right) and continuous parallel coordinates using projected tetrahedra (bottom, left) and the gathering approach (bottom, right). While discrete density-based parallel coordinates show the expected hyperbolic shape, the density is not as well reflected as for continuous parallel coordinates (bottom). Note the similarity between the gathering approach and the approach using projected tetrahedra.

values are shown in red, and high-density values are mapped to yellow/white.

The first example was created from an artificial dataset composed of randomly drawn samples from two Gaussian distributions on a uniform $10 \times 10 \times 10$ grid. Figure 3.7 shows the results for discrete and continuous parallel coordinates. This example illustrates the duality between ellipses in the data domain and hyperbolas in the parallel-coordinates domain (see also Figure 2.5, page 2.5).

A performance comparison of discrete and continuous parallel coordinates is provided in Table 3.1. Although the gathering approach allows for highly interac-

tive computation of continuous parallel coordinates while being independent of the spatial resolution, it depends on the computation of continuous scatterplots, which make up most of the total computation time for the triangulated data algorithm presented by Bachthaler and Weiskopf (2008). Note that more efficient rendering techniques for continuous scatterplots were proposed (Bachthaler and Weiskopf, 2009) that can be used with the gathering approach. An algorithm for the progressive rendering of continuous parallel coordinates will be presented in the next chapter.

**Table 3.1:** Computation time in ms for continuous scatterplots (CS), continuous parallel coordinates (CPC), and discrete parallel coordinates for different spatial resolutions of the hurricane Isabel dataset (see also Section 7.1). Measurements were taken on a Linux PC with an Intel(R) Core(TM) 2 Quad CPU running at 2.4 GHz with 4 GB RAM and an NVIDIA GeForce 8800 GTX graphics card.

| | $50 \times 50 \times 10$ | $100 \times 100 \times 20$ | $500 \times 500 \times 100$ |
|---|---|---|---|
| CS | 5 022 | 7 848 | 661 164 |
| CPC | 5 | 4 | 4 |
| PC | 10 | 80 | 36 631 |

CHAPTER 4

# Splatting and Progressive Refinement

## 4.1 Introduction

As was illustrated in the previous chapter, rendering continuous parallel coordinates for multidimensional multivariate data depends on rendering continuous scatterplots, constituting the major bottleneck for the total computation time (see Table 3.1). Also, analytic solutions for continuous statistical plots have only been presented for linear interpolation schemes. While Bachthaler and Weiskopf (2009) presented an adaptive approach to continuous scatterplots that can be used with arbitrary interpolation schemes, it assumes constant density distributions within single cells in the data domain.

This chapter[1] presents a progressive rendering approach for continuous bivariate scatterplots and multivariate parallel-coordinates plots that allows for flexible density reconstruction kernels, free choice of sample positions, and any type of interpolation scheme while achieving better image quality at comparable rendering speed. To accomplish high frame rates for interactive data exploration, a forward-mapping technique inspired by splatting for direct volume rendering is employed to compute the influence of samples to the final image and successively approximate the true representation of the plot. To this end, a novel closed-form analytic description of the splatted footprint of Gaussian input kernels for scatterplots and parallel coordinates is presented. A new progressive refinement

[1] **Parts of this chapter have been published in:** J. Heinrich, S. Bachthaler, and D. Weiskopf. Progressive splatting of continuous scatterplots and parallel coordinates. *Computer Graphics Forum*, 30(3):653–662, 2011.

algorithm is introduced that allows one to obtain initial results extremely fast and hence complements a very useful property of continuous data plots: for many datasets, a small subsample of the full data may provide a good approximation to the final density distribution[2], making progressive refinement the ideal algorithm for rendering. The chapter is completed by performance and image-quality analysis of the GPU implementation of the presented algorithm.

## 4.2 Related Work

This section covers related work on splatting and progressive refinement techniques for statistical plots. For related work on density representations and clutter reduction, see Sections 2.3.1, 3.2, and 2.6.1.

The application of splatting to parallel coordinates has recently been proposed by Zhou et al. (2009). In their approach, a Gaussian reconstruction filter is used to adjust the density of discrete samples in a fixed neighborhood of a line segment. By successively accumulating these line splats, a dense field of lines is obtained. While the idea of using a reconstruction method is related to splatting for volume rendering, it is important to note that the approach of Zhou et al. reconstructs densities in image space and thus relies on discrete samples. In contrast, the approach presented in this chapter reconstructs densities in the spatial domain and maps the result to the respective image coordinate system.

The construction algorithm presented here adopts and extends the splatting method by Westover (1989), which was introduced as a forward-mapping algorithm for direct volume rendering. In contrast to raytracing techniques where pixel intensities are computed by mapping the image plane to the data space, volumetric splatting computes the contribution of samples from the data space to the image plane. Since ray integration within a sample's reconstruction kernel is independent of its density, the integral can be pre-computed for a given view direction. The resulting image-plane footprint of the kernel is then used to compute the projected image of all data space samples and only has to be recomputed if the viewing direction changes. For continuous statistical plots, however, the viewing transformation depends on the data, such that footprints have to be computed for every sample individually. Furthermore, the splats for parallel coordinates undergo further transformation according to the point–line duality between Cartesian and parallel-coordinates domains; therefore, those splats differ substantially from those in scatterplots and direct volume rendering.

An approach similar to splatting densities has recently been presented by Feng et al. (2010) for the visualization of uncertain data samples. In their work, a probability density function is estimated using normally distributed, uncorrelated kernels. Hence, all samples in the scatterplot are represented by scaled Gaussian footprints. In contrast, the model for data given on the spatio-temporal

[2] see Figure 7.1 for an example

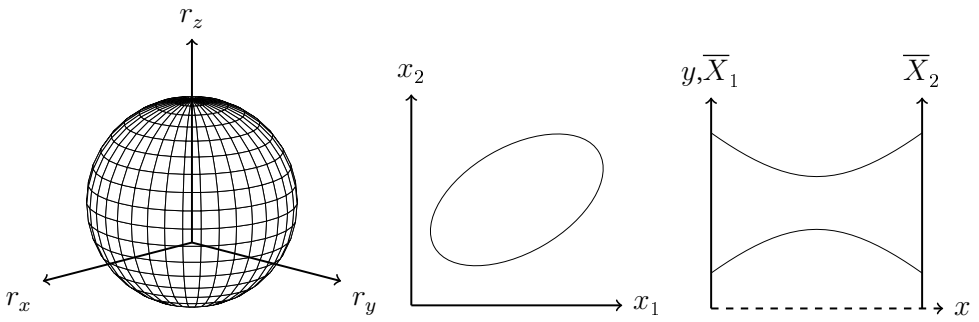



**Figure 4.1:** A spherical reconstruction kernel in the spatio-temporal domain (left) maps to an ellipse in the data domain (middle) and a hyperbola in the parallel-coordinates domain (right).

domain uses gradient information contained in the data to transform the density distribution to both scatterplots and parallel coordinates. Furthermore, densities are derived analytically to allow for the precomputation of splat footprints.

Laur and Hanrahan (1991) extended the original volume splatting approach with a progressive refinement algorithm. They use an octree to subdivide the spatial grid into a hierarchical structure and then draw splats covering the size of a cell in the octree. A similar refinement strategy is employed here, although with arbitrary resampling techniques instead of a subdivision approach.

Rosenbaum et al. (2012) introduced a progressive refinement algorithm for parallel coordinates based on hierarchical subdivison techniques and data compression in the data domain. As there is no spatio-temporal domain involved in progressive parallel coordinates, an appropriate density function has to be estimated from data observations instead. While they also employ density-based rendering, their approach uses convex polygons to represent aggregated data and does not adhere to the point–line duality.

## 4.3 Footprint Computation

This section describes the computation of density footprints for reconstruction kernels in the original spatio-temporal domain of the data and the respective mapping to scatterplots and parallel coordinates. Note that we use terminology from measure theory and thereby use weighted, non-normalized densities (as opposed to the probability density typically used in the statistics literature). The two main ingredients for the mathematical model of continuous statistical plots are:

- Density function $s(\boldsymbol{r})\colon \mathbb{R}^M \to \mathbb{R},\ \boldsymbol{r} \mapsto s(\boldsymbol{r})$, which describes the scalar-valued density over the spatio-temporal domain of the data set, i.e., the importance of the respective spatial part of the data.

- Map $\tau: \mathbb{R}^M \longrightarrow \mathbb{R}^N,\ \boldsymbol{r} \mapsto \tau(\boldsymbol{r})$, which represents the multivariate data set defined on the same spatio-temporal domain as above.

In typical applications of multidimensional multivariate visualization, $\tau$ is given on a grid where each grid point has attached a respective $N$-tuple of data values. In-between values (away from grid points) are reconstructed by an explicitly or implicitly given reconstruction scheme—typically by trilinear interpolation. For 2D scatterplots, $N = 2$ data attributes are visualized, e.g., temperature and pressure from a simulation in computational fluid dynamics.

The density $s$ in the spatio-temporal domain is typically provided by the user or originates from some kind of external feature definition. If not stated otherwise, a constant density $s(\boldsymbol{r}) = 1$ can be assumed.

The mathematical problem setting can now be formulated as follows: what are the transformed density functions $\sigma$ in the data domain and $\varphi$ in the parallel-coordinates domain? $\sigma$ and $\varphi$ explicitly depend on the density $s$ in the spatio-temporal domain and are implicitly affected by the map $\tau$, which connects from the spatio-temporal domain to the other two domains. While the previous chapter covered the generic model for the transformation of $\sigma$ from the data domain to $\varphi$ in the parallel-coordinates domain and presented footprints for tetrahedral grids and time series with linear interpolation in the spatio-temporal domain, this chapter presents the computation of footprints in the data domain and the parallel-coordinates domain for Gaussian density kernels in the spatio-temporal domain.

In the following, we first define splats in the spatial domain and discuss the map $\tau$ in the context of sampling at the center points of splats (Section 4.3.1). Then, we restrict the discussion to the transformation of a single splat template, applying this model to continuous 2D scatterplots (Section 4.3.2). Finally, we describe the analogous transformation of splats to parallel coordinates (Section 4.3.3).

### 4.3.1 Spatio-Temporal Domain and Data-Set Function

Following the splatting approach for direct volume visualization (Westover, 1989), the density in the spatio-temporal domain is represented by a weighted sum of kernels at various spatial locations. Then, the overall density on $\mathbb{R}^3$ is given by

$$s_{\text{overall}}(\boldsymbol{r}) = \sum_i w_i \, s_i(\boldsymbol{r})$$

with scalar-valued weights $w_i$ and kernels $s_i$.

Similar to splatting in volume rendering, the kernels $s_i$ are assumed to be derived from a single template function that is just shifted to different positions. Furthermore, such a template is usually assumed to be spherically symmetric due to reasons of isotropy.

The main idea of splatting is that the template is transformed in a pre-processing step to form a splat. The transformation of the overall density $s_{\text{overall}}$ is then computed by overlaying the splats with the same weights $w_i$. This approach requires that transformation and weighted summation are commutative, which is true for continuous scatterplots and parallel coordinates because both are computed by linear operators.

Depending on the kernel used in the spatial domain, different footprint functions are obtained for the data domain. For spherical volume reconstruction kernels, the generic footprint is a circle centered at the image plane projection of a sample (Westover, 1990). In the next sections, footprints for spherical reconstruction kernels in the spatial domain are derived. This footprint can then be mapped to parallel coordinates. Now, let's consider the concrete example of a 3D Gaussian kernel in the spatial domain:

$$s_i(\boldsymbol{r}) = e^{\frac{-\|\boldsymbol{r}-\boldsymbol{r}_i\|^2}{k^2}}.$$

The kernel $s_i$ is centered at the point $\boldsymbol{r}_i$ with relative "radius", *bandwidth*, or *smoothing factor* $k$. The Gaussian kernel is popular in volume rendering and statistics due to its fast fall-off behavior with increasing distance from the kernel center. Please note that non-normalized Gaussians are applied here; normalization is implicitly absorbed by the weights $w_i$.

The sample positions $\boldsymbol{r}_i$ may be arbitrarily chosen. In particular, they are independent from positions of grid points of the data set. Typically, the sample positions $\boldsymbol{r}_i$ are evenly distributed in space, e.g., by putting them on a regular sampling grid or by applying low-discrepancy point sets (see Section 4.5). The usual case of constant overall density $s_{\text{overall}}$ can be implemented by even distribution of sample positions and constant weights $w_i$.

In the rest of this section, we will only consider a single Gaussian kernel. For simplicity of notation, this kernel is denoted

$$s(\boldsymbol{r}) = e^{\frac{-\|\boldsymbol{r}-\boldsymbol{r}_0\|^2}{k^2}} \tag{4.1}$$

and centered at $\boldsymbol{r}_0$.

Furthermore, we assume that $\tau$ is a $C^1$ continuous function. Then, $\tau$ can be approximated by Taylor expansion around $\boldsymbol{r}_0$ up to first order:

$$\tau(\boldsymbol{r}) = \tau(\boldsymbol{r}_0) + D_\tau(\boldsymbol{r}_0)(\boldsymbol{r}-\boldsymbol{r}_0) + O(\|\boldsymbol{r}-\boldsymbol{r}_0\|^2),$$

where $D_\tau(\boldsymbol{r}_0)$ is the Jacobian of $\tau$ (i.e., the matrix of partial derivatives with respect to spatial locations) as evaluated at $\boldsymbol{r}_0$. This approximation leads to a linearization of $\tau$ around $\boldsymbol{r}_0$, which is appropriate within a sufficiently small neighborhood around the kernel center, i.e., for sufficiently small kernels. In the limit process of infinitesimally small kernels, the linearized $\tau$ converges to the

true map. In the remainder of this section, we will work with the linearization of $\tau$. This implies that derivatives of $\tau$ are constant. We also assume non-degenerate cases where $\tau$ is not constant and its partial derivatives lead to linearly independent gradient vectors.

### 4.3.2 Data Domain

Let us now transform the density function described by the kernel $s(\boldsymbol{r})$ in the spatial domain to the corresponding density function $\sigma(\boldsymbol{x})$ at position $\boldsymbol{x} = (x_1, x_2)^{\mathrm{T}}$ in the 2D data domain. We also call $\sigma(\boldsymbol{x})$ footprint of the splat or, in short, just splat or footprint.

Using Equation (9) from reference (Bachthaler and Weiskopf, 2008), the footprint is

$$\sigma(\boldsymbol{x}) = \int_{\tau^{-1}(\boldsymbol{x})} \frac{s(\boldsymbol{r})}{|\mathrm{Vol}(D_\tau(\boldsymbol{r}))|} \mathrm{d}r,$$

where $D_\tau$ denotes the $2{\times}3$ Jacobi matrix for the bivariate data map $\tau$. The volume measure $|\mathrm{Vol}(D_\tau(\boldsymbol{r}))|$ is given by the vector cross product of the two gradients of the components of $\boldsymbol{x}$ with respect to the spatial domain (see Equation (10) in reference (Bachthaler and Weiskopf, 2008)):

$$|\mathrm{Vol}(D_\tau(\boldsymbol{r}))| = \|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|.$$

Since $\tau$ is linear, the partial derivatives are constant and the volume measure can be moved outside the integral. In addition, by using the definition of the Gaussian kernel from Equation (4.1), we obtain:

$$\sigma(\boldsymbol{x}) = \frac{1}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} \int_{\tau^{-1}(\boldsymbol{x})} e^{\frac{-\|\boldsymbol{r}-\boldsymbol{r}_0\|^2}{k^2}} \mathrm{d}r. \tag{4.2}$$

Since $\tau$ is linear, the isosurfaces corresponding to isovalues $x_1$ and $x_2$ are planes with normal vectors $\nabla \boldsymbol{x}_1$ and $\nabla \boldsymbol{x}_2$, respectively. Then, the intersection of the two isosurfaces (identical to $\tau^{-1}(\boldsymbol{x})$) is a straight line. Integration of the Gaussian kernel along this infinitely long line resembles the projection of splats in volume rendering. Therefore, similar to volume rendering, we also retain Gaussian splats in the scatterplot domain. According to the mathematical derivation in Appendix B, the density in the scatterplot domain reads

$$\sigma(\boldsymbol{x}) = \frac{\sqrt{\pi}}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} e^{-(\boldsymbol{x}-\boldsymbol{x}_0)^{\mathrm{T}} E (\boldsymbol{x}-\boldsymbol{x}_0)}. \tag{4.3}$$

Here, $\boldsymbol{x}_0 = \tau(\boldsymbol{r}_0)$. The matrix $E = \frac{1}{k^2}\left(D_\tau D_\tau^{\mathrm{T}}\right)^{-1}$ is a $2 \times 2$ symmetric, positive definite matrix that can be used to parameterize the Gaussian kernel. Following Westover (1989), $E$ defines the extents and rotation angle of the screen-space

ellipse that is obtained after transformation of a sphere using the Jacobian $D_\tau(\boldsymbol{r})$ as generalized viewing transformation. The transformation to the standard 2D Gaussian only depends on the scales $S_x$ and $S_y$ along the main axes of the ellipse as well as the angle $\theta$ by which the standard Gaussian is rotated. Due to the symmetry of $E$, $S_x$, $S_y$, and $\theta$ are given by the eigenvalues and eigenvectors of the inverse transformation $E^{-1}$:

$$E^{-1} = k^2 D_\tau D_\tau^{\mathrm{T}} = k^2 \begin{bmatrix} a & b \\ b & c \end{bmatrix}$$

with

$$\begin{aligned} a &= \nabla \boldsymbol{x}_1^{\mathrm{T}} \nabla \boldsymbol{x}_1 \\ b &= \nabla \boldsymbol{x}_1^{\mathrm{T}} \nabla \boldsymbol{x}_2 \\ c &= \nabla \boldsymbol{x}_2^{\mathrm{T}} \nabla \boldsymbol{x}_2. \end{aligned}$$

After calculating the characteristic polynomial of $E^{-1}$ and with

$$e = \sqrt{(a-c)^2 + 4b^2},$$

the scales are computed from the eigenvalues

$$S_x = \sqrt{\lambda_1},\ S_y = \sqrt{\lambda_2},$$

where $\lambda_1 = \frac{k^2}{2}(a + c + e)$ and $\lambda_2 = \frac{k^2}{2}(a + c - e)$. If $b = 0$, $E^{-1}$ is a scaling matrix such that $\theta = 0$. With $b \neq 0$, the eigenvectors can be written as

$$\boldsymbol{v}_1 = (b, \lambda_1 - a)^{\mathrm{T}},\ \boldsymbol{v}_2 = (b, \lambda_2 - a)^{\mathrm{T}}$$

and $\theta$ corresponds to the angle of the eigenvectors to the unit vector $(1, 0)^{\mathrm{T}}$:

$$\theta = \arccos\left(\frac{b}{\sqrt{b^2 + (\lambda_1 - a)^2}}\right).$$

Now, the footprint of any sample can be transformed to a generic template footprint (usually the standard Gaussian defined by $e^{-\|\boldsymbol{x}\|^2}$) using a scaling matrix with $S_x$ and $S_y$ multiplied by a matrix for rotation around the $z$-axis with angle $\theta$.

Although the splat computation for scatterplots resembles the one for volume rendering, there is an important difference: the scatterplot-domain splat additionally depends on $\tau$, which is not the case in volume rendering.

### 4.3.3 Parallel-Coordinates Domain

Analogous to the previous transformation from spatial domain to scatterplot domain, let us now examine the transformation of density to the parallel-coordinates domain. According to Equation (3.6) from Chapter 3, the footprint in parallel coordinates reads

$$\varphi(x,y) = \int_\ell \frac{\sigma(\boldsymbol{\ell}(\lambda))}{\|\tilde{\boldsymbol{n}}\|} \mathrm{d}\lambda, \tag{4.4}$$

where $\varphi$ is the density of a point $\bar{\ell} : (x,y)$ in parallel coordinates with dual line

$$\boldsymbol{\ell}(\lambda) = \frac{y}{\|\tilde{\boldsymbol{n}}\|}\boldsymbol{n} + t\boldsymbol{n}^\perp \tag{4.5}$$

in the coordinates of the scatterplot domain. The vector $\boldsymbol{n}$—the vector that is perpendicular to $\boldsymbol{\ell}$—reads

$$\boldsymbol{n} = \frac{\tilde{\boldsymbol{n}}}{\|\tilde{\boldsymbol{n}}\|}, \ \tilde{\boldsymbol{n}} = (1-x, x)^\mathrm{T}.$$

Note that $\boldsymbol{n}$ and its perpendicular tangent vector, $\boldsymbol{n}^\perp$, only depend on $x$, whereas the distance of $\boldsymbol{\ell}$ to the origin,

$$\frac{y}{\|\tilde{\boldsymbol{n}}\|}$$

linearly depends on $y$.

Plugging Equations (4.5) and (4.3) into Equation (4.4) yields the dual footprint of Equation (4.3) in parallel coordinates:

$$\begin{aligned}
\varphi(x,y) &= \frac{1}{\|\tilde{\boldsymbol{n}}\|}\int_\ell \sigma(\boldsymbol{\ell}(t))\mathrm{d}t \\
&= \frac{\sqrt{\pi}}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|\,\|\tilde{\boldsymbol{n}}\|}\int_{-\infty}^{\infty} e^{-(\boldsymbol{\ell}(t)-\boldsymbol{x}_0)^\mathrm{T} E (\boldsymbol{\ell}(t)-\boldsymbol{x}_0)}\,\mathrm{d}t \\
&= \frac{\pi}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|\,\|\tilde{\boldsymbol{n}}\|\,\|\boldsymbol{d}\|} e^{-\frac{\|\boldsymbol{v}\|^2 + A^2}{k^2}}
\end{aligned} \tag{4.6}$$

with $\boldsymbol{v} = D_\tau(\boldsymbol{r})^{-1}(\frac{y}{\|\tilde{\boldsymbol{n}}\|}\boldsymbol{n} - \boldsymbol{x}_0)$, $A = \frac{\boldsymbol{d}\cdot\boldsymbol{v}}{\|\boldsymbol{d}\|}$, and $\boldsymbol{d} = D_\tau(\boldsymbol{r})^{-1}\boldsymbol{n}^\perp$. Equation (4.6) can be derived using a similar approach as for Equation (4.3).

## 4.4 Sampling and Progressive Refinement

The previous section provided the basis for rendering a single splat in the continuous statistical plot. According to the generic scattering algorithm presented in Section 3.4.2, the overall plot is obtained by applying this process to many

different splats that densely cover the spatial domain, leading to an appropriate representation of the overall density by accumulating the splats with additive blending. Besides the data values, the partial derivatives of the data are reconstructed at the splat sample so that the position, size, and orientation of the splat can be determined. In this way, the generic footprint template is transformed according to Section 4.3. Using the inverse transformation, a splat is rendered as a rectangle with the correct texture coordinates and density values.

The generic template footprints for scatterplots are discretized in a 2D texture during pre-processing. The 2D standard Gaussian $e^{-\|\boldsymbol{x}\|^2}$ is sampled on the uniform grid of the texture, which then serves as a lookup table during rendering. The rendering algorithm only needs support for 2D textures and blending. Therefore, it lends itself immediately to direct and efficient GPU implementation. Due to the rotation–translation duality between parallel coordinates and Cartesian coordinates, the hyperbolic standard footprint from Equation (4.6) cannot be sampled to a single 2D texture, but can be evaluated efficiently on the GPU using a fragment program.

Conventional discrete scatterplots and parallel-coordinates plots of multidimensional multivariate data are commonly used to visualize the data values attached at the input grid points. For large datasets, however, rendering becomes prohibitively slow hindering interactive exploration of the data. In this case, splatted continuous plots allow us to trade accuracy for rendering performance by resampling the data at a lower sampling rate. This may be implemented by skipping some of the input data points or by resampling at a few, freely chosen positions on the spatial domain. In contrast, for very small datasets, additional samples may be distributed over the spatial domain to improve image quality. A key observation is that the continuous plots are based on samples that are completely independent from the number and positions of the grid points of the dataset. It is recommended to make use of this flexibility by employing sampling positions obtained from low-discrepancy sequences (Niederreiter, 1992), such as Halton or Hammersley sequences, because they guarantee even coverage of the spatial domain and, at the same time, avoid aliasing or moiré artifacts from regular sampling.

The splatting technique is further improved by extending it to progressive rendering. Due to the linear superposition of splats, progressively sampled intermediate images can be combined by linear superposition as well. More specifically, a sequence of several, independent continuous plots of low sampling resolution is generated that are then accumulated in a separate image (e.g., an offline rendering target on the GPU). To guarantee mass conservation of the transformed densities (see details on mass conservation in (Bachthaler and Weiskopf, 2008, 2009; Heinrich and Weiskopf, 2009)), single images $I_1$ and $I_2$ are composited using alpha blending (Porter and Duff, 1984) and a fixed value for $\alpha$:

$$I = \alpha I_1 + (1-\alpha) I_2, \quad \text{with } 0 \le \alpha \le 1.$$

If both intermediate images observe mass conservation (i.e., each has the same overall mass $M$ as accumulated from the densities or all pixels), then the blended image is guaranteed to have identical mass as well because mass is also subject to alpha blending: $\alpha M + (1 - \alpha)M = M$. Here, $\alpha$ is a parameter controlling the contribution of densities from a single splatting step to the final image.

The footprints from Section 4.3 further depend on the choice of the smoothing parameter $k$, which is a measure for the "radius" or bandwidth of the Gaussian kernel in the spatial domain. This parameter is transported to the scatterplot and parallel-coordinate domains, where it affects the extents of the footprints to be drawn. On the one hand, large kernels provide a better coverage of, and higher overdraw on, the reconstructed spatial area and thus allow one to reduce the sampling resolution. On the other hand, large splats introduce a large error to the overall density, resulting in potentially overblurred images. Furthermore, large splats have a negative impact on rendering performance as the size of the primitives that have to be processed increases. Analogously, small kernels produce a more accurate, but potentially non-smooth sampling of the density distribution.

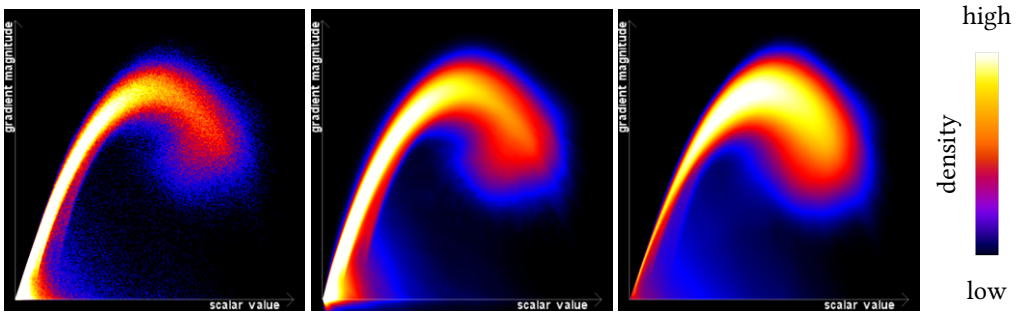
To describe the relative smoothness of the continuous statistical plots, the *coarseness*

$$c = \frac{S}{kn} \tag{4.7}$$

is introduced where $n$ is the number of splats, $S$ is the total number of grid points and $k$ is the bandwidth of the Gaussian reconstruction kernel in the spatial domain. As a reference for uniform grids with spacing one, the coarseness equals one if all grid points are sampled using a Gaussian kernel with $k = 1$. According to Equation (4.7), the relative coarseness increases with decreasing number of splats and with decreasing splat size. This can be used in addition to alpha blending to obtain a progressively refining image by decreasing both the blending factor and the kernel size in every rendering frame.

## 4.5 Results

In this section, the results obtained with splatting are compared with discrete and previous non-splatting continuous plots with respect to rendering performance and visual quality. First, the results of all of the three rendering techniques are compared with respect to their visual appearance. Then, some results obtained using resampling and progressive refinement are shown. The relation of splat size and sampling resolution is also investigated with respect to rendering performance and image coarseness. All measurements (including images) were produced with an implementation based on C++ and OpenGL Shading Language (GLSL). The implementation was tested on a Windows PC with Intel(R) Core(TM)




(a) (b) (c)

**Figure 4.2:** The "bucky ball" dataset rendered using three different approaches for scatterplots: (a) Discrete scatterplot, (b) splatted densities, and the (c) original continuous approach using projected tetrahedra. Density is represented by color, where black denotes low density and white denotes high density. Although the splatted and discrete plots use exactly the same samples, the splatted image provides a better approximation to a continuous density distribution. Please note that the plots are not supposed to be identical, as the original continuous approach uses a piecewise linear interpolation model on a tetrahedral grid, whereas the discrete and splatted versions are based on piecewise trilinear interpolation on a uniform grid. Nevertheless, the plots show an almost identical density distribution, where the prominent arc is an indicator for a material boundary.

2 Quad CPU running at 2.4 GHz with 4 GB RAM and an NVIDIA GeForce 8800 GTX graphics card.

While discrete and prior continuous plots need not to be parameterized, the splatting algorithm depends on the choice of the kernel size and the number of samples. Figure 4.2 compares a discrete scatterplot of the "bucky ball" dataset (a common test dataset in volume rendering representing a spherical fullerene) comprising $32 \times 32 \times 32$ data samples given on a uniform grid with the continuous versions rendered using the splatting approach and the projected tetrahedron algorithm (Bachthaler and Weiskopf, 2008). The variables are the original scalar value (horizontal axis) and its gradient magnitude (vertical axis). The arcs emerging in these types of plots can be used as an indicator for material boundaries (Kniss et al., 2002).

As discussed in Section 4.4, increasing splat size can be used to reduce the sampling resolution, while the level of coarseness of the resulting image is approximately maintained. The relationship between these parameters is illustrated in Figure 4.3. Here, every plot was rendered without progressive refinement but with different sampling resolutions and reconstruction kernel sizes. As expected, the images become smoother with increasing number of samples and with increasing splat size. Coarseness is approximately maintained

when doubling the number of samples with half of the kernel size and vice versa.

Figure 4.4 shows a performance analysis for different splat sizes and sampling rates, compared with traditional discrete scatterplots and continuous scatterplots using the original projected tetrahedra algorithm. From the measurement data, a linear dependency between sampling rate, splat size, and rendering performance can be concluded (note the $log_2$ scale of the $x$ axis). Although performance decreases with the number of splats, the progressive refinement algorithm introduced in the previous section can still be used to achieve interactive frame rates, as the total number of splats is divided into successive rendering frames. As a consequence, the algorithm scales well with dataset size and can easily be adopted for streaming data, as only a fixed number of samples is required for the individual rendering steps. This also includes large simulation data or time-dependent data, where visualization may be required in real-time. Finally, efficient rendering of statistical plots naturally facilitates the implementation of stacked displays or small multiples, where many instances of the same plotting technique are used to visualize different parts, projections, or subareas of the dataset. For example, the scatterplot matrix or the parallel-coordinates matrix (Chapter 6) may be extended using progressive refinement.

Using the derivation in Section 4.3.3, the progressive refinement algorithm can also be applied to parallel coordinates without change. The relation of splat size and sampling resolution regarding the coarseness of the resulting image remains the same, although smooth images are obtained with fewer samples due to the inherently stronger overdrawing of primitives in parallel coordinates. As an alternative, continuous parallel coordinates can be obtained from splatted continuous scatterplots using the gathering approach of Section 3.4.2 for every single frame.

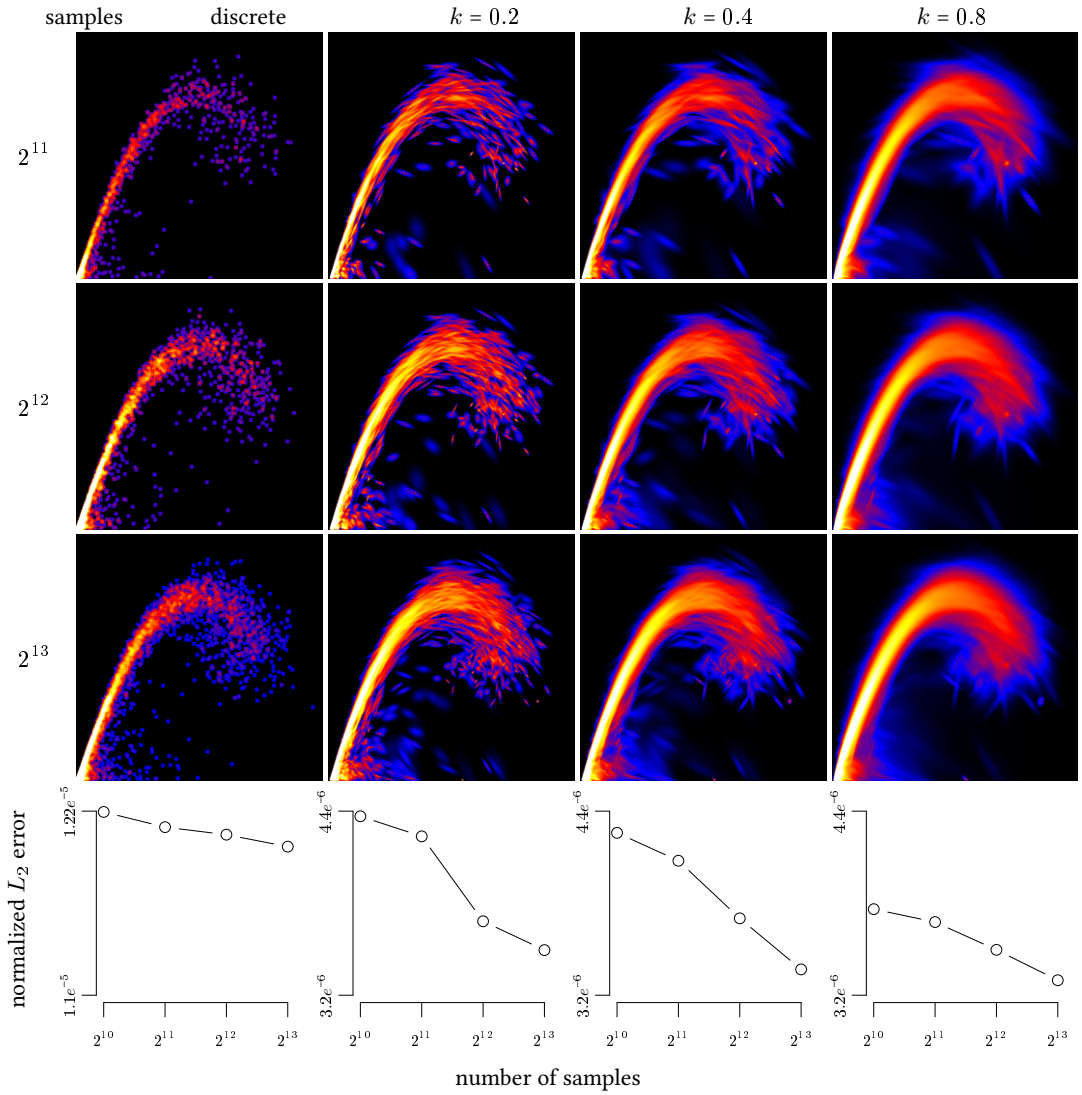



**Figure 4.3:** The effect of the sampling resolution and the kernel size parameter with respect to the overall coarseness of the plot. In the matrix shown above, the number of samples increases from top to bottom while the splat size increases from left to right. In both cases, the corresponding images become smoother. At the same time, however, increasing splat sizes result in less accurate plots, as can be seen in the rightmost column. Here, the blur introduced by larger splats makes the image appear "wider". The bottom row shows the $L_2$ error of densities from the splatted plots with respect to a traditional, discrete scatterplot rendered with $10^7$ samples of point-size four. While the convergence behavior is similar for all columns, the total error decreases with increasing smoothing factor $k$.

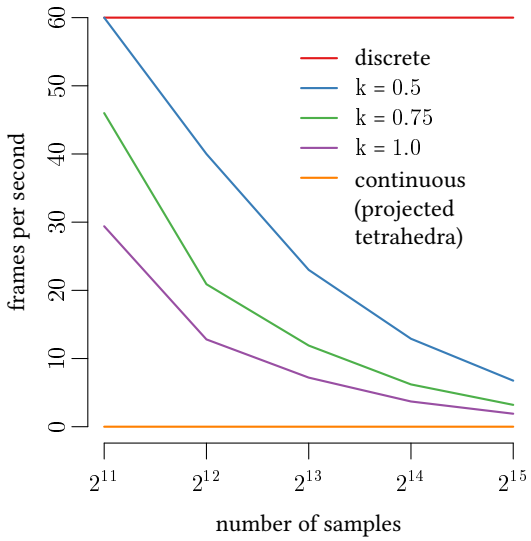



**Figure 4.4:** Rendering performance for splatted continuous scatterplots of the "bucky ball" dataset depending on splat size and the number of samples. Note that the performance of traditional discrete scatterplots rendered with point size one and of continuous scatterplots rendered with the projected tetrahedra algorithm are included. The traditional scatterplots are limited to 60 frames per second by the frame refresh rate of the test hardware. In contrast, the frame rate of continuous scatterplots lies well below one frame per second.

CHAPTER 5

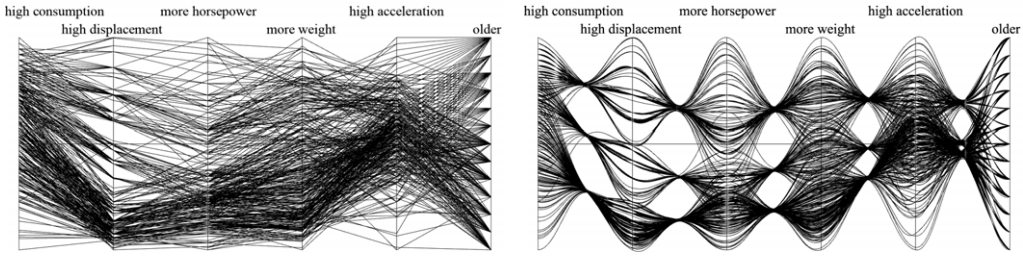
# Bundled Parallel Coordinates

## 5.1 Introduction

In this chapter[1], a variant combining the benefits of replacing polylines with smooth curves (Theisel, 2000; Moustafa and Wegman, 2002; Graham and Kennedy, 2003; Zhou et al., 2008; McDonnell and Mueller, 2008; Yuan et al., 2009; Holten and Wijk, 2010) and geometrically grouping curves of the same cluster (Holten, 2006; Zhou et al., 2008; McDonnell and Mueller, 2008) is introduced. A curve model based on piecewise cubic Bézier curves is presented that supports bundling at the cluster centroids. Given a clustering of the data, this method allows fast construction of the curve bundles while guaranteeing good visual continuation of the lines. In addition to parametrized smoothness of lines (Holten and Wijk, 2010), it also allows for tuning the bundling tightness, providing a range of representations from pure polylines (a special case of curve bundling with minimal smoothness and bundling) to tightly-bundled curve plots, enabling the user to obtain different views of the data.

The effectiveness of polylines and the presented curve bundling technique is compared with a controlled user study with respect to cluster perception

[1] **Parts of this chapter have been published in:** J. Heinrich, Y. Luo, A. E. Kirkpatrick, H. Zhang, and D. Weiskopf. Evaluation of a bundling technique for parallel coordinates. Technical Report Computer Science 2011/08, University of Stuttgart, Faculty of Computer Science, Electrical Engineering, and Information Technology, Germany, University of Stuttgart, Institute of Visualization and Interactive Systems, 2011.
J. Heinrich, Y. Luo, A. E. Kirkpatrick, 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*, pages 594–602, 2012.




(a) (b)

**Figure 5.1:** The "ASA cars" dataset (Ramos and Donoho, 1983) displayed as (a) polyline and (b) and bundled plots. Data are clustered by number of engine cylinders (4, 6, or 8). In the bundled plot, bundling was $\beta = 0.95$ and cluster centroids were plotted at their projected values on the bundle axis. The labels above each value axis indicate the interpretation of values closer to the axis top.

and correlation judgment. While variants of polylines and curves have been evaluated (Li et al., 2009; Holten and Wijk, 2010), no prior study evaluated the joint effect of these two features on the perception of clusters and correlations. See Table 5.1 for a summary of the evaluations.

The study showed that curve bundling maintains the users' ability to recognize correlation between data attributes, a traditional strength of parallel coordinates. Furthermore, for revealing clusters to the user, curve bundling is at least on par with color coding, the traditional way of representing clusters. Figure 5.1 compares the polyline version of parallel coordinates with a version using bundled curves.

The chapter begins with a summary of prior work on polyline parallel coordinates and variants using curves (Section 5.2). Section 5.3 describes the model of bundled curves and Section 5.4 presents the design of the user study and its results. This is followed by an extended example in Section 5.5, showing how bundled curves facilitate detailed analysis of the substructure of clusters in a seven-dimensional data set.

**Table 5.1:** Evaluations of Parallel Coordinates.

| | Correlation | Cluster Identification |
|---|---|---|
| Polylines | Li et al. (2010) | Holten and Wijk (2010) |
| Curves | Heinrich et al. (2011b, 2012a) | Holten and Wijk (2010) |
| Bundling | Heinrich et al. (2011b, 2012a) | Heinrich et al. (2011b, 2012a) |

## 5.2 Related Work

In this section, the work related to curve-based and bundled parallel-coordinates plots as well as evaluations of parallel coordinates is discussed with respect to the differences to the research presented in this section. For more information on these topics, please refer to Sections 2.3 and 2.4.

Two recent publications (McDonnell and Mueller, 2008; Zhou et al., 2008) enhance parallel-coordinates plots following the same perceptual motivation of geometric proximity as the method presented here. Zhou et al. (2008) deform traditional polylines by applying attracting and repelling forces. By construction, their method is based on proximity between the initial polylines and, thus, achieves an implicit, yet fixed type of clustering. Their method emphasizes the proximity of the polylines, rather than showing externally provided clusters. Moreover, their visual clustering is based on a piecewise model: the vicinity of polylines between two neighboring data axes (or dimensions) of the parallel-coordinates plot governs the visual clustering between those two data dimensions; other pairs of neighboring data dimensions are clustered independently. Therefore, multivariate data is not clustered on a per-data-point level, but based on pairs of variables. The resulting visual clustering is thus sensitive to the order of axes in the parallel-coordinates plot.

Holten (2006) introduced edge bundling of tree layouts. McDonnell and Mueller (2008) built on this idea, developing a geometric, spline-based approach to computing visual bundling that is similar to the one presented here. However, McDonnell and Mueller's technique has a different objective: it targets illustrative parallel-coordinates plots, using visual simplification and non-photorealistic rendering techniques such as silhouette lines, halos, and shadows. Therefore, details of the internal structure of data points within clusters are not a focus of their research. Moreover, cluster membership information is still based on color coding, whereas this approach provides a complementary, geometry-based visualization of clusters.

The method described in this chapter improves upon the proximity-based parallel-coordinates techniques of McDonnell and Mueller (2008) and Zhou et al. (2008) in the following ways. First, it makes better use of the available screen space by re-distributing visually clustered curves in a uniform way. Therefore, there is much less overlap in the important parts of the plots—in the regions between two data axes, where users identify correlation of data points. In addition, overdraw and cluttering issues are reduced by this redistribution. Second, $C^1$ continuity of the curved lines is guaranteed across data axes, addressing the cross-over problem. A review on evaluations of parallel-coordinate visualizations was given in Section 2.4. While Li et al. (2009) examined the visualization of correlation for linear parallel coordinates and Holten and Wijk (2010) the visualization of clusters, the user study conducted for this work aims at evaluating the impact of bundling and curves to the judgment of correlation

and the detection of clusters (see also Table 5.1). Finally, McGee and Dingliana (2012) showed that bundling negatively impacts the viewer's ability to trace paths and recognize cluster connectivity in nodel-link diagrams with respect to time and accuracy. While we share the bundling approach, parallel coordinates employ a different layout[2] than the one used by McGee and Dingliana and do not contain edges within clusters. In contrast to their study, we investigate a different set of tasks for bundled parallel coordinates.

## 5.3 Curve Model and Curve Bundling

In this section, the curve models presented in Section 2.3.1 are extended to support bundling curves with the same cluster membership. The extension is required because previous bundling techniques for parallel coordinates either use a different curve model (McDonnell and Mueller, 2008) or a different clustering model (Zhou et al., 2008).

The bundled curve model uses curve geometry to increase the visibility of structure in the data across multiple axes, reveal structure within clusters, and alleviate the line-tracing problem. At the same time, curve control points and tangents are chosen to achieve good approximation to the original polylines. The construction is designed to maintain the desirable characteristics of polyline plots, in particular, their ability to reveal correlations between variables. Finally, the parameters of the model support exploration of the data by adjusting its representation.

### 5.3.1 Overview

Given an input of $N$-dimensional data points with specified cluster membership, we will refer to axes $\overline{X}_1, \overline{X}_2, ..., \overline{X}_N$ in the parallel-coordinates domain as the *value axes* to distinguish them from the additional axes created for curve modeling and bundling. Let us assume (without loss of generality) that the value axes are uniformly distributed and separated by unit intervals: $\boldsymbol{d_N} = (0, 1, ..., N-1)$.

Using the interpolation model introduced in Section 2.2.3, the polyline corresponding to a data point $P = (p_1, p_2, \ldots, p_N)$ is replaced by a piecewise cubic Bézier curve with the following properties:

- The curve interpolates $\overline{p}_i = (d_i, p_i);\ i = 1 \ldots N$ at the value axes.
- The curve is $C^1$ continuous throughout.

[2] Assuming that a parallel-coordinates plot composed of lines can be modeled as a graph where indexed points $\overline{p}_i$ represent vertices and lines represent edges.

- Curves corresponding to data points that belong to the same cluster are bundled between adjacent value axes. This is accomplished by inserting a *bundle axis* midway between the value axes and by appropriately positioning the Bézier control points.
  - To support curve bundling, curves within a given cluster are adjusted by moving their control point on the bundle axis toward the value of their cluster centroid on the bundle axis.
  - The cluster centroid is the projection of the $N$-dimensional centroid on the plane defined by the respective adjacent value axes, intersected with the bundle axis.
  - The use of curves also takes better advantage of the entire plot area. Specifically, the cluster centroids can be arbitrarily distributed along the bundle axis to alleviate the line clutter problem.
- Two parameters, $\alpha$ and $\beta$, adjust the shape of the Bézier curves. The parameter $\alpha$ controls the extent the curve approximates the original piecewise linear polyline, while retaining $C^1$ continuity; we call $\alpha$ the *smoothness scale*. The parameter $\beta$ dictates the *bundling strength*, how tightly the curves within a cluster are pulled together. Note that polyline-based parallel-coordinates plots are a special case of this model, when setting $\alpha = \beta = 0$. Other curve models are obtained by setting $\beta = 0$.

### 5.3.2 Curve Continuity and Smoothness Scale

Consider two adjacent value axes $\overline{X}_i$ and $\overline{X}_{i+1}$ with points $\overline{p}_i$ and $\overline{p}_{i+1}$ on them, respectively. Let the intersection of the line segment $\overline{p}_i\overline{p}_{i+1}$ with the bundle axis $\overline{V}_i$ (halfway between $\overline{X}_i$ and $\overline{X}_{i+1}$) be $\overline{q}_i$ (Figure 5.2). We shall convert the straight line segment $\overline{p}_i\overline{p}_{i+1}$ into two cubic Bézier curve segments: $b_{i,1}$ between $\overline{X}_i$ and $\overline{V}_i$ and $b_{i,2}$ between $\overline{V}_i$ and $\overline{X}_{i+1}$. Due to curve bundling, we may move $\overline{q}_i$ to $\overline{q}'_i$ along $\overline{V}_i$ as a result of attraction by a cluster centroid $\overline{c}_i$; this is explained in Section 5.3.3. The new point $\overline{q}'_i$ will serve as a common control point for the two Bézier curve pieces $b_{i,1}$ and $b_{i,2}$, which are constructed as follows.

We describe the construction of the four control points for $b_{i,1}$; $b_{i,2}$ is symmetric. The first and last control points of $b_{i,1}$ are $\overline{p}_i$ and $\overline{q}'_i$, respectively. To ensure $C^1$ continuity of $b_{i,1}$ and its left neighbor, $b_{i-1,2}$, we force them to share the same tangent, one that makes the same angle with the vector $\overline{\boldsymbol{p}}_{\boldsymbol{i-1}}\overline{\boldsymbol{p}}_{\boldsymbol{i}}$ and the vector $\overline{\boldsymbol{p}}_{\boldsymbol{i}}\overline{\boldsymbol{p}}_{\boldsymbol{i+1}}$. In other words, the curve's tangent line $l_i$ at $\overline{p}_i$ bisects the angle formed by $\overline{\boldsymbol{p}}_{\boldsymbol{i}}\overline{\boldsymbol{p}}_{\boldsymbol{i+1}}$ and the extension of $\overline{\boldsymbol{p}}_{\boldsymbol{i-1}}\overline{\boldsymbol{p}}_{\boldsymbol{i}}$ beyond $\overline{p}_i$; the angle bisector provides a best approximation of the two polyline directions $\overline{\boldsymbol{p}}_{\boldsymbol{i-1}}\overline{\boldsymbol{p}}_{\boldsymbol{i}}$ and $\overline{\boldsymbol{p}}_{\boldsymbol{i}}\overline{\boldsymbol{p}}_{\boldsymbol{i+1}}$.

The second control point for $b_{i,1}$ lies at the intersection of $l_i$ and a *secondary control axis* $\overline{Y}_{i,1}$. The axis $\overline{Y}_{i,1}$ is located at distance $0 \le \alpha < 0.25$ from $\overline{X}_i$, between $\overline{X}_i$ and $\overline{V}_i$. The third control point lies at the intersection between

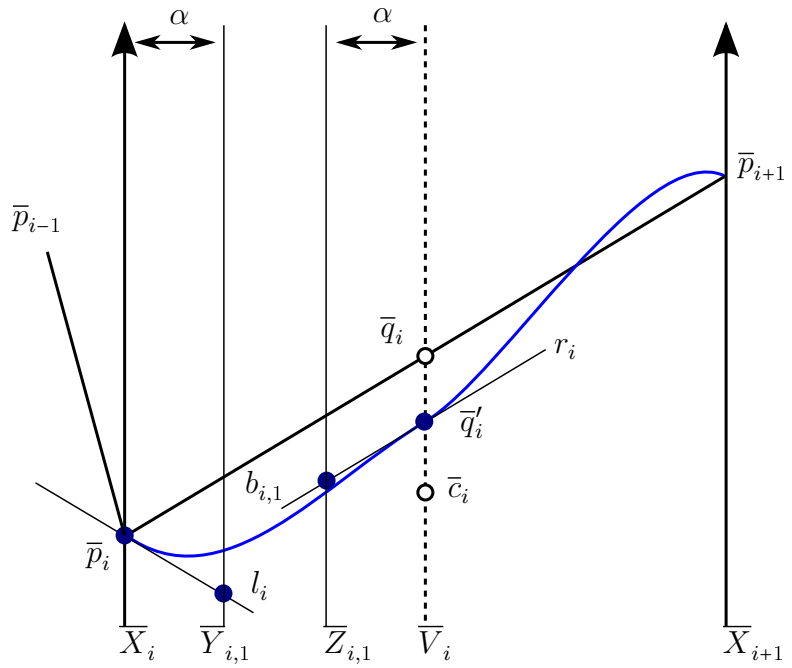
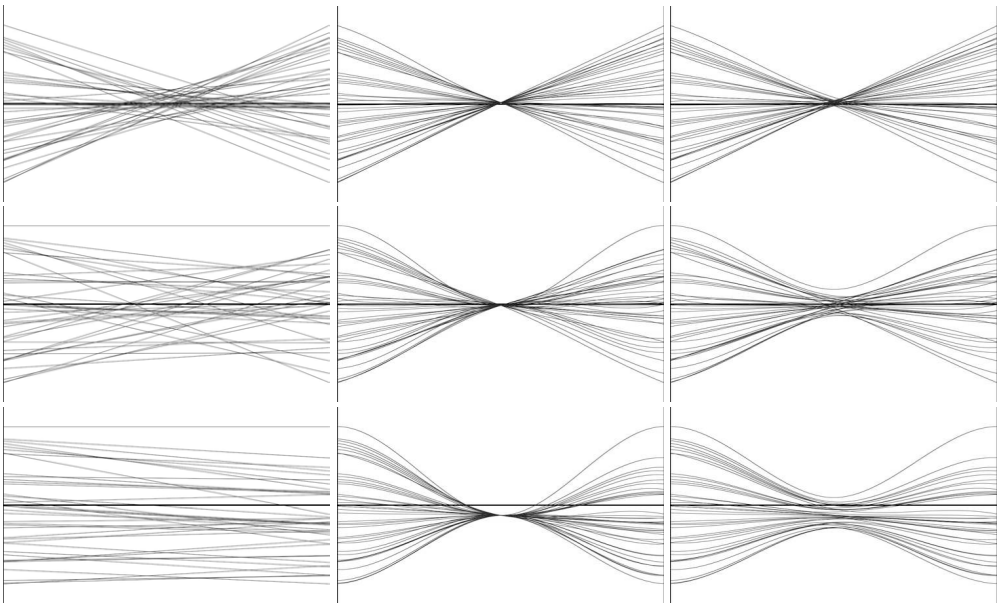



**Figure 5.2:** Construction of Bézier curve pieces between two adjacent value axes $\overline{X}_i$ and $\overline{X}_{i+1}$. We insert a bundle axis $\overline{V}_i$ midway between $\overline{X}_i$ and $\overline{X}_{i+1}$, as well as secondary control axes, $\overline{Y}_{i,1}$,$\overline{Z}_{i,1}$, placed at a distance $\alpha$ away from their corresponding value or bundle axis. Adjacent Bézier pieces share the same tangent line and intersections between the tangent lines, while the value, bundle, or secondary control axes define the control points for the Bézier curves. The control points for $b_{i,1}$ are shown as blue dots. The control points for $b_{i,2}$ (not shown) would be at corresponding locations on the right. The point $\overline{c}_i$ is a cluster centroid which attracts the constructed Bézier curves in curve bundling. As a result, while the original polyline passes through $\overline{q}_i$, the curve pieces now pass through $\overline{q}'_i$.

a new tangent, $r_i$, and an additional secondary control axis $\overline{Z}_{i,1}$. $r_i$ is chosen as the direction formed by the vector $\overline{\boldsymbol{p}}_{\boldsymbol{i}}\overline{\boldsymbol{p}}_{\boldsymbol{i+1}}$. We force the two pieces $b_{i,1}$ and $b_{i,2}$ to share $r_i$ as a common tangent at $\overline{q}'_i$. The secondary control axis $\overline{Z}_{i,1}$ lies between $\overline{X}_i$ and $\overline{V}_i$, and is at a distance $\alpha$ from $\overline{V}_i$. If either $\overline{X}_i$ or $\overline{X}_{i+1}$ is a boundary axis, the tangent at that boundary axis can be set as the bisector of the angle formed by the horizontal axis and the line segment involved, which is either $\overline{p}_1\overline{p}_2$ or $\overline{p}_{N-1}\overline{p}_N$.

The same control points can be obtained in the method of Holten and Wijk (2010) by setting $\overline{q}_i = p$, $\overline{p}_i = p_0$, and $\overline{p}_{i+1} = p_1$.



(a) (b) (c)

**Figure 5.3:** Effect of bundling strength $\beta$ on representing correlations. Column (a) shows polyline plots of two variables with strong negative correlation, no correlation, and strong positive correlation, respectively. Column (b) shows the corresponding bundled curve plots with strict bundling, $\beta = 1$. Column (c) shows bundled curve plots with $\beta = 0.8$.

### 5.3.3 Curve Bundling and Bundling Strength

The model can be extended to represent cluster membership using bundling. Consider the set of polylines belonging to a particular cluster $\mathcal{C}$. For each such polyline, we record its intersections with all the bundle axes. For a particular bundle axis $\overline{V}_i$, we can use the centroid $\overline{c}_i$ of the intersection points corresponding to the polylines in cluster $\mathcal{C}$ as the common control point shared by all the Bézier pieces (those which replace polylines in cluster $\mathcal{C}$) adjacent to $\overline{V}_i$. This will force all the Bézier curves in the cluster to pass through the common point $\overline{c}_i$.

Strict curve bundling may hinder the viewer's ability to distinguish a positive correlation from a negative one. This is illustrated in Figure 5.3, where the first two columns show the polyline plots (Figure 5.3a) and strictly bundled plots (Figure 5.3b) on two variables with strong positive correlation, no correlation, and strong negative correlation, respectively. It is still possible to detect the minor differences in the bundled plots, but not as easy as for the polyline plots.

As a remedy, the bundling strength parameter $\beta$ is introduced. As shown in Figure 5.2, let the intersection between a straight line segment and the bundle axis $\overline{V}_i$ be $\overline{q}_i$. The bundling strength controls the extent to which $\overline{q}_i$ will be

pulled or attracted toward the centroid $\overline{c}_i$ defined above. With a linear weighting scheme, the new $\overline{q}'_i$ will be given by

$$\overline{q}'_i = (1 - \beta)\overline{q}_i + \beta\overline{c}_i,$$

where $0 \leq \beta \leq 1$. The choice $\beta = 1$ corresponds to strict curve bundling, whereas $\beta = 0$ disables bundling altogether.

In the current implementation, the same $\beta$ is applied to all instances of curve bundling. Obviously, one can gain more fine-grained control by tuning $\beta$ individually for specific curves, bundle axes, or clusters; this would be at the expense of introducing a more complex control interface.

By adjusting the bundling strength properly, correlations can be detected in bundled curve plots just as easily as in polyline plots, even in the presence of clear curve bundling. As shown in Figure 5.3c, with $\beta = 0.8$ more negatively correlated data form denser, narrower crossing bands. A quantitative, empirical comparison of the effectiveness of bundled curve and polyline for discovering variable correlations is given in the next section.

### 5.3.4 Redistribution of Cluster Centroids

To take advantage of the entire available plot area and further alleviate the line cluttering problem, the cluster centroids can be sorted along each bundle axis and then redistributed *uniformly* across the length of the bundle axis in an *order-preserving* manner, where order is separately defined by the centroid for every adjacent pair of axes. Figure 5.5 shows an example of redistributed centroids. An immediate extension to uniform distribution is to take into account the sizes of the clusters and adjust the positioning of the cluster centroids accordingly.

A given analysis may or may not benefit from distributing the centroids. Cluttered plots will have their clutter reduced through distribution. However, for plots with greater separation between clusters, plotting the centroids at their projected values can provide useful information for the analyst. Both methods are used here: the user studies and most example plots used distributed clusters, while the example application in Section 5.5 does not use distributed clusters.

## 5.4 User Study

To compare the effectiveness of polylines and bundled curves, a user study was performed. Observers were asked to estimate (a) correlations and (b) the number of clusters, in datasets represented using polylines and bundled curves. It was expected that bundled curves would support correlation estimation at least as well as polylines do, and that bundled curves would support superior estimation of the number of clusters.

### 5.4.1 Overview

For the experiment, the performance of analysts skilled both in an underlying domain and at interpreting parallel coordinates of a given type (polyline or bundled curves) had to be estimated. Such users are not merely difficult to find, for the case of bundled curves they do not yet exist. This constraint was addressed with an approach often used for visualization user studies:

1. Participants who had little to no experience with either form of parallel coordinates were recruited. The participants were given a short tutorial on strategies for estimating correlations in parallel-coordinates plots of each style. This created a pool of participants equally skilled at reading both styles, somewhere between novice and intermediate skill.

2. Data sets generated solely according to specified probability distributions were used, with no underlying semantics. This ensured that no participant would be able to apply domain knowledge to interpret the plots.

Accuracy was used as the sole dependent measure, time was not recorded. The argument is that this untimed task matches the context in which data analysts typically use parallel coordinates, taking enough time to consider their data in depth. This choice emphasized that participants take as long as necessary to make their best estimate. It also minimized fatigue by allowing participants to rest whenever they wished, without regard for their score. This choice also eliminated the potential confound of different participants adopting different speed–accuracy trade-offs, because accuracy was uniformly emphasized.

The curve styles were compared for two tasks, estimating correlation and estimating number of clusters. The tasks were performed in a fixed order for every participant, with participants estimating correlation first. This design permits more direct interpretation of the results because all participants performed each task with a fixed level of prior experience. In particular, their experience reading plots in the correlation task would carry over to enhance their performance in the cluster estimation task.

By contrast, a design that counterbalanced task order would have split participants' prior experience, increasing the variance and making the results harder to interpret. Given that a counterbalanced design would only protect against the case that doing the correlation estimate first would *differentially* advantage one curve style, a prospect that can be considered highly unlikely, a fixed task order was chosen for its more straightforward interpretation.

### 5.4.2 Design

The study design was single-factor, two-level, and within-subjects. Observers viewed two data series. The first was always the correlation estimation series,

the second the cluster estimation series. Within each series, line style was a blocked factor, with all trials performed first in one line style, then the other. Order of the two line styles was counterbalanced across participants, with participants randomly assigned to the order. Dependent measures, computed separately for each series, were the Pearson correlation $r$ between the actual dataset correlation and the correlation estimated by participants, and the Fleiss $\kappa$ measure of agreement amongst participants.

Before running the full study, a pilot study with five participants was conducted to determine the best values of bundling strength $\beta$ and smoothness scale $\alpha$. The values $\beta = 0.8$ and $\alpha = 0.16$ achieved the best balance of correlation detection and cluster visualization. These values were used for the bundled plots in both series of trials.

### Participants

A convenience sample of 14 participants (9 men, 5 women, ages 23–37) was recruited from graduate students in computing science and engineering science at Simon Fraser University, British Columbia, Canada. Of these 14, 2 had previously used polyline parallel coordinates, 8 had experience with some form of information visualization but had never used parallel coordinates, and 4 had never used any visualization software. Volunteers were paid CDN$ 20.

### Procedure for the Session

Participants first answered a brief series of questions assessing their level of experience with information visualization and computers in general. They were next tested for any color deficiencies using a web-based test[3]. All 14 participants had acceptable color vision. They next read a tutorial on the basic principles of parallel coordinates and their instantiation in polylines and bundled curves. The tutorial defined correlation and gave examples of positive and negative correlation using both line types.

Participants then began the first series of trials, in which they estimated correlations. After completing that series, participants began the second series, in which they estimated the number of clusters in plots. After completing the second series, they indicated which line style they preferred and answered a short list of open-ended questions about their experience during the study.

Participants were allowed to take as long as they wished on each trial. Total time to complete the session varied widely, from 50 to 110 minutes. Most participants completed the study in less than 90 minutes.

[3] http://vision.healthcommunities.com/HealthProfiler/healthpro_cb.shtml

### 5.4.3 First Series: Estimating Correlations

In the correlation estimation trials, participants viewed a series of datasets in 2D parallel coordinates, plotted in either polylines or bundled curves. For each plot, the user was asked to categorize the correlation as "strong negative correlation", "negative correlation", "no correlation", "positive correlation", or "strong positive correlation".

#### Procedure

Before starting each line style, participants read a short tutorial on estimating correlations in that style. For polylines, the tutorial suggested looking for whether the lines crossed or not, the distribution of line crossings (whether only in the middle or distributed throughout the range), and the overall shape of the plot. For bundled curves, the tutorial suggested looking at the width of the middle band and the overall shape of the plot. For each style, the tutorial presented example plots of all seven degrees of correlation. To map the seven values of actual correlation to the five categories of user response, the tutorial recommended reporting $z$ values of $-1.0$ and $-0.5$ as both "negatively correlated", and similarly for $+0.5$ and $+1.0$.

Participants began each line style with a training session. The training session presented one plot of each correlation level in the given style. Participants estimated the correlation and were then told which answers would have been appropriate for the dataset. Since there were seven levels of correlation but only five levels of user response, two possible answers were suggested for every example. After estimating all seven practice correlations, a page was displayed reminding participants of the strategies for estimating correlation for this plot style. Users pressed a button to start the first experimental trial.

Experimental trials had the same interface as the practice trials, but provided no feedback about the actual correlation. When the participant was satisfied with their estimate for the current trial, they pressed a button to start the next.

#### Trial Data

Three groups of seven datasets were generated, each with $n = 40$ data pairs. The pairs were generated from normally distributed random series $x$ and $y$, selected to ensure that each set of 40 pairs had the given correlation coefficient. Each group of datasets had exactly one set for each level

$$z = -1.5, -1.0, -0.5, 0.0, +0.5, +1.0, +1.5,$$

where $z$ is the Fisher transform of the correlation. These were the same levels of correlation used in prior work (Li et al., 2009). One group of datasets was always used for the training phase. The remaining 14 datasets were each used

twice, once for each line style. Within each line style, order of datasets varied randomly for each participant.

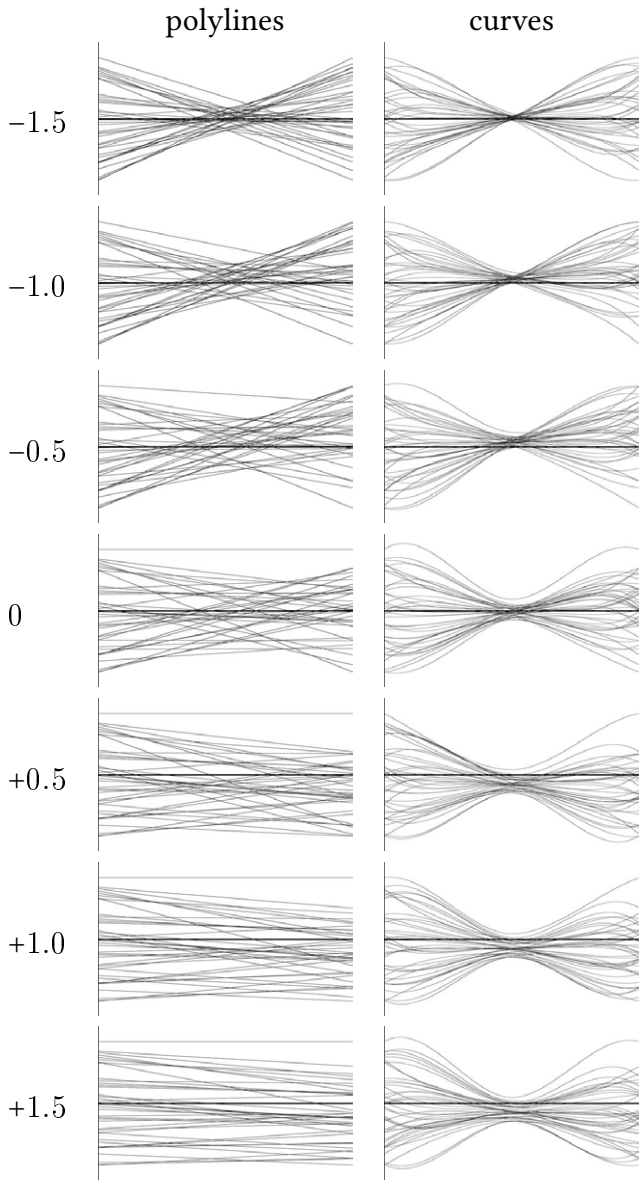



**Figure 5.4:** One of the three sets of plots used in the correlation estimation series. The correlation, specified in Fisher $z$, is shown left of each plot.

The bundled curve representation required two additional parameters for each data point: the directions of the line leaving each axis. These are not required for polyline plots, where the direction of the line leaving an axis is independent of the direction the line entered that axis from the other side. However, the exit direction of bundled curves is affected by the direction of the line entering from the other side, due to the $C^1$ continuity requirement. Pilot tests showed that if

all curves entered the axes at a constant horizontal direction, observers used the consistent bending of curves at the axes as a cue to estimate correlation. Since this cue would not occur in actual use of bundled curves, which would in fact enter their axes at varying angles, the direction at which each curve entered each axis was randomly perturbed. This random perturbation likely made correlation detection slightly more difficult for bundled curves than it would be in practice, where entry to the axes would vary but not be random.

Figure 5.4 illustrates example datasets with all seven different correlation coefficients used for the experimental trials. The left column shows polyline plots and the right column shows bundled curve plots.

### 5.4.4 Second Series: Estimating Clusters

In the cluster estimation trials, participants viewed a series of clustered datasets in parallel coordinates ranging from two to six dimensions, plotted in either polylines or bundled curves. Clustering was indicated by color (for polyline plots) and bundling (for bundled curve plots). Color Brewer (Harrower and Brewer, 2003) was used to define effective color maps for the polyline plots. For each plot, the user was asked to estimate the number of clusters.

#### Procedure

Before starting the series, participants read a description of how clusters are represented in both line styles. They then began working with either polyline plots or bundled curve plots, depending upon which order had been assigned. They practiced estimating the number of clusters in three trial plots, with five, three, and eight clusters. Figure 5.5 shows typical examples of such plots. After each training trial, the correct number of clusters was reported. After the three training trials, a page redisplayed the three datasets and the number of clusters in each. Users pressed a button to start the first experimental trial.

Experimental trials had the same interface as the training trials, but provided no feedback about the actual number of clusters. After entering their estimate for the clusters, participants pressed a button to move on to the next trial. Once they had completed a series in one line style, they did the training and experimental trials for the next style.

#### Trial data

Trial datasets were created from three real-world and three synthetic datasets (Table 5.2). The real-world datasets are popular test datasets, taken from the Xmdv Web page[4]. The synthetic datasets were generated by sampling normally

[4] http://http://davis.wpi.edu/xmdv/

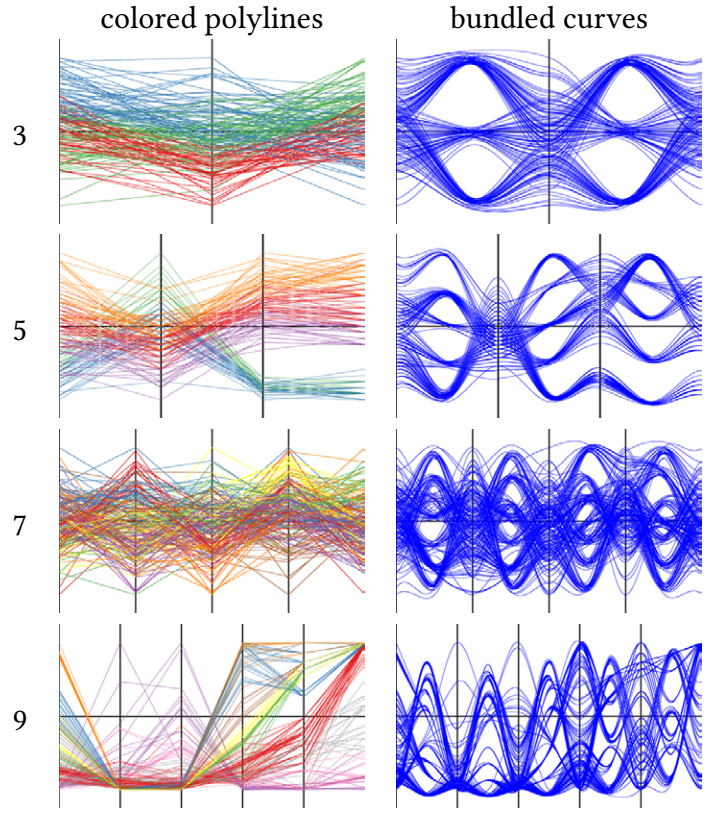



**Figure 5.5:** Representative plots used in the cluster estimation series: polyline plots with color coding of clusters (left column), and bundled curve plots (right column). From top to bottom, the data sets are g160 (number of $k$-means clusters $k = 3$), iris ($k = 5$), g200 ($k = 7$), and netperf ($k = 9$).

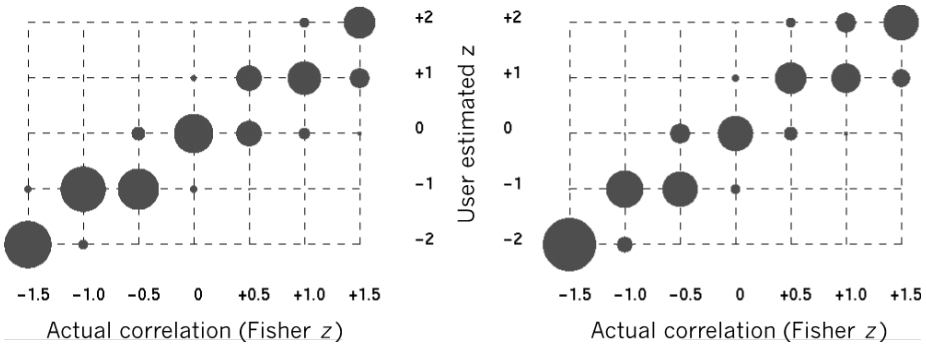
distributed series, selected to ensure the required correlation across each dimension. Each of the 6 datasets was then clustered by the $k$-means technique into $k = 3, 5, 7$, and 9 clusters. This series of 24 datasets was plotted using both line styles. Within each series, the order of trials varied randomly for each user.

### 5.4.5 Results

Figure 5.6 shows the distribution of participants' responses for the correlation estimation series. There was a strong linear correlation between participants' estimates and actual correlation for polylines and bundled curves (both $r = 0.90$). Considering the estimates for positive and negative correlations separately, estimates for negative correlations were stronger ($r = 0.75$ for polylines, $r = 0.79$ for bundled curves, difference of the equivalent $z$-scores $\Delta z = 0.10$) than for positive correlations ($r = 0.55$ for polylines, $r = 0.39$ for bundled curves, $\Delta z = -0.21$). Agreement amongst participants was moderate ($\kappa = 0.43$ for

**Table 5.2:** Datasets for the cluster estimation series ($N$ is the number of dimensions, $n$ is the number of data points)

| Name | $N$ | $n$ | Source |
|---|---|---|---|
| iris | 4 | 150 | Botany |
| netperf | 6 | 179 | Computer Science |
| htong | 4 | 365 | Earth Science |
| g40 | 2 | 40 | Synthetic |
| g160 | 3 | 160 | Synthetic |
| g200 | 5 | 200 | Synthetic |




(a) Polyline plots (b) Curve plots

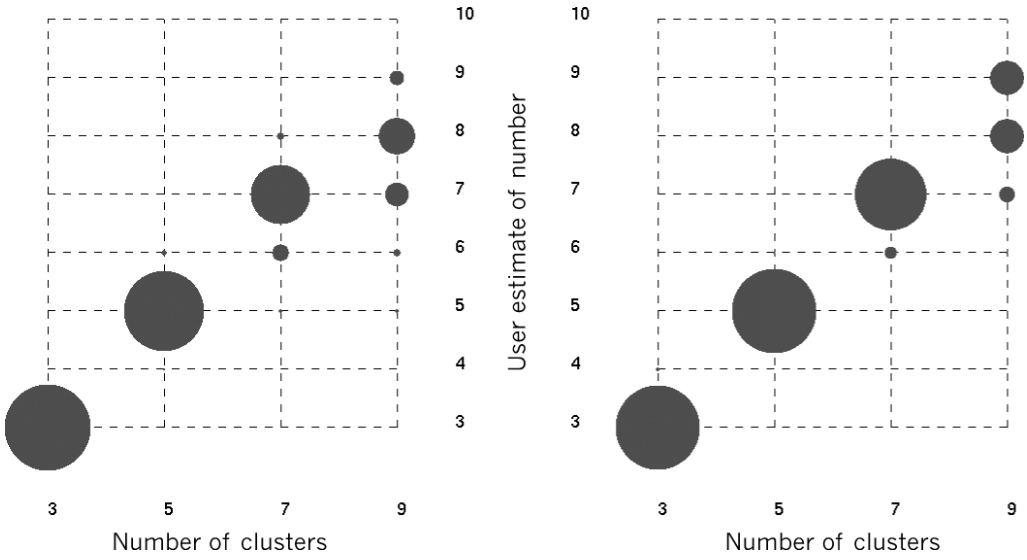
**Figure 5.6:** Distribution of responses for the estimated correlations of 2D parallel coordinates for polyline plots and bundled curve plots. Circle radius represents the frequency with which participants estimated a correlation strength for each actual correlation.

polylines, $\kappa = 0.41$ for bundled curves). The results for polylines are comparable to those of Li et al. (2009).

For all comparisons, the one-sided width of a 95% confidence interval, which is entirely determined by the sample size of 14, is $\Delta z_{95\%} = 0.59$. All the $\Delta z$ scores presented above were substantially within this bound, indicating that none of the differences was statistically significant.

Figure 5.7 shows the distribution of participants' responses for the cluster estimation series. The overall correlation is strong for both line styles ($r = 0.92$ for polylines, $r = 0.96$ for bundled curves, $\Delta z = 0.36$). The correlations were much stronger for datasets with three or five clusters ($r = 0.98$ for both line styles) than those with seven or nine clusters ($r = 0.41$ for polylines, $r = 0.68$ for bundled curves, $\Delta z = 0.39$). As with the correlation estimation series, all $\Delta z$ values were substantially below $0.59$, indicating that none of the differences was statistically significant.

Agreement amongst participants for cluster estimation was slightly higher for




(a) Colored polyline plots

(b) Bundled curve plots

**Figure 5.7:** Distribution of responses for the estimated number of clusters in parallel-coordinates plots in two line styles. Circle radius represents the frequency with which participants estimated a dataset to have a given number of clusters.

bundled curves ($\kappa = 0.65$) than for polylines ($\kappa = 0.56$). Each line style had higher agreement than their corresponding levels for correlation estimation.

### 5.4.6 Discussion

The results for the two series of plots demonstrate important strengths of the bundled curve representation. The correlation estimation series demonstrates that correlation is as readily recognizable when parallel coordinates are rendered in bundled curves as when rendered in polylines. This result is not obvious. Polyline plots provide a clear focal point for estimating correlations: the width of the center region is an excellent indicator of correlation, with strong negative correlations producing a narrow center region and strong positive correlations producing a wide center region. In contrast, bundled curve plots by definition draw the curves into one or more narrow center regions. The width of those regions is only mildly determined by the correlation of the dataset. Yet bundled curves nonetheless provided sufficient cues (width of center region, shape of lines) that participants could estimate correlation from bundled curves as readily as from polylines.

The cluster counting series demonstrates that viewers could identify clusters with the respective bundles. This is not surprising, as bundling provides a strong cue of cluster identity. Participants likely determined cluster membership by looking at the bundle axes, where bundling has its strongest effect. In effect, a

bundled curve plot uses different regions to geometrically represent the spread and the clustering of the dataset. The spread of values for a cluster is represented at the value axis. The cluster identity of a datum is represented at the bundle axis. In contrast, polylines provide no geometric representation of cluster membership, so it must be represented using a different cue, color. Whereas polylines provide only correlation information in the inter-axis regions, bundled curves use that region to display correlation, number of clusters, and cluster membership—a much more effective use of the space.

The geometric representation of cluster and distribution must be simultaneous if the analyst is to compare the distributions of the different clusters. The bundling and $C^1$ continuity of bundled curves are essential for this comparison to occur, for these features allow the viewer to be aware of both clusters and distribution simultaneously. Bundling exploits the Gestalt principle of proximity, visually grouping the lines of a cluster in the middle of the plot. $C^1$ continuity exploits the Gestalt principle of continuity to maintain this visual grouping on the value axes, where the distribution is represented. This allows the viewer to compare the distributions of different clusters. As a secondary benefit, the $C^1$ continuity allows this cluster identification to be maintained across value axes, the membership reinforced at each bundle axis.

The same bundling strength was used for both the correlation and the cluster counting series. This demonstrates that each task can be achieved without sacrificing the other.

## 5.5 Application

The user study demonstrated that bundling is sufficient for observers to distinguish clusters. This section presents an example demonstrating that a bundled parallel plot can represent the influence of clusters more directly than its polyline counterpart. We assume a use case where a clustering has been found and the analyst wishes to explore an initial hypothesis of the predictive power of cluster membership.

The car performance data ("ASA cars") of Ramos and Donoho (1983) is used, a sample data set for the 1983 American Statistical Association (ASA) Data Exposition. Altogether the original data set includes eight fields, of which relationships amongst seven are analyzed: the number of cylinders, the miles per gallon, the displacement, the horsepower, the weight, the number of seconds to complete a quarter-mile from a standing start, and the model year. A total of 406 cars are in the dataset.

A natural basis for clustering this data is the number of cylinders in each model's engine. Clustering the data this way produces five clusters. Two clusters, the seven cars with either three or five cylinders, are so small that they increased the plot complexity with no gain in explanatory power, so they were deleted

from the plot. The remaining 399 cars were grouped into clusters for four, six, or eight cylinders. A reasonable initial hypothesis is that the number of cylinders has the following correlations to the other fields:

- More cylinders should reduce the miles per gallon (negative correlation).
- More cylinders should increase displacement (positive correlation).
- More cylinders should increase horsepower (positive correlation).
- More cylinders should increase weight (positive correlation).
- More cylinders should decrease the number of seconds to accelerate over a quarter mile (negative correlation).

For model year, there was no strong *a priori* assumption for any correlation with the other fields. Older cars were plotted at the top, as there was a mild likelihood that they would be less efficient than newer models.

Figure 5.1 shows the data plotted in both polyline and bundled parallel coordinates. The direction of each value axis is selected so that the values hypothesized to correlate most directly to the maximum number of cylinders are displayed at the top (value axes with expected negative correlations have their smaller values at the top). At each bundle axis, curves within a cluster are pulled to the projections of their centroids.

Comparing the between-axes region in the two plots, the polyline version (Figure 5.1a) features a lot of geometry that engages the eye but provides no useful information. Between every pair of value axes many lines overlap and at each axis there are many abrupt changes of direction.

By contrast, in the between-axes region of the bundled version (Figure 5.1b) the geometry is informative. The projection of each cluster's centroid is indicated on the bundle axes. It is possible to immediately assess both the relative order and the magnitude of the difference between the projected centroids by number of cylinders.

Comparing the two plots at the value axes, for the polyline plot it is impossible to compare the ranges of different clusters for most variables because it is too difficult to associate a line segment with its cluster. By contrast, in the bundled plot it is easy to trace an individual curve to its cluster bundle, making it easy to compare the cluster ranges on a given axis. This includes detecting outliers within clusters, such as the most powerful car in terms of horse power within the six cylinder group. From the polyline plot, the correspondence to its cluster could not be determined without color or additional axes. Consequently, a glance at a value axis is sufficient to determine the predictive power of the clustering on that variable. For the given cars, the number of cylinders strongly predicts displacement, horsepower, and weight. For consumption and acceleration however, the number of cylinders only predicts the extreme values for each cluster.

The polyline plot is slightly more informative about the relationship between model year and acceleration (simply because they were plotted as adjacent axes). Older cars have a wider range of acceleration than more recent cars, with the quickest cars all coming from the earliest years. The bundled plot obscures the relationship of model year with any other axis because the bundle points make it impossible to track the model year to any adjacent axis.

Overall, the bundled representation is clearer because it separates distinct components of the data into distinct regions of the plot: Cluster membership is displayed on the bundle axes, while the range of individual points within the cluster is displayed on the value axes. The curves allow straightforward correlation of individual values and their membership. By contrast, the polyline representation gives equal prominence to changes within clusters and changes of the clusters as a group, making it impossible to separate them visually.

This example use case assumed that the analyst had an *a priori* hypothesis of the correlations between cluster membership and other variables. This is a common and important case because it represents the goal of many other use cases. For example, where there is no prior hypothesis, interactive bundled plots would allow the analyst to consider alternative relationships between the clusters and other variables by varying the direction and order of axes, together with the bundling parameters $\alpha$ and $\beta$. The clear geometric separation afforded by bundling would likely allow the analyst to converge on the best explanation far more quickly than with polyline plots.

### 5.5.1 Bundles Aid Interpreting Clusters

The purpose of bundling is to highlight the cluster membership of individual points. Bundling is not itself a technique for determining clusters. Rather, it is used after the analyst has already derived clusters, whether from *a priori* domain knowledge or statistical clustering methods. Bundling aids in the interpretation of clusters in several ways.

At the local level of analysis of a single axis, bundling makes it easier to see the spread of values for one cluster and to compare spreads for several clusters. In the car example above, much of the analysis of the influence of engine cylinders was done this way. Bundling also helps compare the relative order of clusters along one axis or along two adjacent axes. These local analyses become more powerful when combined with interactive selection of the direction of each axis.

At the global level of analyzing clusters across all axes, bundling is only partially useful. When the relative order of the clusters remains the same across all or most axes (perhaps after flipping some axes), clustering is sufficient to differentiate the clusters without recourse to additional methods such as color. Figure 5.1 demonstrates this case for the car example.

When the relative order of clusters changes from axis to axis, however, as seen in the bundled iris and netperf plots in Figure 5.5, bundling is insufficient to convey the global picture of where clusters fall on every axis. The zig-zags introduced by displaying the bundling points might even make it harder for the eye to track clusters. In these cases, bundling may have to be supplemented by color to convey the global flow of each cluster. Note that color by itself may not be particularly informative about the global flow of clusters, as demonstrated by the colored polyline plots in Figure 5.5.

Overall, bundling is a powerful addition to the representation of clusters in parallel coordinates, sufficient to support many kinds of local analyses and some cases of global analysis, while offering powerful synergies with color for complex global relationships.

CHAPTER 6

# The Parallel Coordinates Matrix

## 6.1 Introduction

The scatterplot is one of the most popular and widely applied visual representations of two-dimensional data in information visualization and statistical graphics. It is used to convey static information and serves as explorative data analysis tool in interactive setups. While single scatterplots represent two dimensions, the scatterplot matrix (SPLOM) (Hartigan, 1975) visualizes all two-dimensional, axis-aligned projections of a multivariate dataset. This is achieved by laying out 2D scatterplots in a matrix such that every row $i$ and every column $j$ represent one variable (Figure 6.1a).

In contrast, the parallel layout of axes in a parallel-coordinates plot of $N$ variables adds the constraint of a fixed ordering of axes, hindering the visualization of all pairwise relations in a single parallel-coordinates plot. A simple solution to this problem is to layout 2D parallel-coordinates plots in a matrix, as illustrated in Figure 6.1b. However, such a layout breaks the traceability of lines over all axes, which is one of the nice features of parallel coordinates. Also, note that switching axes in a scatterplot or 2D parallel-coordinates plot does not reveal any new information, such that $\frac{N(N-1)}{2}$ pairwise plots suffice to show all pairwise relations of an $N$-variate dataset. Hence, more than half of the space required by a traditional SPLOM is used to show redundant information. A common method to make use of the redundant space is to render only the lower (or upper) triangle of the matrix and use the other half and the diagonal for additional information about the data. While this is a reasonable technique

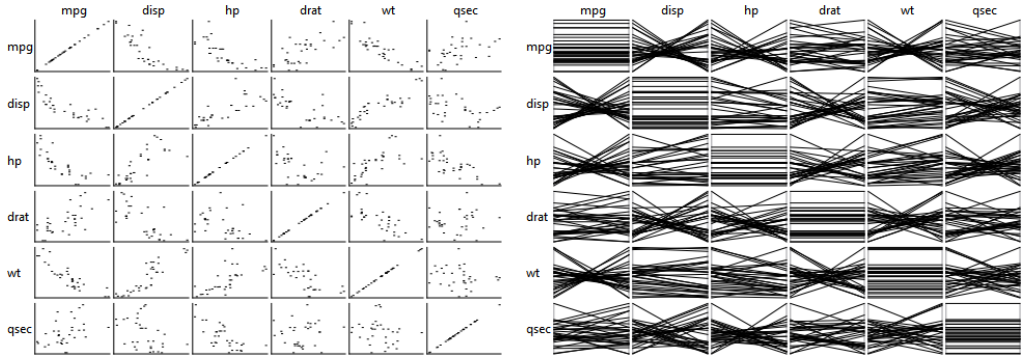



**Figure 6.1:** (a) SPLOM of six variables from the "ASA cars" dataset (Ramos and Donoho, 1983). For $N$ variables, $N^2$ plots are laid out in a matrix such that the cell at coordinates $(i, j)$ represents a scatterplot of the variable at the $i$-th row and the $j$-th column. (b) Replacing each scatterplot with a 2D parallel-coordinates plot breaks the continuity of lines representing data points.

if the space is filled with relevant information for the task at hand, one might be interested in using the available screen real-estate only for the visualization of pairwise correlations.

To combine the benefits of parallel coordinates and the scatterplot matrix, this chapter introduces the *parallel-coordinates matrix (PCM)*[1] as counterpart to the SPLOM. The design goals of the PCM are to

1. visualize all pairs of axes *without redundancy* using parallel coordinates while
2. each parallel-coordinates plot *contains all dimensions* of the original dataset and
3. all parallel-coordinates plots *show the same number of variables*.

As a result, the PCM is a list of high-dimensional parallel-coordinates plots, each with a different axis ordering, where the orderings are determined using a graph-theoretic approach. As the PCM is composed of a set of independent parallel-coordinates plots, existing ordering algorithms, interaction techniques, or visual representations can be used with the PCM.

[1] **Parts of this chapter have been published in:** J. Heinrich, J. Stasko, and D. Weiskopf. The parallel coordinates matrix. In *EuroVis–Short Papers*, pages 37–41, 2012.

## 6.2 Related Work

This section compares related work on ordering axes in parallel coordinates with the research presented in this section. Please refer to Section 2.3.2 for an extensive review on axis ordering in parallel coordinates.

Many algorithms have been proposed to measure the *goodness* of bivariate scatterplots in a preprocessing step (Wilkinson et al., 2005; Sips et al., 2009; Ankerst et al., 1998; Hurley, 2004) and use it as a measurement for its visual prominence, e.g. by ordering or highlighting the respective plots in a SPLOM. The SPLOM can also be used as navigational infrastructure for interactive exploration (Elmqvist et al., 2008).

For the ordering of axes in parallel coordinates, similar measures have been proposed to determine the weight of two-dimensional relations and to sort the axes accordingly (Ankerst et al., 1998; Hurley, 2004; Dasgupta and Kosara, 2010; Wilkinson et al., 2006; Ferdosi and Roerdink, 2011). The PCM is designed such that any of these ordering algorithms can be applied to one of the parallel-coordinates plots in the matrix.

Other layouts for bivariate parallel coordinates can be used to visualize single-to-many (Johansson et al., 2005a) and many-to-many (Lind et al., 2009) relations. For the latter, line continuity is not achieved while the former does not represent all pairwise relations. Yuan et al. (2009) use a combined visualization, where points are scattered between pairs of axes in parallel coordinates. Viau et al. (2010) presented the parallel scatterplot matrix (P-SPLOM) as a hybrid visualization between SPLOM and parallel coordinates. The P-SPLOM comprises the same number of bivariate plots as the SPLOM and therefore contains the same redundancy, which conflicts with the first design goal for the PCM. Albuquerque et al. (2009) order 3D parallel-coordinates plots in a matrix with $(N-1)/2$ columns and $N$ rows, rendering a total of $N^2 - 1$ 2D pairwise relations. Hence, their matrix shows redundant information and does not use N-D parallel coordinates.

Claessen and van Wijk (2011) have recently presented a general technique for the layout of two-dimensional plots using both point and line representations of data points. In their work, axes can be placed freely in a Cartesian space such that both a SPLOM and a PCM could be generated manually. However, their approach relies on the manual definition of an *attribute relation graph of interest* and a manual layout of axes. While their approach is generic, it requires a significant amount of manual labor to design a SPLOM or a PCM. We compare this to graph drawing: a manual layout of a node-link diagram is most general; still, automatic layouts are required in order to render even small graphs efficiently.

The method presented here is based on observations from Wegman (1990) and the related approach recently published by Hurley and Oldford (2010): Instead of using a single ordering, Hurley and Oldford compute all axis orderings required to see all pairwise dimensions in parallel coordinates and visualize them in one

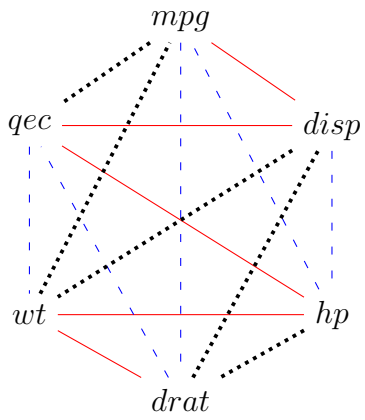



**Figure 6.2:** Hamiltonian decomposition of the complete graph $K_6$ for the cars dataset. Each line type (red, blue, and black) represents a Hamiltonian path.

large parallel-coordinates plot. However, in some cases, using a linear layout of axes introduces redundant pairs of dimensions. In contrast, the layout presented here displays all pairwise correlations without redundancy for any number of variables. In addition, the PCM makes a more effective use of the available vertical space.

## 6.3 Construction

### 6.3.1 Pairwise Correlation Graph

The first design goal of the PCM is to visualize all (unordered) pairwise relations in an $N$-dimensional dataset. The problem of finding these relations can be translated to visiting all edges in the undirected complete graph $K_N = (V, E)$, where the set of vertices $V = \{1, \ldots, N\}$ represents the indices of variables of the input dataset and edges $E = \{e_{ij} \mid i, j = 1, \ldots, N\}$ represent a bivariate relation between dimensions $i$ and $j$. Figure 6.2 shows the complete graph for the cars dataset with $N = 6$.

As the number of edges in $K_N$ is

$$e = \frac{N(N-1)}{2},$$

a minimum of $e$ 2D plots has to be investigated to see all pairwise relations. From a graph-theoretic point of view, this corresponds to finding a path in $K_N$ such that every edge is visited at least once.

Using 2D data representations, where each plot is rendered independently, the plots may be distributed arbitrarily over the available space. For parallel coordinates, however, it might be beneficial to exploit the multidimensional nature

of the plot, such that polylines representing individual data points can be followed over more than two axes. Hence, for $N$-variate data, it is desirable to see every dimension at least once in every parallel-coordinates plot. Using a graph description, this translates to a path in the corresponding complete graph such that all vertices are visited at least once. Adding this constraint to the goal of visiting all $e$ edges leads to the following problem statement:

*Given the complete graph $K_N$, find a minimal set of trails of length $N$ such that every edge is visited exactly once.*

### 6.3.2 Eulerian Trails and Hamiltonian Decomposition

To solve the previous problem statement, a *Hamiltonian decomposition* of an *Eulerian trail* of the complete graph $K_N$ is used. Here, we give only a short recapitulation of the graph-theoretic background required. Please refer to the work of Hurley and Oldford (2010) for details.

A Hamiltonian decomposition is an edge decomposition of a graph into *Hamiltonian paths* or *Hamiltonian cycles*. An *Eulerian trail* is a trail in a graph that visits every edge exactly once and an *Eulerian cycle* is an Eulerian trail that ends in the starting vertex. Similarly, a Hamiltonian path is a path in a graph that visits every vertex exactly once, while a Hamiltonian cycle is a Hamiltonian path that ends in the starting vertex. Now the problem can be restated accordingly:

*Given the complete graph $K_N$, find a Hamiltonian decomposition of an Eulerian trail (cycle) into Hamiltonian paths (cycles).*

While there are $(N-1)!$ Hamiltonian cycles for the complete graph $K_N$, we employ the *Lucas-Walecki Hamiltonian decomposition* to obtain $M = \frac{N}{2}$ *Hamiltonian paths* for even $N$ and $M = \frac{N-1}{2}$ *Hamiltonian cycles* for odd $N$. In the following, we use the construction algorithms as described by Hurley and Oldford (2010).

For $N = 2M$, we construct the $M \times N$ layout-matrix $H^N$ by defining

$$\begin{aligned} H^N[1,1] &= 0 \\ H^N[1,j] &= H^N[1,j-1] + (-1)^j(j-1) (\text{mod } N) \\ H^N[k,j] &= H^N[k-1,j] + 1 (\text{mod } N) \end{aligned}$$

where $j = 2, \ldots, N$ and $k = 2, \ldots, M$. Adding 1 to every value results in the final matrix that is used to lay out axes on the available canvas. The rows of $H^N$ are now Hamiltonian paths in $K_N$. As an example, the layout matrix for $N = 6$ reads:

$$H^6 = \begin{matrix} 1 & 2 & 6 & 3 & 5 & 4 \\ 2 & 3 & 1 & 4 & 6 & 5 \\ 3 & 4 & 2 & 5 & 1 & 6 \end{matrix}$$

Hurley and Oldford (2010) concatenate the rows to form the Eulerian trail $T$ which they use to render a single parallel-coordinates plot. This introduces

duplicate edges between vertices $H^N[i, N]$ and $H^N[i+1, 1]$. Instead, we simply use the rows of $H^N$ as the final axis orderings of the PCM.

For $N = 2M + 1$, $H^N$ is constructed by adding $N$ as the first and last vertex in each row of $H^{N-1}$. This results in $M$ Hamiltonian cycles in $K_N$. Concatenating these cycles and merging the common vertices at $H^N[i, N]$ and $H^N[i + 1, 1]$ eventually results in an Eulerian cycle. According to this algorithm, $H^7$ reads:

$$H^7 = \begin{matrix} 7 & 1 & 2 & 6 & 3 & 5 & 4 & 7 \\ 7 & 2 & 3 & 1 & 4 & 6 & 5 & 7 \\ 7 & 3 & 4 & 2 & 5 & 1 & 6 & 7 \end{matrix}$$

Hence, in contrast to the solution proposed by Hurley and Oldford, this layout does not introduce duplicate edges for neither odd nor even $N$.

Note that the construction algorithm used here is based on indices, such that the order of the first path or cycle is not fixed. Any order of axes can be used for the first parallel-coordinates plot.

## 6.4 Results

This section shows some example visualizations using the PCM for various datasets. The case studies presented in this section and accompanying figures are no in-depth analyses, but are intended to show how the PCM can be used to spot "interesting" patterns in the set of parallel-coordinates plots. With more selection techniques, time, and experience, an expert will be able to discover much more in the data (see (Inselberg, 2009) for some examples). We also want to stress that the analysis conducted here was driven by looking for patterns first, followed by investigating which variables contribute to these patterns. This complies with the visual information-seeking mantra (Shneiderman, 1996), as no particular question about the data has been raised prior to the analysis.

Figure 6.3 shows the PCM for the "ASA cars" dataset that was also used in Figure 6.1. In addition to the correlations that are also visible in the SPLOM (Figure 6.1), the PCM further shows lines expressing a similar pattern over a subset of variables. This is probably most striking in the third row, where a small set of lines with high values for "disp" move to the top of "wt" before dropping to low values for "mpg".

Another example was presented in Chapter 2 (page 38), where Figure 2.11 shows 8 dimensions of a census dataset, where each line represents one of the 50 states of the USA. In this example, some negative correlations are prominent, such as the relation between life expectancy ("Life Exp") and the murder rate ("Murder"). There also is a cluster of low percentage of high-school graduates ("HS Grad") showing that these states also have a low per-capita income ("Income"'). Using interactive features such as tooltips or a linked geographic map reveals that

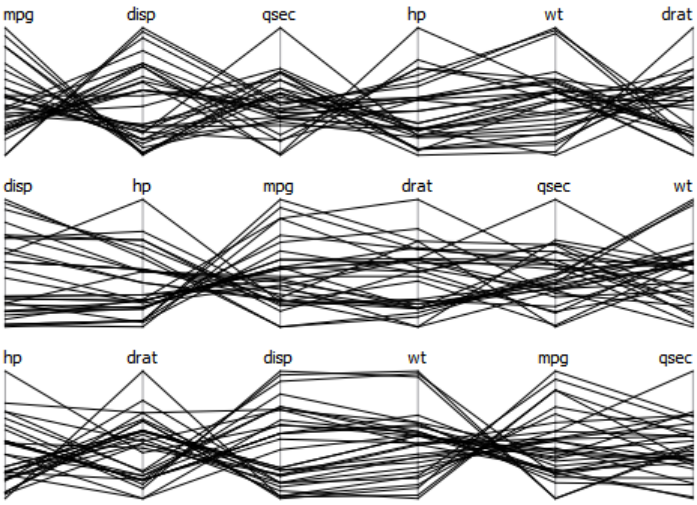



**Figure 6.3:** Parallel-coordinates matrix of the "ASA cars" dataset.

most of these states are located in the south-east. Note that the low-density region hinting at this discovery is most striking where "HS Grad" is paired with "Income" and might not have been spotted with a different axis ordering.

Figure 6.4 shows a seven-dimensional financial dataset that was used by Inselberg (2009, Chapter 10) for demonstration. In this example, one of the dimensions (which could be interactively chosen) appears on either side of all parallel-coordinates plots ("SP500"). In the figure, every line represents one of 384 weekly stock-market quotes of several currency exchange rates over a period of several years. Starting in the bottom row of the first PCM (top), we note a small cluster with low values of the "SP500" index and "GOLD" prices that indicates a positive correlation. The left part of the second parallel-coordinates plot shows another positive correlation between "SP500", "GDM" (German DMark), and "YEN". Being interested in this pattern, we wish to see more detail and focus on the center parallel-coordinates plot (bottom PCM), where a brushing operation can be applied at a higher level-of-detail. Now, we see that low "SP500", "GDM", "YEN", and the British Pound Sterling "BPS" go with a negative correlation between "BPS" and "TB3M". As expected, "GOLD" prices are low, while "TB30Y" varies in the mid-price section.

Finally, Figure 6.5 demonstrates a PCM of a twelve-variate dataset comprising 1 338 cameras that was used by Elmqvist et al. (2008). This example shows that even with 12 variables, the PCM can be used effectively to find interesting patterns and relations between dimensions. In this example, the "price" of cameras in the first parallel-coordinates plot shows only a small set of expensive models, but all with high resolution. In contrast, resolutions in the mid-price section vary, but are not very low and not very high. The "storage" dimension in row four shows that for expensive cameras, less or no storage is included.

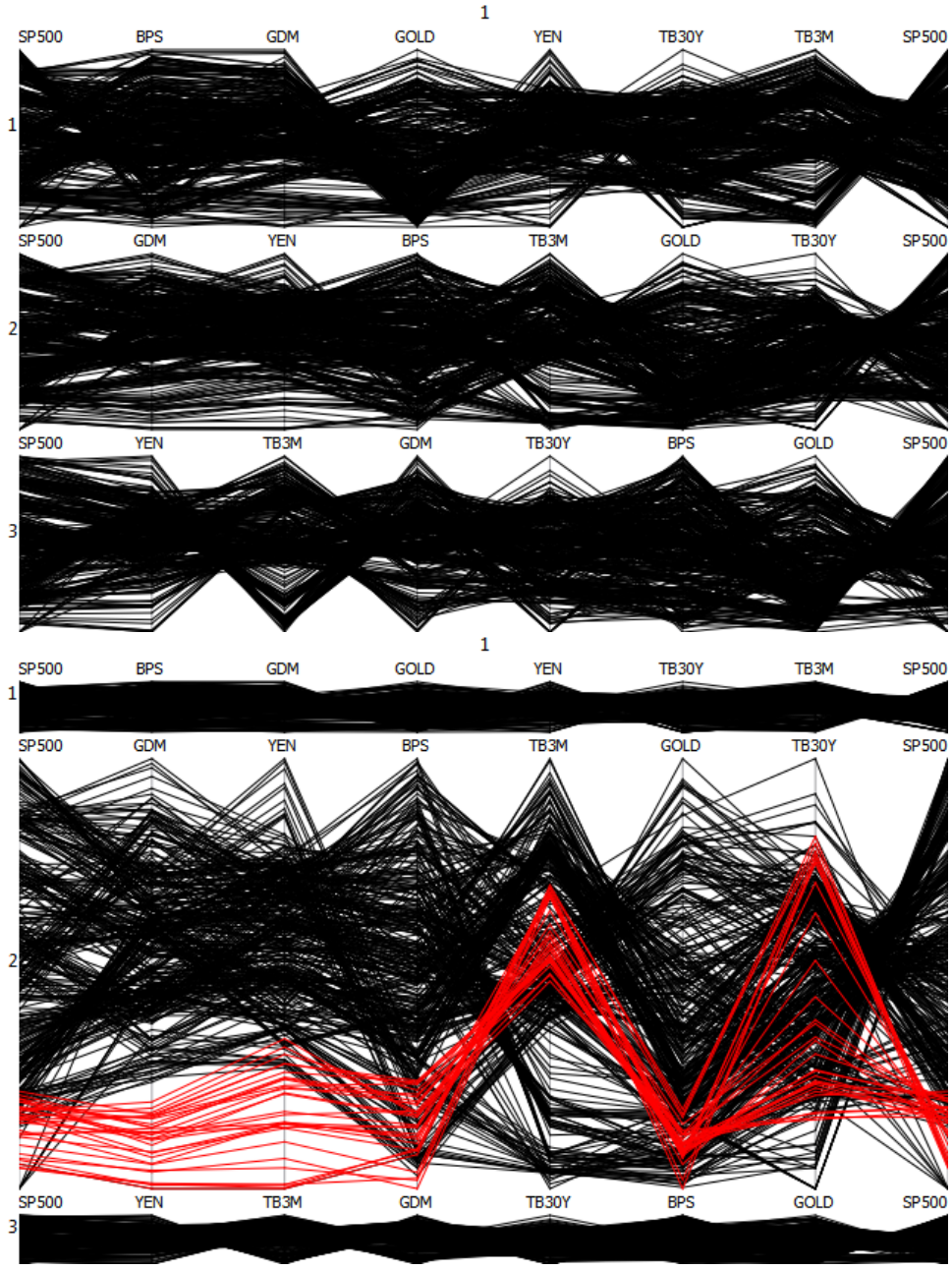



**Figure 6.4:** Parallel-coordinates matrix of a 7-dimensional financial dataset. See main text for details.

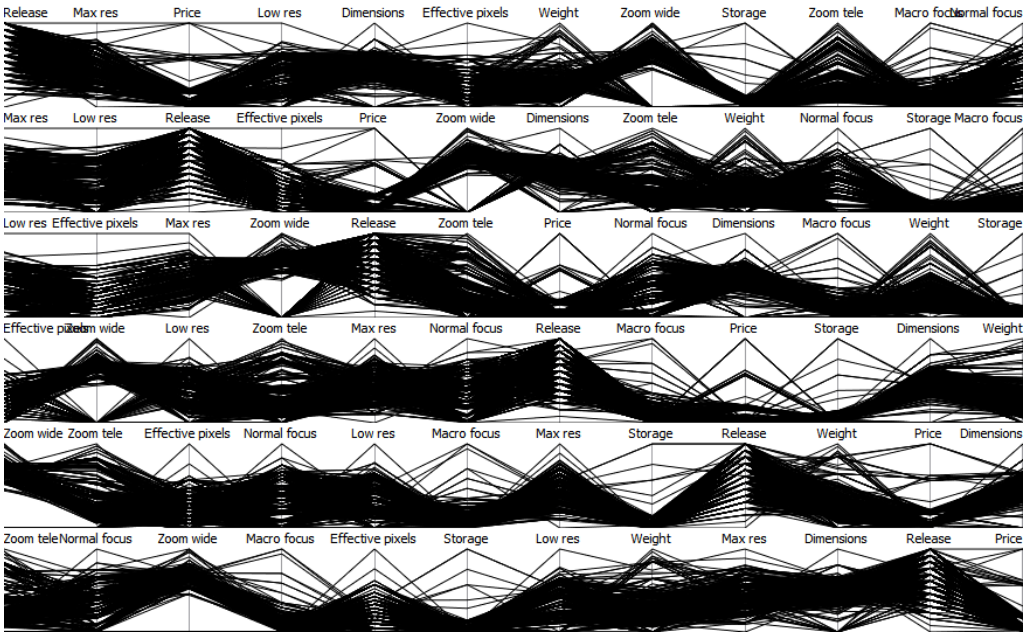



**Figure 6.5:** PCM of the 12-dimensional "cameras" dataset. The "price" and neighboring "Max resolution" and "Low resolution" in the first row show us that (1) there are three dense price segments: two low-cost segments, a small set of mid-price models and only three expensive cameras. From the direction of lines leaving the "price" axis for the mid-priced models, we can tell that the distributions of "Max" and "Low" resolutions are similar and there are no outliers. This is more difficult to say for low-cost cameras, as their resolutions seem to have a "wider" distribution over the neighboring axes. The "price" in row four suggests that the price for a camera does not necessarily predict the storage included. The most expensive models come without storage. Regarding the "Zoom wide" dimension, if an analyst only had the bottom-most parallel-coordinates plot for analysis, they might think at first glance that there is an outlier with no zoom at all, as we see a perfectly horizontal line to the neighboring axes. Comparing this with row number four, it becomes evident that there are many of such models.

## 6.5 Discussion

The PCM is a visualization presenting all pairwise correlations using parallel coordinates without redundancy for any number of dimensions. Using a simple layout algorithm, the PCM serves as a promising overview for parallel-coordinates plots making it a valuable tool to get an idea of a dataset and then focus on individual relations or plots. Due to the fact that the rows of a PCM are composed of multivariate parallel-coordinates plots, different rendering or interaction techniques can easily be incorporated, including continuous parallel coordinates or bundling as described in Chapters 3 and 5.

In contrast to the SPLOM, the PCM makes more efficient use of the available screen real-estate, as pairwise relations appear only once. However, the layout of the SPLOM facilitates labeling and navigation to particular scatterplots. Although this would require a thorough user study, we hypothesize that the SPLOM performs better at finding the relation of a particular pair of dimensions. If the task is exploratory such that recognition of patterns is more important than finding a specific pair or dimensions, however, we argue that an analyst benefits more from the space gained using a PCM.

CHAPTER 7

# CASE STUDIES

This chapter provides applications of parallel coordinates for different datasets from computational fluid dynamics, motion capturing, bioinformatics, and systems biology. The examples provided in this chapter are intended to give an impression of the usefulness of parallel coordinates for the interactive visual analysis of multivariate data. Both traditional line-based parallel coordinates as well as some of the techniques described in the previous chapters are demonstrated.

It is important to note that the case studies presented here do not necessarily represent the state of the art in the respective fields of analysis, as there are usually different ways to come to the same conclusions. The following case studies were conducted employing an explorative approach using parallel coordinates for visualization and have not been designed to be compared with alternative visualization approaches.

## 7.1 Computational Fluid Dynamics

This section compares discrete density-based parallel coordinates to continuous parallel coordinates for a multivariate CFD application.[1] Figure 7.1 illustrates discrete and continuous 4D parallel coordinates of the IEEE Visualization 2004 contest dataset "hurricane Isabel". The original data consists of 48 timesteps,

[1] **Parts of this section have been published in:** J. Heinrich and D. Weiskopf. Continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1531–1538, 2009.
J. Heinrich and D. Weiskopf. State of the art of parallel coordinates. In *STAR Proceedings of Eurographics 2013*, pages 95–116. Eurographics Association, 2013.

each containing measurements of the following $13$ attributes[2] with a spatial resolution of $500 \times 500 \times 100$:

- Height over ground (absolute)
- Cloud moisture mixing ratio
- Graupel mixing ratio
- Cloud ice mixing ratio
- Snow mixing ratio
- Water vapor mixing ratio
- Total cloud moisture mixing ratio
- Total precipitation mixing ratio
- Pressure (weight of atmosphere above a grid point)
- Temperature (Celsius)
- X wind speed (positive means winds from west to east)
- Y wind speed (positive means winds from south to north)
- Z wind speed (positive means upward wind)

For this case study, the first timestep and six dimensions are used. The visualized dimensions are the vertical spatial position (height), temperature, pressure, and wind velocity. Both temperature and pressure are contained in the original dataset, whereas wind velocity is computed from wind speed in x-, y-, and z-direction. Every dimension was normalized independently to the range $[0, 1]$ before computation. The original uniform grid was triangulated and continuous parallel coordinates were computed using the triangulated data algorithm presented in Section 3.4.1. Furthermore, tetrahedra containing invalid attribute data such as N/A-values were discarded.

For analysis on commodity hardware, it is common practice to downsample the original dataset before visualization (Doleisch et al., 2004). Figure 7.1 therefore compares line-based density-plots and continuous parallel coordinates for four dimensions in three different spatial resolutions (original, and downsampled to $50 \times 50 \times 10$ and $100 \times 100 \times 20$). The most prevalent character of the series of standard parallel coordinates is the increasing amount of clearly visible clusters resulting from the discrete mapping of the vertical spatial coordinate (height). Only at high resolutions the true character of the first dimension can be revealed, indicating a linearly increasing function defined on a continuous domain. If only one of the plots were available, it could falsely be interpreted as a set of multivariate clusters with equal values on the first dimension. Continuous parallel coordinates do not suffer from this problem, as linear interpolation of

[2] As described on the corresponding webpage: http://vis.computer.org/vis2004contest/data.html

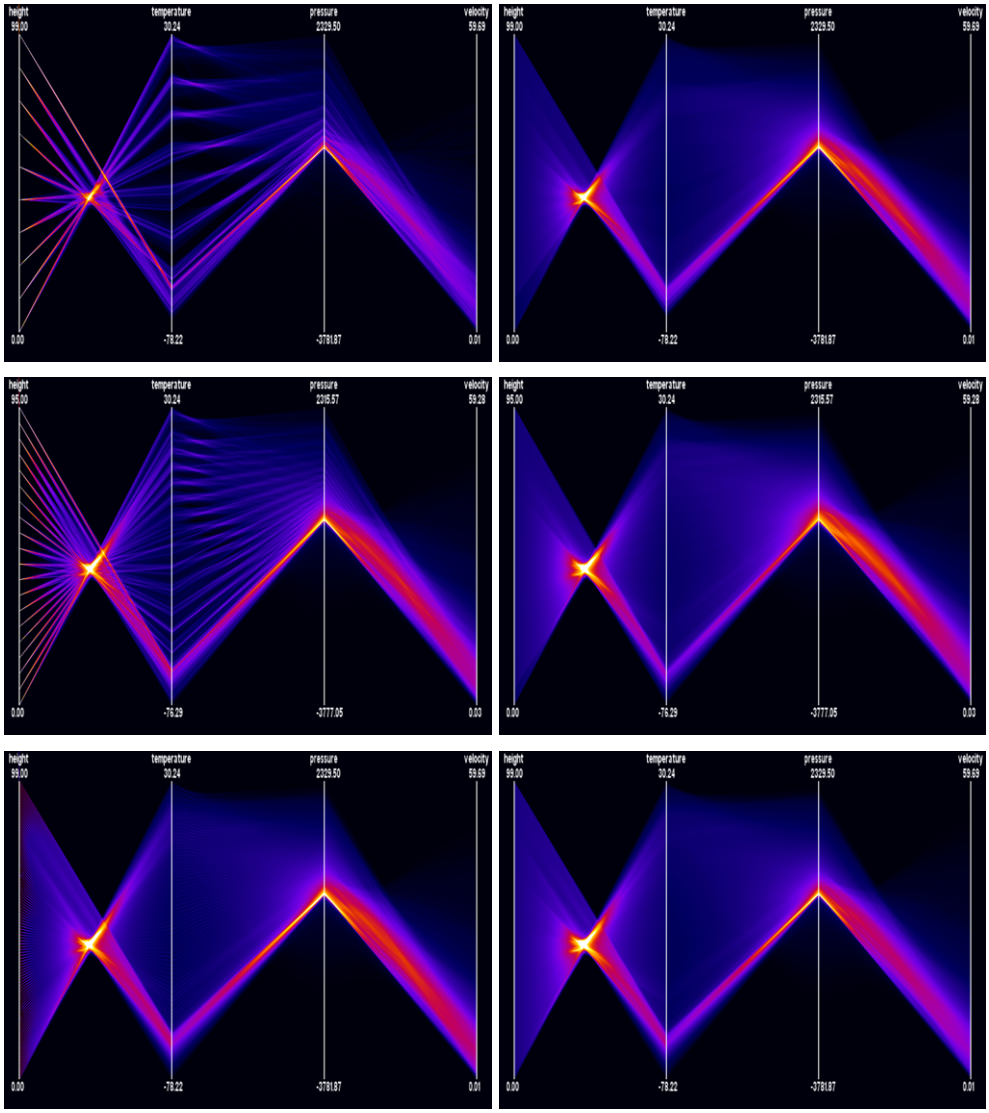



**Figure 7.1:** Discrete and continuous parallel coordinates for the "hurricane Isabel" dataset at different spatial resolutions (50×50×10, 100×100×20, 500×500×100 from top to bottom). On the left side, discrete parallel coordinates are shown with the corresponding continuous version on the right side. Sampling artifacts stemming from the discrete mapping of the vertical spatial coordinate (height) lead to misrepresentation of key information in discrete parallel coordinates.

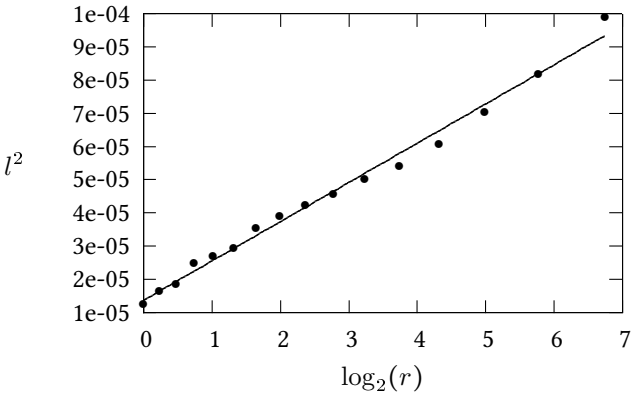



**Figure 7.2:** Relation of the relative $l^2$ distance with spatial sampling resolution $r = \frac{500 \cdot 500 \cdot 100}{n_x \cdot n_y \cdot n_z}$, where $n_i$ denotes the number of samples in dimension $i$. Note that $r = 1$ corresponds to the full-resolution dataset with decreasing resolutions for increasing $r$. Both $l^2$ and $r$ are given relative to ground truth, i.e. to the full-resolution dataset. Due to the logarithmic scale on the $x$-axis, an exponential relation between sampling resolution and $l^2$ distance can be concluded. To emphasize this relation, a linear regression line was added to the plot.

values is inherently contained in the density model. This can nicely be seen in Figure 7.1, where the equal distribution of samples on the first dimension can already be observed at low resolutions. Note that this is a key information that is entirely missing in discrete parallel coordinates.

Another observation is that continuous parallel coordinates of low-resolution data rapidly converge to ground truth, i.e. plots computed from full-resolution data. In order to obtain a numerical measure for similarity, the $l^2$-norm of the difference of density for different spatial sampling rates to the original dataset was computed with floating-point precision for a single time step of the dataset. The results (Figure 7.2) show that difference decreases exponentially with increasing spatial sampling resolution (note the logarithmic scale in the figure). Furthermore, the largest $l^2$ value of $10^{-4}$ is still very small, emphasizing that the main information contained in the data is already captured by low-resolution plots.

Figure 7.3 illustrates three dimensions of the same dataset in parallel coordinates and scatterplots using the progressive refinement algorithm of Chapter 4. For illustration purposes, images were constructed using different sample sizes, sampling algorithms, kernel sizes, and colormaps. Sample values were obtained using trilinear interpolation for all plots. In the parallel-coordinates plot, the same features as in Figure 7.1 are visible.

Figure 7.4 shows pressure, velocity, and wind speed of the hurricane from west to east and north to south with line-based density plots and continuous parallel

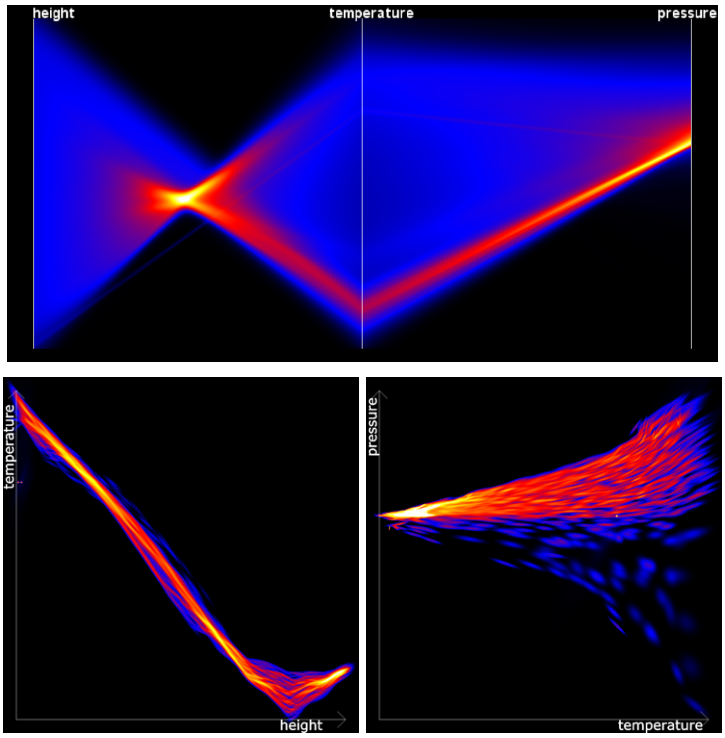


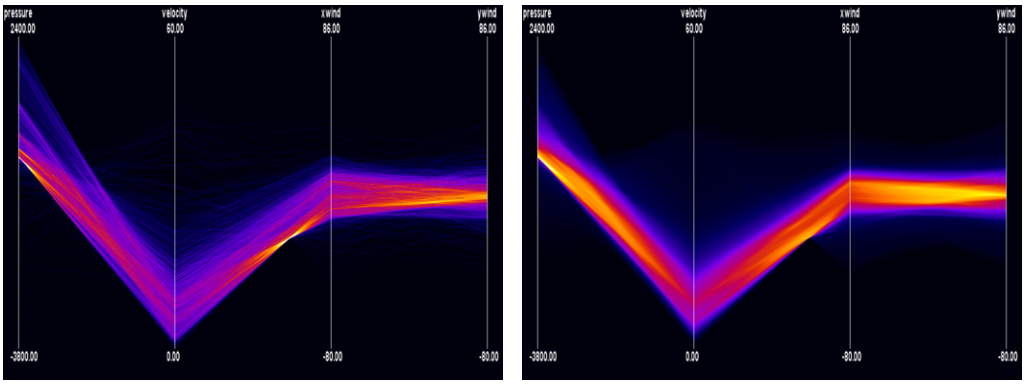

**Figure 7.3:** Splatted parallel coordinates (top) and scatterplots (bottom) showing "height", "temperature", and "pressure" for the first time step of the "hurricane Isabel" dataset.

coordinates. The images were created using a downsampled version of the original dataset ($25 \times 25 \times 5$), and a different transfer function was applied to emphasize dense regions between pressure and velocity. A feature apparent in both plots is the high-density region at low pressure and velocity, constituting the eye of the hurricane. However, the characteristics of the distribution of values within this region differs considerably between the plots: continuous parallel coordinates show a single high-density peak for small velocities, where the discrete density plot suggests two peaks. An analysis of the full dataset (Doleisch et al., 2004) shows that velocities have an approximately Gaussian shape.

## 7.2 Multivariate Trajectories

This section presents an application of continuous parallel coordinates for the analysis of angular values from motion capture data (Carnegie Mellon University, 2013), e.g. to identify postural or mobility restrictions.

Here, multivariate trajectories represent data values (angles) of a single entity (the joint) at different points in time. Samples are taken along a continuous path within a high-dimensional space, allowing for interpolation between samples as




(a) (b)

**Figure 7.4:** (a) Discrete and (b) continuous parallel coordinates for the "hurricane Isabel" dataset at a spatial resolution of $25 \times 25 \times 5$. Continuous parallel coordinates show a single peak at low velocities where two peaks can be seen in discrete parallel coordinates. The high-density region at low pressure and low velocity constitutes the eye of the hurricane.

part of signal reconstruction. Continuous parallel coordinates can then be used to visualize trajectories independent from the temporal sampling resolution.

To analyze the data, an application based on MegaMol™ (Grottel et al., 2009, 2010) is used[3]. The system (Figure 7.5) comprises three coordinated views that show one spatial (right) and two time-based representations of the data (left): (i) the temporal heatmap visualizes the value of angles (represented by vertical axes) from the first time point (bottom) to the last time point (top), while (ii) continuous parallel coordinates show the correlation between angles (also represented by vertical axes) aggregated over time. For brevity, however, we will restrict the discussion to continuous parallel coordinates and discard the other visualizations.

For this case study, datasets provided by the Carnegie Mellon University Motion Capture Database (Carnegie Mellon University, 2013) are used, each representing a single animated skeleton of a human. With the exception of dataset 1.1 that was captured from a jumping movement, all datasets describe motion capture data from various subjects walking straight in one direction at different speeds. For visualization in parallel coordinates, angle rotations for each joint are mapped to axes in parallel coordinates. Then, for each consecutive pair of time points, the respective footprint is computed as described in Section 3.3.4 and the final density is accumulated into a framebuffer. Finally, a colormap is applied to the density field.

Figure 7.6 shows three angles of the right leg from datasets 1.1 (jumping forward) and 7.1 (normal walking). The jumping motion in dataset 1.1 generates negative

[3] The software was provided by Sebastian Grottel

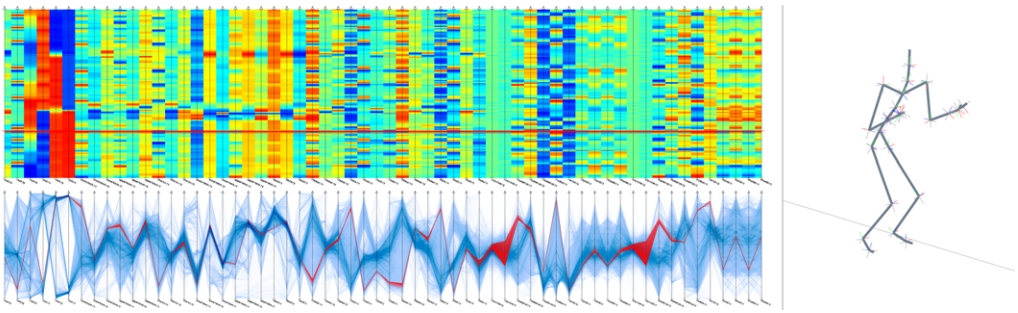


**Figure 7.5:** Screenshot of the system used for the analysis of motion capture data. The software comprises three coordinated views: a spatial view (right) showing the bone skeleton, a temporal heatmap (top left) visualizing the time-dependency of rotational angles at the joints of the skeleton, and continuous parallel coordinates (bottom left) displaying the angles aggregated over time.

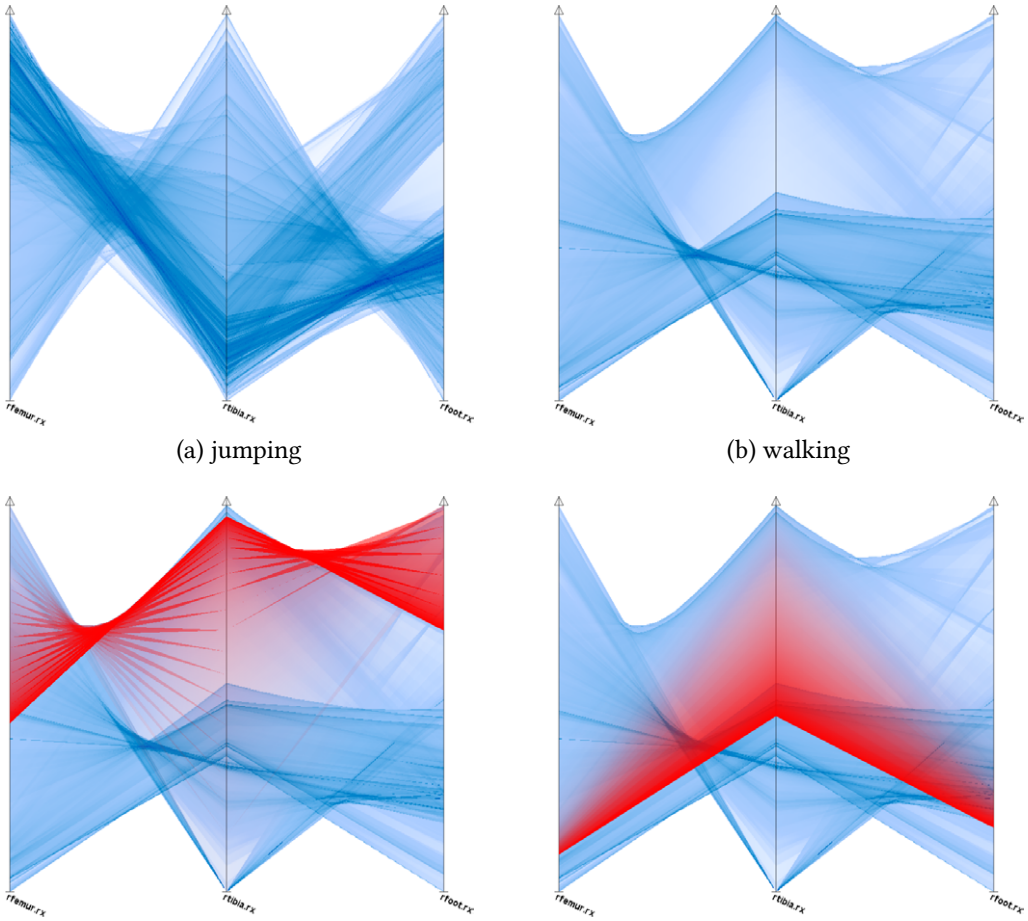
correlations between both pairs of dimensions that become visible in continuous parallel coordinates as high-density regions between adjacent axes. For the walking motion of dataset 7.1, correlations are less obvious in the aggregated view. However, correlations become visible if single time points during an animation are highlighted (together with a number of prior time points): negative and positive correlations alternate as a result of the periodic movement. As the same pattern was observed for other walking subjects as well, it might be a typical pattern for normal walking.

## 7.3 Gene Expression Data

This section presents several case studies employing parallel coordinates for the visualization of gene expression data.[4]

Microarray-based gene expression studies generate data for several thousands of genes (data samples) under numerous different conditions (dimensionality of the data). The data itself is stored in the *gene expression matrix* as the fundamental structure, which is typically used as the basis for visual analysis as well. This matrix contains the expression values of a gene under different conditions in its rows and the gene expression values of a certain condition in its columns. Conditions imply a large variety of different meanings, which can be external or internal stress factors (e.g., heat or chemical irritation) under which the cell is growing, pathological states of the cell, mutated cells, or time points of time series.

[4] **Parts of this section have been published in:** J. Dietzsch, J. Heinrich, K. Nieselt, and D. Bartz. SpRay: A visual analytics approach for gene expression data. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 179–186, 2009.




(a) jumping (b) walking

(c) walking with highlight at $t_a$ (d) walking with highlight at $t_b$

**Figure 7.6:** Continuous parallel coordinates of three dimensions from the right leg of datasets (a) 1.1 and (b) 7.1. The negative correlation between these dimensions in (a) is a result of the subject jumping during motion capture. This type of correlation is not visible for the other subject walking in (b). However, animating the view with a red highlight for the current point-in-time reveals (c) negative correlations and (d) positive correlations. The same colormap as in Figure 3.6 was applied.

There is a strong need for adequate methods to reveal relevant effects that are latently contained in the data and to separate these from the noise attributed to the measuring procedure. Several statistical methods already exist that attempt to achieve this goal. Nevertheless, the analysis of a microarray-based gene expression experiment is still a challenging task. Often the application of only one method is not successful and it is necessary to employ a number of different methods. A promising approach to address this issue and to profile the statistical methods used in an analysis is the conjoined visual exploration of the original data together with the associated deduced statistical data in a common data

space (Dietzsch et al., 2009; Turkay et al., 2011). This combination of automatic (statistical) and visual analysis leads to a visual analytics (Thomas and Cook, 2005) approach that provides more insights in the structure of the data and that prevents misleading impressions as much as possible at the same time. The following case studies were conducted with SpRay (Dietzsch et al., 2009), a visual analytics system that allows this type of analysis by providing interactive parallel coordinates coupled with a popular statistical-computing package (R Core Team, 2012). The first example focuses on genes that are active during the yeast cell cycle, and hence expose a cyclic expression pattern, which in turn is used for the model-based analysis. For the second example, parallel coordinates are used as a tool to guide the statistical analysis of differential expression. It is furthermore used to explore the effect of different model-free statistical correction methods to support the selection of the most appropriate one. Both examples demonstrate typical daily use applications of microarray analysis.

### 7.3.1 Finding Periodic Patterns

The first dataset is well-known in bioinformatics and describes genes of the yeast Saccharomyces cerevisiae that are influenced by the cell cycle (cycle-regulated). Spellman et al. (1998) investigated the periodical variation of gene transcript levels in association with the cell cycle in a comprehensive microarray-based analysis. To get reliable gene expression signals, the cells from yeast cultures were first synchronized by an arrest-release synchronization method resulting in three different gene sets ($\alpha$, CDC15, elutriation). Messenger RNA (mRNA) was extracted at consecutive time points following synchronization, and gene expression values of more than $6000$ genes were measured using two-color complementary DNA (cDNA) microarrays. The arrays were scanned and the basic analysis was done with common methods for background correction, normalization, and quality filtering of the spot signals. On top of this analysis, cyclicity, correlation, and clustering procedures were employed to quantify and characterize the association of the gene transcript levels with the cell cycle phases. Spellman et al. found $800$ genes which satisfy the minimum criterion for cell cycle regulation that was defined. Follow-up analysis of the response of these genes to induce a certain cell phase and the analysis of promoter sites of these genes showed further evidence for a cell cycle association for a subset of these genes. The “yeast cell cycle dataset” was closely examined in many papers. Shedden and Cooper (2002) re-analyzed the data and derived a more specific conclusion. They found out that the randomization of data showed less strong periodic patterns than the experimental data. Therefore, noise and random data fluctuations could be ruled out to contribute to the cyclicity of the data.

For this case study, only the data of the $\alpha$ factor arrested cells was used. Visualizing the raw data in a parallel-coordinates plot, where axes denote time points[5] $t_j$, results in a heavily cluttered view, as illustrated in Figure 7.7.

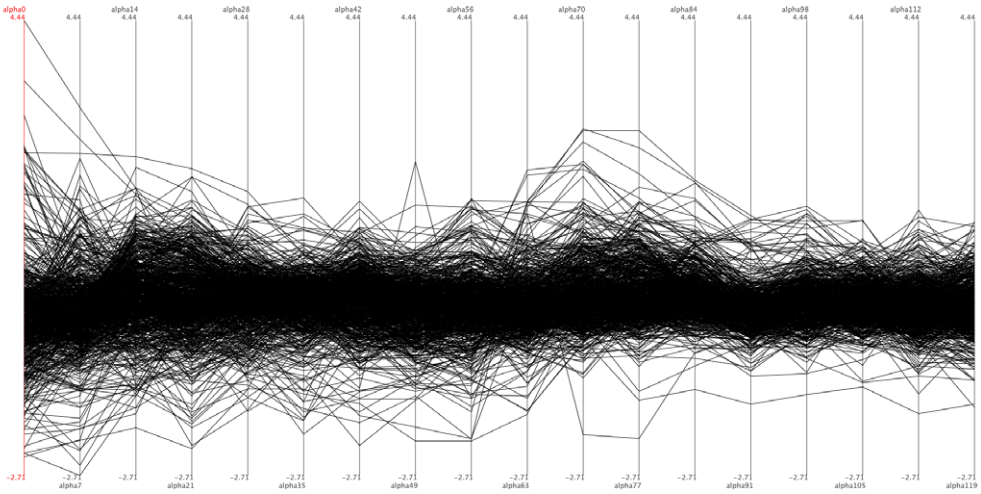


**Figure 7.7:** Parallel-coordinates plot of 18 time points for the $\alpha$ factor arrested cells of the "yeast cell cycle dataset" (Spellman et al., 1998).

Hence, the data was re-analyzed in a similar way to Shedden and Cooper's analysis with a sinusoidal regression fit of cell cycle genes. The expression values $y(t_j)$ of a gene at time points $t_j$ were least square fitted against a linear model with the two harmonic basic curves:

$$y(t_j) = \beta_s \sin\left(\frac{2\pi}{T} t_j\right) + \beta_c \cos\left(\frac{2\pi}{T} t_j\right) + r(t_j). \tag{7.1}$$

To detect the sinusoidal expression pattern of genes according to the cell development, the period $T$ was set to the nominal interdivision time of 66 minutes specified by Shedden and Cooper (2002). The value $y(t_j)$ is decomposed using Equation (7.1) into the interesting harmonic part:

$$h(t_j) = \beta_s \sin\left(\frac{2\pi}{T} t_j\right) + \beta_c \cos\left(\frac{2\pi}{T} t_j\right)$$

and the residual part $r(t_j)$ that quantifies the aperiodic content of $y_j$ or oscillations with a significantly different period in comparison to the selected value of $T$. For visualization, the harmonic part $h(ti)$ was expressed as a single modulated and shifted sine wave:

$$h(t_i) = A \sin\left(\frac{2\pi}{T_k} t_i + \varphi_0\right). \tag{7.2}$$

The amplitude $A$ and the zero-phase angle $\varphi_0$ are determined by the coefficients $\beta_s$ and $\beta_c$ and can be calculated with the help of the addition theorems of

[5] Note that this type of plot can also be interpreted as a time-series visualization due to the fixed axis order.

trigonometry. Adding $\varphi$ as additional variable to the raw dataset shown in Figure 7.7 now allows us to use this statistically derived additional information to visualize the distribution of phase-shifts for every gene in parallel coordinates. Figure 7.8 shows the results where the coloring of all genes is defined by the zero-phase angle $\varphi_0$.

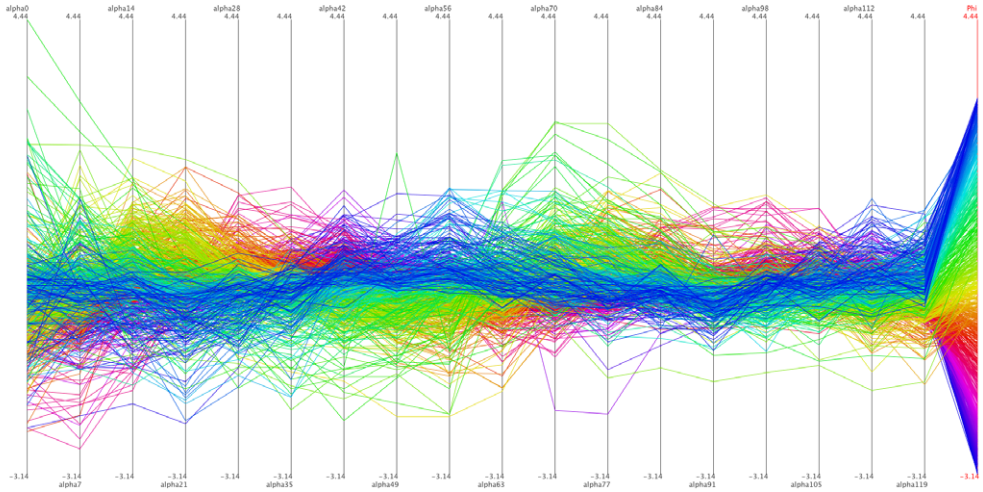


**Figure 7.8:** The same data as in Figure 7.7 with color mapped to the zero-phase angle $\varphi_0$ (rightmost axis).

With brushing in parallel coordinates, it is possible to emphasize further different aspects of the data, for instance to search genes that show a nearly anti-correlation pattern along the cell cycle. Figure 7.9 shows the result of a specific zero-phase selection on the $\varphi_0$-dimension, exposing two antisymmetric cycles.

### 7.3.2 Emphasizing Relevant Expression Patterns

The second dataset is taken from a study (Zieker et al., 2005) that investigated the effects of an exhausting endurance exercise on the immune system. It is generally believed that a strong influence exists, which is attributed to both a cellular shift in the composition of the peripheral blood and to changes in gene expression levels. The study used a custom-made cDNA microarray of immune and stress response related genes to investigate these different aspects in a systematic way. Blood samples were taken from eight well-trained male half-marathon runners in rest before the run ($t_0$), immediately (up to 15 minutes) after the run ($t_1$), and 24 hours after the run ($t_2$). The most interesting effects were seen between the status before the run ($t_0$) and immediately after the run ($t_1$), hence only these time points are included in this investigation. The study indicated interesting changes in the transcript level of inflammatory genes and even more interesting evidence for an association with the antioxidative defense.

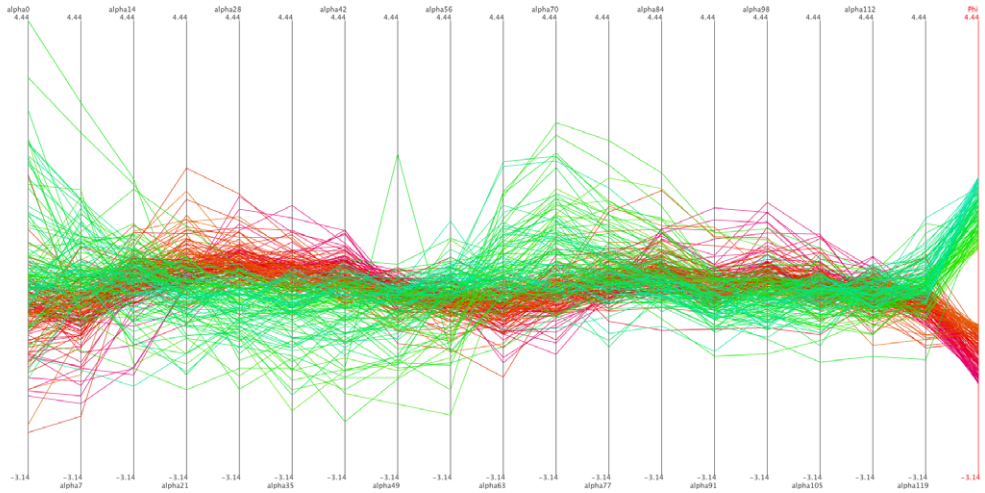


**Figure 7.9:** Selection of two groups of genes that show an anticorrelated gene expression pattern along the cell cycle.

Both indicate the higher stress level of the body. Here, the evaluation of the behavior of the ten different p-value correction methods is of interest. The data of the eight male runners are interpreted as biological replicates and are assigned to $8$ dimensions of the parallel-coordinates plot. They are represented as the log-ratios of the gene expression values at times $t_0$ and $t_1$: $\log\left(\frac{y(t_1)}{y(t_0)}\right)$. $10$ added dimensions represent deduced statistical values; mean and standard deviation of the log ratios of all runners, the p-value of a t-test against the null hypothesis of no difference in gene expression between time points $t_1$ and $t_0$, and seven p-values corrected by different methods to address the multiple testing problem (Bonferroni, Holm, Hochberg, Sidak ("SidakSS", "SidakSD"), Benjamini-Hochberg ("BH"), and Benjamini-Yekutieli ("BY")), which are standard methods in bioinformatics[6]. Overall, this results in a parallel-coordinates plot with $345$ polylines for all genes and $18$ dimensions.

The specific choice of an adequate correction method is a nontrivial problem in the context of microarray data analysis. If the correction method is too rigorous, many interesting gene expression changes could be missed (high false negative rate). Also, if the method is not strict enough, too many false positives render the follow up investigations time-consuming, extensive, and expensive. An applicable trade-off must be found based on the goals of the study. Figure 7.10 gives a good impression of the eight measured values and the ten deduced statistical parameters.

The isomorphic luminance-based two-color coding is defined by the dimension that represents the Bonferroni-corrected p-values. This method is the most rig-

[6] Although all these methods are standard, the question which method is the most appropriate one for a specific situation is still disputed.

orous and was selected for the study to get very reliable results and a very low false positive rate. It furthermore exhibits a very regular spread over the whole significance interval. The most interesting genes are genes whose corrected p-values fall below the defined level of statistical significance, which was predefined to $0.05$ in this study. As can be seen in Figure 7.10, the most interesting genes are largely hidden by the large amount of other gene data values. Hence, it is necessary to de-emphasize the irrelevant genes and to emphasize the most interesting genes (see Figure 7.11, red colored samples).

## 7.4 Biological Reaction Networks

This section presents a visual-analytics approach employing parallel coordinates and support vector machines (SVMs) for the analysis of models of heterogeneous cancer cell populations.[7]

Most properties of intracellular biological systems arise from the complex interaction of biochemical species such as genes and proteins. The interaction structure can be modeled using biological reaction networkss (BRNs), which consist of chemical species (vertices) and reactions (edges). An example of the proapoptotic signaling pathway is given in Figure 7.12. The dynamics of a BRN is typically modeled for a single cell using ordinary differential equations (ODEs), where initial conditions and, e.g., kinematic reaction rate coefficients, are described using a parameter-vector $\theta$. The values for $\theta$, however, can usually not be measured directly, but must be estimated either from experimental data or simulations.

Many complex mechanisms such as cell death or proliferation depend on the state of other cells in a population, which can be heterogeneous in terms of, e.g., cell age or protein abundance. Thus, to ultimately understand and control the behavior of populations, the key sources of cell-to-cell variability have to be unraveled.

Unfortunately, this is challenging due to experimental constraints. Most experimental systems and measurement devices only allow for the simultaneous assessment of a few cellular properties on a single cell basis. This prohibits the purely experimental analysis of processes that depend on many different cellular properties. To some extent, such experimental limitations can be overcome using mathematical models and simulation.

[7] **Parts of this section have been published in:** J. Hasenauer, J. Heinrich, M. Doszczak, P. Scheurich, D. Weiskopf, and F. Allgöwer. Visualization methods and support vector machines as tools for determining markers in models of heterogeneous populations: Proapoptotic signaling as a case study. In *Proceedings of the Workshop on Computational Systems Biology*, pages 61–64, 2011.
J. Hasenauer, J. Heinrich, M. Doszczak, P. Scheurich, D. Weiskopf, and F. Allgöwer. A visual analytics approach for models of heterogeneous cell populations. *EURASIP Journal on Bioinformatics and Systems Biology*, 2012(1):1–13, 2012.

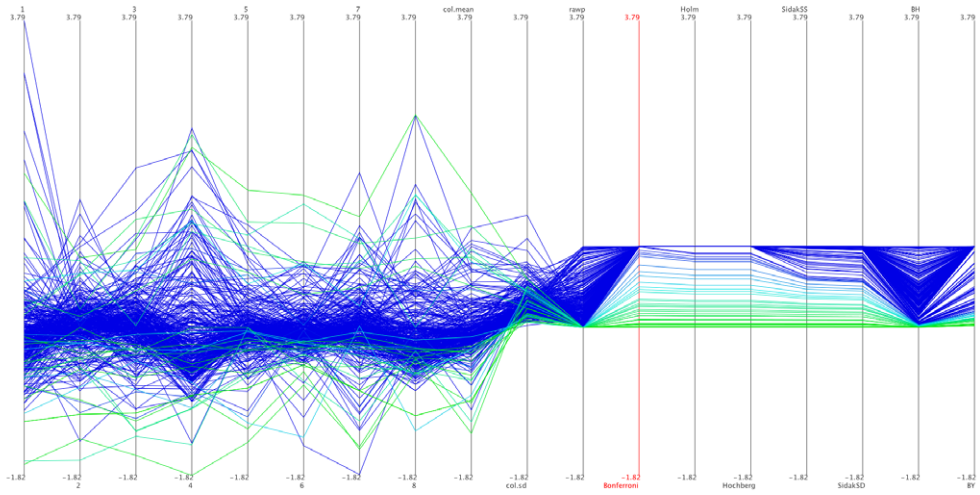



**Figure 7.10:** Parallel-coordinates plot of the "half-marathon" data with 10 added dimensions corresponding to deduced statistical values. This figure shows that the p-value of the majority of samples is of very low (blue) significance. The view of the parallel-coordinates plot (with a color-coding according to the p-value corrected after Bonferroni (highlighted in red) shows the influence of the different correction methods.

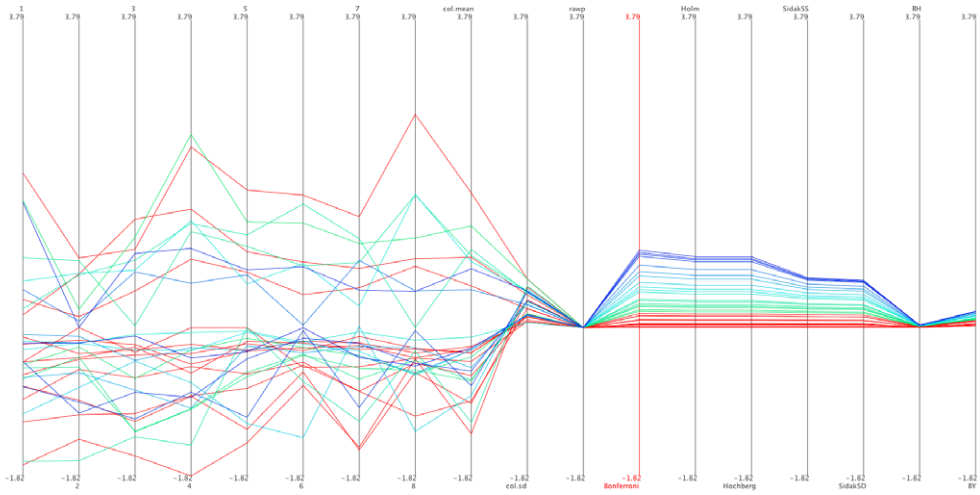



**Figure 7.11:** The same data as in Figure 7.10, but with all genes with an insignificant difference between gene expression values culled and highly significant genes emphasized in red.

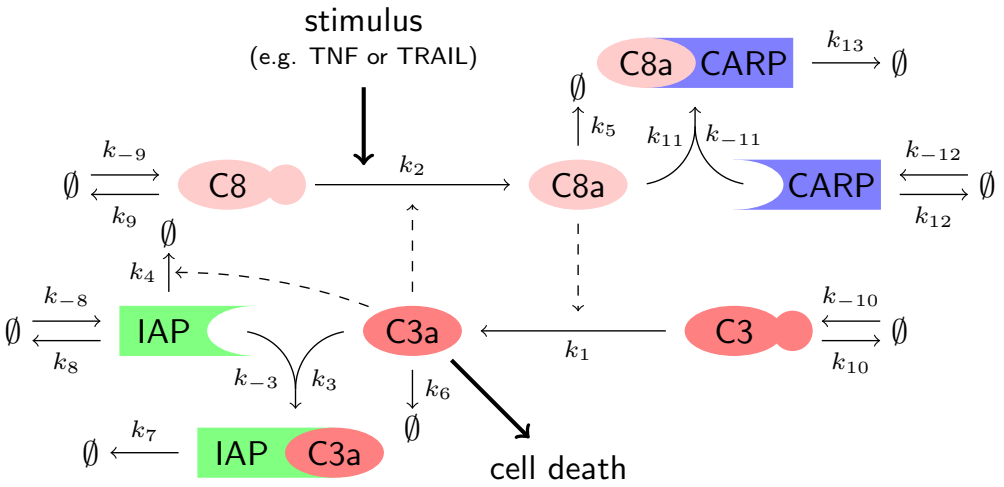



**Figure 7.12:** Illustration of proapoptotic signaling pathway. Solid arrows refer to conversion reactions, dashed arrows indicate enzymatic activity, and thick arrows illustrate inputs and outputs of the system.

In this case study, we focus on the deterministic differences among cells in populations of non-interacting cells, which can be modeled with differential parameter values and initial conditions for each cell. Several methods exist to infer the distribution of parameters and initial conditions from experimental data (Hasenauer et al., 2011b,c) and to obtain quantitative, mechanistic models for cell populations. Unfortunately, the resulting agent-based models are in general highly complex, preventing the analysis using common tools for dynamical systems. To the best of our knowledge, for models of heterogeneous cell populations, no structured analysis approach is available. To study population models and to facilitate a model-driven analysis of heterogeneity, flexible methods are required that do not rely on a purely analytical methods.

We propose to use parallel-coordinates plots and SVMs to analyze complex models of heterogeneous cell populations, particularly addressing the question: "Which parameters cause the heterogeneity of the population's response?". We show that parallel-coordinates plots can be used to obtain a qualitative understanding of the system, whereas SVMs allow for assessing the performance of marker combinations quantitatively. Good markers are defined as single cell parameters that facilitate a good prediction of the cell fate decision or the quantitative property under consideration of the individual cell.

### 7.4.1 Models for Heterogeneous Cell Populations and Decision Processes

#### Mechanistic Population Model

Population dynamics are described using an ensemble of cells (agents). This yields the agent-based population model:

$$\Sigma_{\text{pop}} = \left\{ \Sigma(\theta^{(i)}) \mid i = \{1, \dots, N\}, \theta^{(i)} \sim \Theta(\theta) \right\},$$

in which the superscript $(i)$ specifies individual cells within the population, $N \in \mathbb{N}$ denotes the size of the cell ensemble and $\Sigma(\theta^{(i)})$ is the model of the $i$-th cell. The single cell model $\Sigma(\theta^{(i)})$ may belong to the class of Markov jump processes, stochastic differential equations, or ODEs (Hasenauer et al., 2011b). Since we are mainly interested in signal transduction and decision making in this study, we consider ODE models. Each individual cell of $\Sigma_{\text{pop}}$ is described by

$$\Sigma(\theta^{(i)}) \colon \dot{x}^{(i)}(t) = f(x^{(i)}(t), \theta^{(i)}), \quad x^{(i)}(0) = x_0(\theta^{(i)}),$$

with state vector $x^{(i)}(t) \in \mathbb{R}^n_+$ and parameter vector $\theta^{(i)} \in \mathbb{R}^q_+$. The vector field $f \colon \mathbb{R}^n_+ \times \mathbb{R}^q_+ \to \mathbb{R}^n$ describing the cell dynamics is locally Lipschitz and the mapping $x_0 \colon \mathbb{R}^q_+ \to \mathbb{R}^n_+$ is continuously differentiable. Parameters $\theta^{(i)}$ may be kinetic constants, such as synthesis, degradation, or reaction rates.

Heterogeneity among cells of the ensemble is modeled with differential parameter values $\theta^{(i)}$ and initial conditions $x_0(\theta^{(i)})$ among individual cells. The density of parameters $\theta^{(i)}$ is given by a probability density function $\Theta \colon \mathbb{R}^q_+ \to \mathbb{R}_+$. Thus, the probability of observing $\theta^{(i)} \in \Omega$ is

$$\text{Prob}(\theta^{(i)} \in \Omega) = \int_\Omega \Theta(\theta) \mathrm{d}\theta.$$

#### Qualitative and Quantitative Properties of the Single Cell Response

Given the mathematical models introduced above, we study qualitative and quantitative properties of the single cell responses. Qualitative properties are defined as the outcome of a discrete decision process, e.g. whether the state of a bistable system converges to one or another steady state, or whether a certain concentration threshold is reached. In contrast, quantitative properties allow the assessment of small differences among cells, such as the time point when a particular threshold is exceeded.

To define single cell properties given the single cell trajectory $x^{(i)}(\cdot)$, functionals $F_\varphi : \ell^1 \to \mathbb{R}$ and $F_\delta : \ell^1 \to \{-1, +1\}$ are introduced. $F_\varphi$ is used to evaluate the quantitative property $\varphi^{(i)} = F_\varphi(x^{(i)}(\cdot)) \in \mathbb{R}$, while $F_\delta$ determines the qualitative property $\delta^{(i)} = F_\delta(x^{(i)}(\cdot)) \in \{-1, +1\}$.

To exemplify the functionals, we consider a process in which threshold exceeding and its timing are of interest. Such processes are important, for example, in

apoptotic signaling and cell cycle progression, and allow for two outcomes. Either the concentration of a molecule $x_j^{(i)}$ within the $i$-th cell exceeds the threshold $x_{j,\mathrm{th}}$, $\delta^{(i)} = +1$, or it does not, $\delta^{(i)} = -1$. This yields the decision functional

$$F_\delta(x^{(i)}(\cdot)) := \begin{cases} +1 & \text{if } \max_t x_j^{(i)}(t) \geq x_{j,\mathrm{th}} \\ -1 & \text{otherwise.} \end{cases} \tag{7.3}$$

For the subgroup of cells exceeding the threshold, the time of threshold exceeding is defined by the second functional

$$F_\varphi(x^{(i)}(\cdot)) := \arg\min_t \left\{ x_j^{(i)}(t) \geq x_{j,\mathrm{th}} \right\}, \tag{7.4}$$

and may be employed to achieve a quantitative understanding.

Note that the response $x^{(i)}(\cdot)$ of a cell merely depends on the cell's parameters $\theta^{(i)}$, as the single cell model is deterministic. Therefore, quantitative and qualitative properties of a single cell can be viewed as a function of parameters $\varphi^{(i)} = \varphi(\theta^{(i)})$ and $\delta^{(i)} = \delta(\theta^{(i)})$. Differences in the parameters—as they arise between different cells—may hence influence $\delta^{(i)}$ and $\varphi^{(i)}$, which determine cell fate decision and qualitative properties of the cells.

### Response Markers

To understand the heterogeneity within the population response $\Sigma_{\mathrm{pop}}$, it is necessary to assess the dependency of $\delta^{(i)}$ and $\varphi^{(i)}$ on the individual parameters $\theta_j$. In particular, the question arises which subset $\theta_{\boldsymbol{m}}$ of parameters,

$$\theta_{\boldsymbol{m}} := [\theta_{m_1}, \ldots, \theta_{m_r}]^{\mathrm{T}}, \quad \text{with } \boldsymbol{m} \subseteq \{1, \ldots, q\},$$

is responsible for which aspect of the population heterogeneity. Mathematically, $\boldsymbol{m}$ is an index set and, e.g., for $\boldsymbol{m} = [2, 4]^{\mathrm{T}}$ only $\theta_{\boldsymbol{m}} = [\theta_2, \theta_4]^{\mathrm{T}}$ is considered. The question of the relative importance of different parameters directly relates to the common problem of biomarker selection for stem cells and tumor cells, which is experimentally challenging.

If there exists a subset $\theta_{\boldsymbol{m}}$ of the parameters $\theta$ which allows for the reliable prediction of the response, not all sources for heterogeneity have to be assessed but only those associated to $\theta_{\boldsymbol{m}}$. This allows one to focus on model development, as well as the reduction of the experimental effort.

## 7.4.2 Analysis of Population Models Using Data Analysis Tools

We illustrate the application of parallel-coordinates plots and SVMs for the study of parameter dependencies and the selection of markers $\boldsymbol{m}$. First, the cell

ensemble is simulated for $N \gg 1$. This yields many pairs of parameters and trajectories,

$$\left(\theta^{(i)}, x^{(i)}(\cdot)\right), \quad i = 1, \ldots, N,$$

which are then used to obtain samples of quantitative,

$$\mathcal{S}_\varphi = \left\{\left(\theta^{(1)}, \varphi^{(1)}\right), \ldots, \left(\theta^{(N)}, \varphi^{(N)}\right)\right\}, \quad \text{with } \varphi^{(i)} = F_\varphi(x^{(i)}(\cdot)),$$

and qualitative

$$\mathcal{S}_\delta = \left\{\left(\theta^{(1)}, \delta^{(1)}\right), \ldots, \left(\theta^{(N)}, \delta^{(N)}\right)\right\}, \quad \text{with } \delta^{(i)} = F_\delta(x^{(i)}(\cdot))$$

cell properties. These samples contain information about the dependency of $\varphi$ and $\delta$ on $\theta$. To study the high-dimensional mappings $\delta = \delta(\theta)$ and $\varphi = \varphi(\theta)$, parallel-coordinates plots will be employed. For the quantitative assessment of particular marker combinations SVMs will be applied. By combining both approaches, it is possible to quickly gain an overview of important interrelations and to quantify those.

### Combining Parallel-Coordinates Plots and SVMs to a Visual Analytics System

The proposed simulation-based analysis circumvents a purely analytical analysis of the system equations, which would be time consuming and could only be carried out by experts. However, the simulation-based approach creates the need for analyzing the large, high-dimensional datasets $\mathcal{S}_\delta$ and $\mathcal{S}_\varphi$.

The analysis of such datasets often relies on a reduction of complexity while preserving the important information. Visualization can help in such a situation to determine the important parameters and to avoid information loss. In this work, parallel-coordinates plots are used to gain insight into the high-dimensional dependencies and to find interesting dimensions. In this particular setting, interesting dimensions are those that clearly separate a given set of classes and thus are good candidates for the selection of potential markers $\boldsymbol{m}$, which are then used to train a SVM. The resulting SVMs allow for a quantitative evaluation of marker quality. By combining SVMs and parallel-coordinates plots, the number of evaluations of SVMs can be substantially decreased, resulting in reduced computational complexity. The overall workflow of the analysis illustrated in Figure 7.13.

Besides an improved understanding of the model, results obtained during the analysis can be used to adapt the population model or to select additional experiments. This framework thus incorporates important aspects of the combination of visualization and data-mining, as was discussed in Chapter 1.

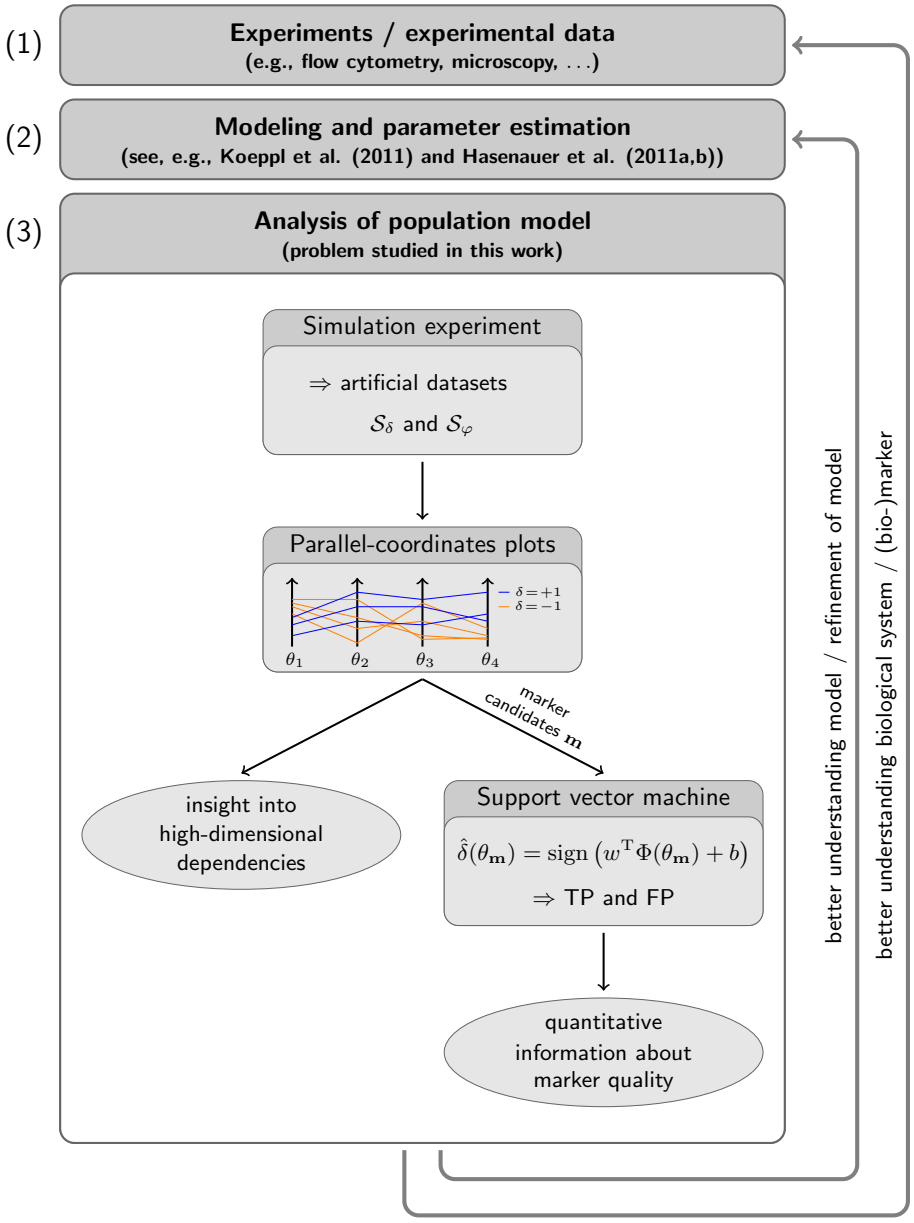



**Figure 7.13:** Illustration of the overall workflow of (1) experiments and the collection of measurement data, (2) modeling and parameter estimation, and (3) model analysis. Based on simulation data generated from the model, a visual analysis is performed using parallel-coordinates plots. Employing this visualization, insight can be gained into the dependencies of the considered properties on the parameters.

### 7.4.3 Proapoptotic Signaling

Our approach is illustrated on a model of proapoptotic signaling, which is involved in the process of apoptosis, an important physiological process to remove infected, malfunctioning, or no longer needed cells from a multicellular organism. The apoptotic signaling pathways converge at the caspase cascade, where initiator caspases (e.g., caspase 8 (C8)) and effector caspases (e.g., caspase 3 (C3)) are activated. If the activity of effector caspases exceeds a certain threshold, apoptosis is induced.

A variety of single-cell and cell-population models have been proposed to describe cellular apoptosis (see (Hasenauer et al., 2012, and references therein)). In this case study, we consider the signal-transduction model of Eissing et al. (2004) which is illustrated in Figure 7.12. As the process of apoptosis induction is known to be heterogeneous, we extend the common single-cell model to account for cell-to-cell variability. This is achieved by introducing differences in parameter values and initial conditions:

- The amount of C8, C3, caspases 8- and 10-associated RING protein (CARP), and inhibitor of apoptosis protein (IAP) is known to be different among individual cells from flow cytometric experiments. This variability is modeled with different synthesis rates ($k_{-8}$, $k_{-9}$, $k_{-10}$, and $k_{-12}$) among individual cells. The distribution of $k_{-8}$, $k_{-9}$, $k_{-10}$, and $k_{-12}$ within the population is modeled as log-normal distribution, with mean as published by Eissing et al. (2004) and a coefficient of variation of 0.4 (Hasenauer et al., 2012). The initial conditions of C8, C3, CARP, and IAP are set to their steady state values.
- Similar to the original publication (Eissing et al., 2004), the activation of the caspase cascade is modeled by a non-zero initial condition of active caspase 8 (C8a(0)). In the population, C8a(0) is log-normally distributed with a median of 4 000 molecules per cells and a coefficient of variation of 0.4. The variation of C8a(0) accounts for variability up-stream of the caspase cascade.

Binding affinities and kinetic rates are the same for all cells. For the numerical values, we refer to the article of Eissing et al. (2004).

Given this model of the heterogeneous cell population, we analyzed (i) how the decision whether or not a cell undergoes apoptosis during the first 12 hours and (ii) how the time of cell death $T_d$ is influenced by parameters

$$\theta = \left[\mathrm{C8a(0)}, k_{-8}, k_{-9}, k_{-10}, k_{-12}\right]^{\mathrm{T}}.$$

This yields two variables of interest: $\delta$ (= $+1$ $\Rightarrow$ cell survived; = $-1$ $\Rightarrow$ cell died) providing the outcome of the decision process; and $\varphi$ (= $T_d$) providing the time of apoptosis commitment. As indicator for apoptosis, the amount of

active caspase 3 (C3a) is used. If more than 5,000 copies of C3a are present in a cell, this cell is assumed to undergo apoptosis within 10 minutes, defining the time of cell death $T_d$. The functionals associated to the considered $\delta$ and $\varphi$ are similar to (7.3) and (7.4), respectively. In the remainder, we search for a lower-dimensional subset of the parameters $\theta$ which provide good markers for cell death and survival as well as the time of cell death.

To study the life-death-decision, a sample $\mathcal{S}_\delta$ with 100 000 members is visualized in parallel coordinates (Figure 7.14). As only two classes (dead and alive) are considered, alpha blending can be used to visualize the density of each class as well as the density at the overlapping regions, where the transparent red color, representing dead cells, and the transparent blue color, representing living cells, are blended with $\alpha = 0.03$.

From Figure 7.14, it is apparent that the second and fourth parameters ($\theta_{\boldsymbol{m}} = [k_{-8}, k_{-10}]^{\mathrm{T}}$) provide a reasonable separation between the classes (red = dead, blue = alive). Most of the surviving cells have high values of $k_{-8}$ and low values of $k_{-10}$, which corresponds to a high IAP and low C3 expression. The influence of the other parameters seems to be minor.

Given the results of the visual analysis, we consider $\theta_{\boldsymbol{m}} = k_{-8}$, $\theta_{\boldsymbol{m}} = k_{-10}$, as well as $\theta_{\boldsymbol{m}} = [k_{-8}, k_{-10}]^{\mathrm{T}}$ and compute the classification quality using SVMs. The results (see (Hasenauer et al., 2012)) show that the predictive power of the individual parameters is limited while both markers together yield a reasonable classification performance.

After analyzing the decision process, we study the dependency of time of cell death $T_d$ on the parameters. The time of cell death $T_d$ is a quantitative property and can take any positive value, therefore an alternative visualization has to be used. One approach would be to add an axis for $T_d$ and apply a different color for each line in parallel coordinates, similar to the approach presented in Section 7.3. Due to the large number of lines, however, this approach is not suitable here. Instead, we split the data into three classes and created separate plots for each class.

Figure 7.15 visualizes the parameter distribution in different percentile intervals for $T_d$. A comparison of Figure 7.15A, showing the cells that die early (0 to 10th percentile), and Figure 7.15C, depicting the cells that die late (90 to 100th percentile), unravels offsets in all parameter dimensions. The differences are particularly prominent for C8a(0), $k_{-10}$, and $k_{-12}$, showing that the abundance of C3 also plays an important role in determining whether cells die early or late. Unfortunately, a closer look at Figure 7.15 also reveals that the parameter distributions associated to cells that undergo apoptosis at early, intermediate, and late points in time strongly overlap in parallel coordinates. This indicates that $T_d$ may depend on all parameters. Therefore, a reliable prediction of $T_d$ using only a few parameters might be infeasible. A quantification of the predictive

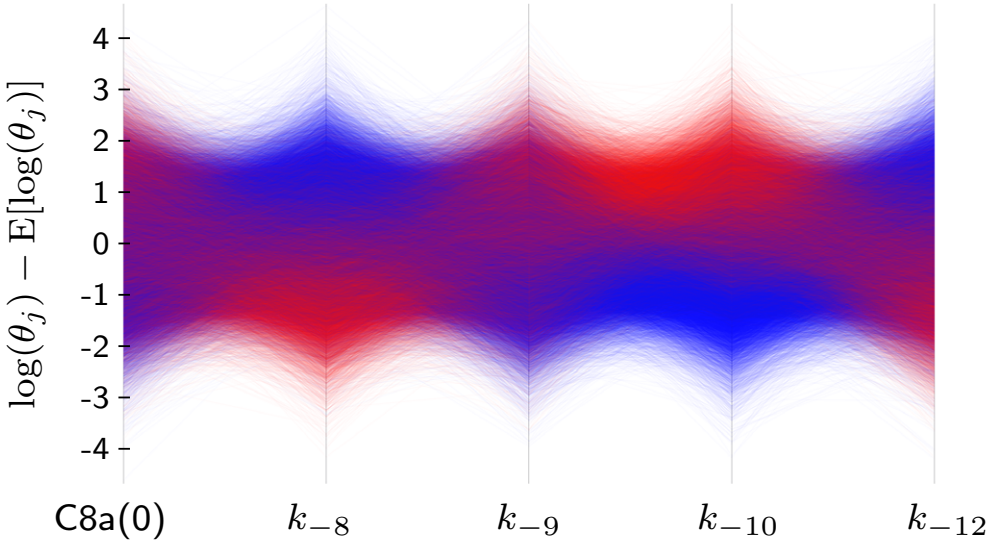



**Figure 7.14:** Density-based parallel coordinates with polylines representing the parameter of a single cell. Color encodes whether the cell survived (blue) or died (red). In order to emphasize dense regions, alpha blending with $\alpha = 0.03$ was used for all lines. The parameters $k_{-8}$ and $k_{-10}$ show the best separation of colors and hence correspond to potential markers.

power of different marker combinations with respect to $T_d$ using support vector regression is given in (Hasenauer et al., 2012).

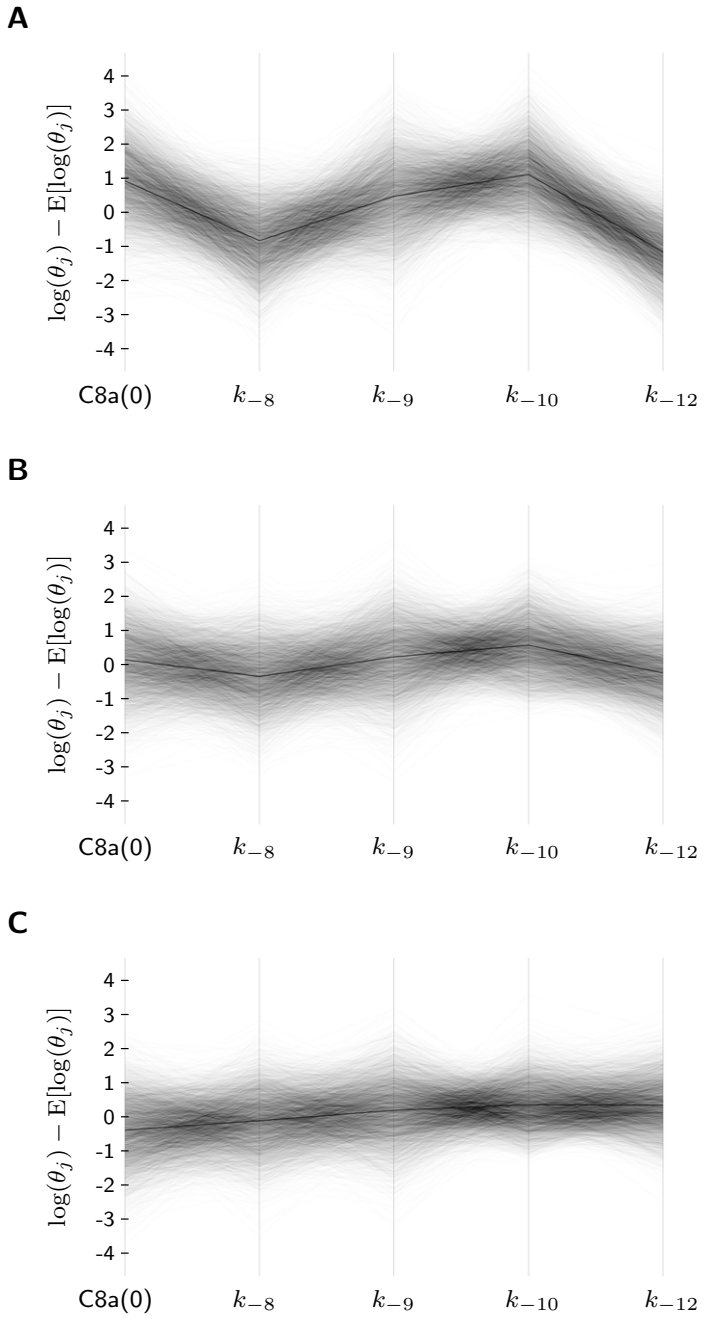



**Figure 7.15:** Visualization of three subsets of the sample $\mathcal{S}_\varphi$ in parallel coordinates. The subplots depict the parameter vectors of the cells with $T_d^{(i)}$: (A) below the 10th percentile; (B) between 45th percentile and the 55th percentile; and (C) above the 90th percentile of all $T_d$ values. The mean is emphasized (thick black line).

CHAPTER 8

# Conclusion

This thesis presented visualization techniques for parallel coordinates to overcome existing limitations with respect to classification, overplotting, continuous input data, scalability, line-tracing, clustering, and axis-ordering.

The second chapter introduced parallel coordinates and presented an overview of the research previously conducted in modeling, creating, understanding, and interacting with parallel coordinates. The taxonomy introduced in Chapter 2 captures the variety of research directions covered so far and enables the identification of "white spots" that require further research. Where applicable, rendering and interaction techniques were related to existing knowledge discovery tasks. The major challenges for parallel-coordinate visualizations were identified and existing approaches to address them were discussed.

Chapter 3 presented continuous parallel coordinates for multidimensional multivariate data. The construction of such a high-dimensional density field relies on the concept of two-dimensional continuous scatterplots that are mapped to the parallel-coordinates system using point–line duality. The mathematical density model was derived based on mass conservation during the mapping from the data to the parallel-coordinates domain. The consecutive application of this mapping allows for an arbitrary number of data dimensions. Different numerical integration techniques for the computation of the density model were presented. It was shown that both gathering and scattering techniques can be used for the approximation of density in parallel coordinates. For triangulated data, an analytic solution was provided.

An important benefit of continuous parallel coordinates is that typical sampling artifacts are avoided that otherwise occur in discrete parallel coordinates. Distracting visual patterns are removed that are not related to patterns in the

data, but emerge from the dependency of discrete parallel coordinates on the sampling rate in the data domain. In contrast, continuous parallel coordinates are largely independent of the resolution: plots generated from low-resolution data are very similar to the full-resolution version. However, the accuracy of the plots from coarsened data depends on the interpolation function used in the reconstruction step.

This behavior demonstrates the fundamental aggregation character of density-based parallel coordinates. Like other statistical visualization techniques, such as histograms, this approach is robust under sampling effects and other external influences, capturing the essence of a dataset. It is important to note that although sparse data probably benefits most from this method, sampling artifacts can also occur from high-resolution data. These sampling artifacts are guaranteed to be removed by continuous parallel coordinates. Another practical advantage of continuous parallel coordinates is the scalability with increasing data set size: the overplotting problem is avoided without the need for parameters such as bucket size or any other density approximation technique.

Apart from differences regarding the sampling of the data, however, continuous parallel coordinates share most of the advantages and problems of discrete parallel coordinates. Many of the improvements and extensions to parallel coordinates presented in the other chapters can thus be applied to continuous parallel coordinates. For instance, the PCM can be used in conjunction with continuous parallel coordinates in order to show multiple axis orders. In principle, interactive techniques such as brushing are also applicable to continuous parallel coordinates. Smooth brushing (Doleisch and Hauser, 2002) is particularly interesting for continuous data representations because a density gradient can directly be obtained from the plots. However, methods depending on individual lines such as angular brushing (Hauser et al., 2002) cannot be used directly.

In the limit process, continuous parallel coordinates share the same visual signature with classic density plots, where the characteristics of parallel coordinates are fully captured but single lines cannot be perceived. Using brushing, however, the line structure of discrete parallel coordinates can be reconstructed in a controlled manner by sampling the continuous version.

As the computation of continuous parallel coordinates is time consuming, a splatting algorithm for constructing continuous bivariate scatterplots and multivariate parallel-coordinates plots was presented in Chapter 4. The core element of the approach is the analytic transformation of a 3D Gaussian kernel from the spatio-temporal domain to the scatterplot and parallel-coordinates domains, respectively. Discretized versions of these splats are pre-computed and stored in textures. During runtime, a progressive rendering algorithm subsequently adds more and more splats with decreasing variance to the plot, improving image quality. The main advantage of progressive rendering is that previews of the exact plot are available extremely fast, ideally supporting interactive data

analysis especially for large data sets. Another advantage of splatting is that image quality can be gradually balanced with computation speed by adjusting the number of splats and splat size.

Although the algorithm presented was restricted to Gaussian kernels in the spatio-temporal domain, the rendering approach is not necessarily limited to this kernel type. Other kernels would just require a different pre-computation of footprints. An important open question is how the minimum number of splats can be determined beforehand that would guarantee a certain plot quality.

The notion of a line density for parallel coordinates was introduced by Miller and Wegman (1991). However, their model only supports normal and uniform distributions in the data domain, while the model presented in this thesis is more general and provides a closed-form solution for any density distribution in a 2D scatterplot. Continuous parallel coordinates have been adopted in the visualization community and have been used to visualize uncertainty (Feng et al., 2010), to extract features (Lehmann and Theisel, 2011), or to define multidimensional transfer functions (Guo et al., 2011b). The integration of statistical plots in systems for the visualization and analysis of spatial data such as CT scans or flow simulations (Doleisch et al., 2003) is an ongoing development in the research community that has implications for industrial applications. While there are several papers using parallel coordinates to visualize such diverse data as facial dynamics (Tam et al., 2011) or hurricanes (Steed et al., 2009), it will take time and effort to incorporate a continuous model into those systems. This is partly due to a visual encoding that is even more "unusual" than traditional parallel coordinates, and partly due to implementation issues regarding the computational complexity of the model. Finally, the success of continuous statistical plots in general largely depends on the success of continuous scatterplots (Bachthaler and Weiskopf, 2008).

Bundled parallel-coordinates plots, presented in Chapter 5, are designed to alleviate some limitations of traditional polyline plots and to geometrically reveal cluster structures specified for the input data. Bundled curve plots are constructed from piecewise cubic Bézier curves with control points judiciously selected to ensure $C^1$ continuity; this can alleviate the well-known cross-over problems for polyline plots without any additional visual aid beyond curve geometry. Cluster structures in the data are emphasized by curve bundling, pulling curves belonging to the same cluster toward their cluster centroid. The greatest advantage of bundling is that a wide range of views from abstract high-level to detailed low-level representations are easily obtained by tuning the bundling strength. For large datasets, density plots for each cluster can be employed to avoid the potential over-plotting problem. As traditional parallel coordinates are a special case of bundled parallel coordinates, many established extensions and interaction techniques such as brushing-and-linking can be applied without further enhancements. The user study presented in this work supports the following conclusions: Firstly, curve bundling is effective in displaying clustering

information purely based on geometry. As a consequence, the color channel can be used for other attributes or, in the case of complex cluster patterns, be used in concert with color. Secondly, with a properly chosen bundling strength, bundled curve plots retain the same strength as polyline plots in revealing correlations between visualized variables. Hence one of the core aspects of analysis using parallel coordinates carries over using bundling.

The PCM presented in Chapter 6 is a new visualization for multivariate data representing all pairwise relations of dimensions without redundancy. The PCM fulfills three constraints: (1) every pair of dimensions is rendered exactly once, (2) every parallel-coordinates plot contains all dimensions at least once, and (3) every parallel-coordinates plot shows the same number of dimensions. The PCM does not solve the axis-order problem in general, but it represents the counterpart of parallel coordinates to the widely used SPLOM. In contrast to the SPLOM, less screen space is required for the PCM for the same information to be presented. However, the layout of 2D parallel-coordinates plots in the PCM does not allow for easy navigation to a given pair of variables as in the SPLOM. With one of the main applications of parallel coordinates being the exploratory data analysis where new hypotheses are identified by browsing patterns instead of asking specific questions, addressing a particular bivariate plot might not be required in the initial steps of an analysis.

The mathematical models and visualization algorithms described in the individual chapters are versatile: they are independent of a specific application domain or dataset and can be used for many tasks such as illustration, data-mining, knowledge-discovery, aesthetic visualization, and many others that involve parallel coordinates. Beyond that, the theoretical foundations behind continuous statistical plots, progressive refinement, bundling, and the matrix layout for ordering dimensions are extensible to many other visualizations such as star plots or radviz (Hoffman et al., 1997). The respective research questions that have been addressed in this thesis coincide with established challenges in scientific visualization (Hibbard, 1999; Johnson, 2004), information visualization (Chen, 2005), and visual analytics (Keim et al., 2006). The mathematical model of continuous parallel coordinates (Chapter 3) is an important step toward the *integration of scientific and information visualization* (Johnson, 2004), in particular for *multi-field* and *time-dependent* visualization (Johnson, 2004). Based on the underlying continuous model, splatting and progressive refinement (Chapter 4) provide *scalability* (Chen, 2005; Keim et al., 2006) and the foundations for visualizing the *dynamics* (Chen, 2005) of processes, e.g. in the form of *streaming data* (Keim et al., 2006). Bundling (Chapter 5) and the parallel-coordinates matrix (Chapter 6) address the challenge to show overview and detail (Shneiderman, 1996) in order to visualize both *local and global perspectives* (Johnson, 2004) on the data. Finally, the geometric mappings and rendering algorithms provided for parallel coordinates and scatterplots are reproducible and can be implemented in existing visualization toolkits, such as R (R Core Team, 2012), VTK (Schroeder

et al., 2006), D3 (Bostock et al., 2011), etc. The practical implications are apparent for many applications, as was demonstrated in Chapter 7.

In the future, some of the ideas presented here could also be applied to other visualization techniques. For instance, the continuous rendering model including progressive refinement could be adopted for other point and line-based visualizations, such as radviz (Hoffman et al., 1997), stardinates, or graphs. The layout algorithm used in the PCM can be applied to any visualization of bivariate data such as scatterplots or adjacency matrices (for undirected graphs). There are many questions that remain to be answered for traditional parallel coordinates. For example, it has not been investigated yet in which order to render lines and to what extent the visibility of patterns depend on it. The inter-axis distance is a free parameter that has no impact on the theoretical properties of a parallel-coordinates plot. In practice, however, the rasterization of lines may result in undesirable distortions of the patterns contained in the data. From a perception point-of-view, it is desirable to use angles close to 45 degrees, as the precision in estimating angles is best for angles in this range. This opens up the question how to measure the angle for more than one line and how to handle more than one pair of axes in a single parallel-coordinates plot. Line-based and density-based parallel coordinates have to be evaluated with respect to ease of interpretation and usability for real-world tasks. Unlike Cartesian coordinates, the patterns emerging in parallel coordinates are not being taught in school and are thus very unfamiliar to most people, analysts, and even researchers.

Inselberg extended the point–line duality to a more general point–hyperplane duality. For instance, a plane in three dimensions can be represented by the coordinate system itself (defined by the axis-distance) and two points. This can further be extended to $p$-flats in $N$-dimensional space. However, most of the research conducted in the visualization community use parallel coordinates to visualize 2D projections of points. Hence, to "see" higher-order structures in the data with the techniques presented here, their adaption to higher-dimensional spaces has to be investigated.

APPENDIX

# Line Density

This chapter provides the derivation of Equation (3.6), the line density of a point $\bar{\ell} = (x, y)$ in parallel coordinates:

$$\varphi(x, y) = \int_{\ell} \frac{\sigma(\boldsymbol{\ell}(\lambda))}{\|\tilde{\boldsymbol{n}}\|} \mathrm{d}\lambda.$$

Assuming mass conservation, the mass $M$ of the interval $\Omega$ in the parallel-coordinates domain and the area $\Phi$ in the data domain must be equal (see Figure 3.1, page 3.1):

$$M = \int_{\Omega} \varphi(x, y) \mathrm{d}y = \int_{\Phi} \sigma(\boldsymbol{x}) \mathrm{d}^2 x. \tag{A.1}$$

Applying the fundamental theorem of calculus yields

$$\varphi(x, y) = \frac{\mathrm{d}M}{\mathrm{d}y} = \frac{\mathrm{d}}{\mathrm{d}y} \int_{\Phi} \sigma(\boldsymbol{x}) \mathrm{d}^2 x. \tag{A.2}$$

Now, the integration domain $\Phi$ is split in two perpendicular directions $\Phi_{\parallel}$ and $\Phi_{\perp}$.

For this purpose a rotation $\nu \colon \mathbb{R}^2 \longrightarrow \mathbb{R}^2,\ \boldsymbol{x} \mapsto \nu(\boldsymbol{x})$ is defined that maps the unit vector $\boldsymbol{x}_2$ to $\boldsymbol{n} = \frac{\tilde{\boldsymbol{n}}}{\|\tilde{\boldsymbol{n}}\|}$:

$$\nu(\boldsymbol{x}_2) = \tilde{\boldsymbol{x}}_2 = \boldsymbol{n}. \tag{A.3}$$

Now, the transformation theorem for integrals can be applied to (A.1):

$$\int_{\nu(\Phi)} \sigma(\tilde{\boldsymbol{x}}) \mathrm{d}^2\tilde{x} = \int_{\Phi} \sigma(\nu(\boldsymbol{x})) \left|\det(D\nu(\boldsymbol{x}))\right| \mathrm{d}^2 x \tag{A.4}$$

where $D$ denotes the respective Jacobian matrix. Note that $\left|\det(D\nu(\boldsymbol{x}))\right| = 1$. Now, splitting the region $\Phi = \Phi_{\parallel} \otimes \Phi_{\perp}$ remains only a matter of splitting integrals:

$$M = \int_{\Phi_{\parallel}} \left[ \int_{\Phi_{\perp}} \sigma(\tilde{\boldsymbol{x}}) \mathrm{d}\tilde{x}_2 \right] \mathrm{d}\tilde{x}_1. \tag{A.5}$$

For the computation of the density follows:

$$\varphi(x, y) = \int_{\Phi_{\parallel}} \left[ \frac{\mathrm{d}}{\mathrm{d}y} \int_{\Phi_{\perp}} \sigma(\tilde{\boldsymbol{x}}) \mathrm{d}\tilde{x}_2 \right] \mathrm{d}\tilde{x}_1. \tag{A.6}$$

In order to transform the integration along $\Phi_{\perp}$ to an integration over $\Omega$, we use that

$$\frac{\mathrm{d}D_{\boldsymbol{x}}}{\mathrm{d}y} = \frac{1}{\|\tilde{\boldsymbol{n}}\|} \tag{A.7}$$

as a result of (3.3). Then, the inner integral of (A.5) yields the desired transformation to the parallel-coordinates domain:

$$\int_{\Phi_{\perp}} \sigma(\tilde{\boldsymbol{x}}) \mathrm{d}\tilde{x}_2 = \int_{\Omega} \frac{\sigma(\tilde{x}_1, D_{\boldsymbol{x}}(y))}{\|\tilde{\boldsymbol{n}}\|} \mathrm{d}y. \tag{A.8}$$

With (A.6), the density in the parallel-coordinates domain then becomes:

$$\varphi(x, y) = \int_{\Phi_{\parallel}} \frac{\sigma(\tilde{x}_1, D_{\boldsymbol{x}}(y))}{\|\tilde{\boldsymbol{n}}\|} \mathrm{d}\tilde{x}_1. \tag{A.9}$$

Returning to the original coordinate system finally describes the line density in a point $\bar{\ell} = (x, y)$ of the parallel-coordinates system by integrating over the corresponding line in the data domain:

$$\varphi(x, y) = \int_{\ell} \frac{\sigma(\boldsymbol{\ell}(\lambda))}{\|\tilde{\boldsymbol{n}}\|} \mathrm{d}\lambda \tag{A.10}$$

with $\boldsymbol{\ell}(\lambda)$ being the arc-length parametrized line $\ell$.

APPENDIX B

# Footprint Computation

This appendix provides the derivation of Equation (4.3)

$$\sigma(\boldsymbol{x}) = \frac{\sqrt{\pi}}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} e^{-(\boldsymbol{x}-\boldsymbol{x}_0)^{\mathrm{T}} E (\boldsymbol{x}-\boldsymbol{x}_0)}$$

for the computation of the density $\sigma(\boldsymbol{x})$ at a point $\boldsymbol{x}$ in the data domain using a Gaussian reconstruction kernel in the spatio-temporal domain.

We first note that the density of a point $\boldsymbol{x}$ in the data domain is computed using the integration in Equation (4.2):

$$\sigma(\boldsymbol{x}) = \frac{1}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} \int_{\tau^{-1}(\boldsymbol{x})} e^{\frac{-\|\boldsymbol{r}-\boldsymbol{r}_0\|^2}{k^2}} \, \mathrm{d}r.$$

Without loss of generality, we assume that the spatio-temporal domain is unitless.

Since $x_1$ and $x_2$ both represent isosurfaces in the spatio-temporal domain that are planes, their intersection $\tau^{-1}(\boldsymbol{x})$ is a straight line that we write in parametric form as

$$\boldsymbol{l}(t) = \boldsymbol{p}' + t\boldsymbol{q}$$

with normalized tangent vector $\boldsymbol{q} = \frac{\nabla x_1 \times \nabla x_2}{\|\nabla x_1 \times \nabla x_2\|}$ and with

$$\boldsymbol{p}' = \boldsymbol{r}_0 + D_\tau^{-1}(\boldsymbol{x} - \boldsymbol{x}_0) \tag{B.1}$$

where $D_\tau$ denotes the Jacobian of $\tau$ evaluated at $\boldsymbol{r}_0$ and $D_\tau^{-1} = D_\tau^{\mathrm{T}}(D_\tau D_\tau^{\mathrm{T}})^{-1}$ is the right inverse of $D_\tau$.

Please note that Equation (B.1) also defines a plane spanned by the gradients $\nabla x_1$ and $\nabla x_2$ containing $\boldsymbol{r}_0$. Now, solving Equation (4.2) remains a matter of integrating a Gaussian over a line, which can be solved using a similar derivation as presented by Kniss et al. (2003). We set $\boldsymbol{p} = D_\tau^{-1}(\boldsymbol{x} - \boldsymbol{x}_0)$ and note that $\boldsymbol{q}$ is perpendicular to $\boldsymbol{p}$ such that $\boldsymbol{q}^{\mathrm{T}}\boldsymbol{p} = 0$. Also, $\boldsymbol{x}_0 = \tau(\boldsymbol{r}_0)$. Using the derivation given in (Kniss et al., 2003) (replacing $K$ with $D_\tau^{-1}$, $\boldsymbol{d}$ with $\boldsymbol{q}$, and $\boldsymbol{v}'_1$ with $\boldsymbol{p}$), we obtain:

$$\begin{aligned}
\sigma(\boldsymbol{x}) &= \frac{1}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} \int_{\tau^{-1}(\boldsymbol{x})} e^{-\frac{1}{k^2}(\boldsymbol{r}-\boldsymbol{r}_0)^{\mathrm{T}}(\boldsymbol{r}-\boldsymbol{r}_0)} \mathrm{d}^2 r \\
&= \frac{1}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} \int_{-\infty}^{\infty} e^{-\frac{1}{k^2}(\boldsymbol{p}'+t\boldsymbol{d}-\boldsymbol{r}_0)^{\mathrm{T}}(\boldsymbol{p}'+t\boldsymbol{d}-\boldsymbol{r}_0)} \mathrm{d}t \\
&= \frac{\sqrt{\pi}}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} e^{-\frac{1}{k^2}\boldsymbol{p}^{\mathrm{T}}\boldsymbol{p}} \\
&= \frac{\sqrt{\pi}}{\|\nabla \boldsymbol{x}_1 \times \nabla \boldsymbol{x}_2\|} e^{-(\boldsymbol{r}-\boldsymbol{r}_0)^{\mathrm{T}} E (\boldsymbol{r}-\boldsymbol{r}_0)}
\end{aligned} \tag{B.2}$$

with $E = \frac{1}{k^2}\left(D_\tau D_\tau^{\mathrm{T}}\right)^{-1}$.

# Bibliography


H. Albazzaz and X. Z. Wang. Historical data analysis based on plots of independent and parallel coordinates and statistical control limits. *Journal of Process Control*, 16(2):103–114, 2006. Cited on page 45.

G. Albuquerque, M. Eisemann, D. Lehmann, H. Theisel, and M. Magnor. Quality-based visualization matrices. In *Proceedings of the Vision, Modeling, and Visualization Workshop*, pages 341–349, 2009. Cited on pages 39, 40, and 101.

R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 111–117, 2005. Cited on page 8.

D. F. Andrews. Plots of high-dimensional data. *Biometrics*, 28(1):125–136, 1972. Cited on pages 6 and 24.

G. Andrienko and N. Andrienko. Constructing parallel coordinates plot for problem solving. In *Proceedings of the 1st International Symposium on Smart Graphics*, pages 9–14, 2001. Cited on pages 31, 34, and 45.

G. Andrienko and N. Andrienko. Parallel coordinates for exploring properties of subsets. In *Proceedings of the Second International Conference on Coordinated and Multiple Views in Exploratory Visualization*, pages 93–104, 2004. Cited on pages 25 and 37.

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*, pages 52–60, 1998. Cited on pages 40 and 101.

F. J. Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1): 17–21, 1973. Cited on pages 3 and 4.

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*, pages 81–88, 2004. Cited on page 26.

T. Avidan and S. Avidan. Parallax–A data mining tool based on parallel coordinates. *Computational Statistics*, 14(1):79–90, 1999. Cited on pages 14, 33, and 36.

S. B. Azhar and M. J. Rissanen. Evaluation of parallel coordinates for interactive alarm filtering. In *Proceedings of the 15th International Conference on Information Visualisation*, pages 102–109, 2011. Cited on pages 30 and 45.

S. Bachthaler and D. Weiskopf. Continuous scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1428–1435, 2008. Cited on pages 9, 28, 47, 48, 49, 51, 53, 62, 64, 70, 73, 75, and 135.

S. Bachthaler and D. Weiskopf. Efficient and adaptive rendering of 2-D continuous scatterplots. *Computer Graphics Forum*, 28(3):743–750, 2009. Cited on pages 9, 61, 64, 65, and 73.

N. Barlow and L. Stuart. Animator: A tool for the animation of parallel coordinates. In *Proceedings of the Eighth International Conference on Information Visualisation*, pages 725–730, 2004. Cited on page 43.

A. Barsky, T. Munzner, J. Gardy, and R. Kincaid. Cerebral: Visualizing multiple experimental conditions on a graph with biological context. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1253–1260, 2008. Cited on page 45.

O. M. Becker. Representing protein and peptide structures with parallel-coordinates. *Journal of Computational Chemistry*, 18(15):1893–1902, 1997. Cited on page 45.

R. A. Becker and W. S. Cleveland. Brushing scatterplots. *Technometrics*, 29(2): 127–142, 1987. Cited on page 31.

R. A. Becker, W. S. Cleveland, and A. R. Wilks. Dynamic graphics for data analysis. *Statistical Science*, 2(4):355–383, 1987. Cited on pages 31 and 40.

R. A. Becker, J. M. Chambers, and A. R. Wilks. *The New S Language: A Programming Environment for Data Analysis and Graphics*. Wadsworth & Brooks/Cole, Pacific Grove, CA, USA, 1988. Cited on page 38.

F. Bendix, R. Kosara, and H. Hauser. Parallel sets: Visual analysis of categorical data. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 133–140, 2005. Cited on page 25.

W. Berger, H. Piringer, P. Filzmoser, and E. Gröller. Uncertainty-aware exploration of continuous parameter spaces using multivariate prediction. *Computer Graphics Forum*, 30(3):911–920, 2011. Cited on page 45.

M. Berthold and L. Hall. Visualizing fuzzy points in parallel coordinates. *IEEE Transactions on Fuzzy Systems*, 11(3):369–374, 2003. Cited on page 25.

J. Blaas, C. Botha, and F. Post. Extensions of parallel coordinates for interactive exploration of large multi-timepoint data sets. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1436–1451, 2008. Cited on pages 45, 47, and 49.

M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011. Cited on page 137.

Carnegie Mellon University. CMU graphics lab motion capture database, 2013. [Online]. Available: http://mocap.cs.cmu.edu/, accessed 2013/02/12. Cited on pages 113 and 114.

H. Carr, D. Brian, and D. Brian. On histograms and isosurface statistics. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1259–1266, 2006. Cited on pages 9 and 49.

C. Chen. Top 10 unsolved information visualization problems. *IEEE Computer Graphics and Applications*, 25(4):12–16, 2005. Cited on pages 5 and 136.

C. Chen, W. Härdle, and A. Unwin, editors. *Handbook of Data Visualization*. Springer-Verlag, Berlin, Heidelberg, 2008. Cited on page 6.

H. Chen. Compound brushing. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 181–188, 2003. Cited on pages 31 and 33.

H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68(342):361–368, 1973. Cited on page 6.

H. Choi, H. Lee, and H. Kim. Fast detection and visualization of network attacks on parallel coordinates. *Computers & Security*, 28(5):276–288, 2009. Cited on page 46.

J. H. T. Claessen and J. J. van Wijk. Flexible linked axes for multivariate data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17 (12):2310–2316, 2011. Cited on pages 39 and 101.

W. S. Cleveland. *Visualizing Data*. Hobart Press, Summit, NJ, USA, 1993. Cited on page 6.

W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554, 1984. Cited on page 37.

W. S. Cleveland and R. McGill. Graphical perception: The visual decoding of quantitative information on graphical displays of data. *Journal of the Royal Statistical Society. Series A (General)*, 150(3):192–229, 1987. Cited on page 37.

C. Collins, F. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 91–98, 2009. Cited on page 28.

J. R. Comfort, T. R. Warner, E. P. Vargo, and E. J. Bass. Parallel coordinates plotting as a method in process control hazard identification. In *Proceedings of the IEEE Systems and Information Engineering Design Symposium*, pages 152–157, 2011. Cited on page 45.

M. Crider, S. Bergner, T. N. Smyth, T. Möller, M. K. Tory, A. E. Kirkpatrick, and D. Weiskopf. A mixing board interface for graphics and visualization applications. In *Proceedings of Graphics Interface*, pages 87–94, 2007. Cited on pages 33 and 45.

T. N. Dang, L. Wilkinson, and A. Anand. Stacking graphic elements to avoid over-plotting. *IEEE Transactions on Visualization and Computer Graphics*, 16 (6):1044–1052, 2010. Cited on pages 28 and 29.

A. Dasgupta and R. Kosara. Pargnostics: Screen-space metrics for parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 16(6): 1017–1026, 2010. Cited on pages 26, 30, 40, and 101.

A. Dasgupta, M. Chen, and R. Kosara. Conceptualizing visual uncertainty in parallel coordinates. *Computer Graphics Forum*, 31(3):1015–1024, 2012. Cited on page 43.

J. Dietzsch, J. Heinrich, K. Nieselt, and D. Bartz. SpRay: A visual analytics approach for gene expression data. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 179–186, 2009. Cited on pages 12, 14, 43, 45, and 117.

H. Doleisch and H. Hauser. Smooth brushing for focus+context visualization of simulation data in 3D. *Journal of WSCG*, 10(1):147–154, 2002. Cited on page 134.

H. Doleisch, M. Gasser, and H. Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proceedings of the Symposium on Data Visualisation*, pages 239–248, 2003. Cited on pages 47, 48, and 135.

H. Doleisch, P. Muigg, and H. Hauser. Interactive visual analysis of hurricane Isabel with SimVis. Technical Report TR-VRVis-2004-058, VRVis Research Center, 2004. Cited on pages 45, 110, and 113.

M. Dubska, A. Herout, and J. Havel. PClines–Line detection using parallel coordinates. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1489–1494, 2011. Cited on pages 22, 30, and 34.

R. Dunia, T. F. Edgar, and M. Nixon. Process monitoring using principal components in parallel coordinates. *AIChE Journal*, pages 1–12, 2012. Cited on pages 15 and 45.

R. M. Edsall. The parallel coordinate plot in action: Design and use for geographic visualization. *Computational Statistics & Data Analysis*, 43(4):605–619, 2003. Cited on page 45.

S. Eick and G. Wills. High interaction graphics. *European Journal of Operational Research*, 81(3):445–459, 1995. Cited on pages 4 and 9.

M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998. Cited on page 45.

T. Eissing, H. Conzelmann, E. D. Gilles, F. Allgöwer, E. Bullinger, and P. Scheurich. Bistability analyses of a caspase activation model for receptor-induced apoptosis. *Journal of Biological Chemistry*, 279(35):36892–36897, 2004. Cited on page 128.

G. Ellis and A. Dix. Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):717–724, 2006. Cited on pages 35 and 36.

G. Ellis and A. Dix. A taxonomy of clutter reduction for information visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1216–1223, 2007. Cited on page 36.

N. Elmqvist and J. D. Fekete. Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):439–454, 2010. Cited on pages 36 and 44.

N. Elmqvist, P. Dragicevic, and J. D. Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1539–1148, 2008. Cited on pages 101 and 105.

E. Fanea, S. Carpendale, and T. Isenberg. An interactive 3D integration of parallel coordinates and star glyphs. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 149–156, 2005. Cited on pages 28 and 29.

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37–54, 1996a. Cited on pages 2, 14, and 22.

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34, 1996b. Cited on pages 2 and 44.

J.-D. Fekete, J. van Wijk, J. Stasko, and C. North. The value of information visualization. In *Information Visualization*, volume 4950 of *Lecture Notes in Computer Science*, pages 1–18. Springer, Berlin, Heidelberg, 2008. Cited on page 9.

D. Feng, L. Kwock, Y. Lee, and R. M. Taylor II. Matching visual saliency to confidence in plots of uncertain data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):980–989, 2010. Cited on pages 19, 33, 66, and 135.

B. J. Ferdosi and J. B. T. Roerdink. Visualizing high-dimensional structures by dimension ordering and filtering using subspace analysis. *Computer Graphics Forum*, 30(3):1121–1130, 2011. Cited on pages 40 and 101.

M. Ferreira de Oliveira and H. Levkowitz. From visual data exploration to visual data mining: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):378–394, 2003. Cited on page 30.

M. A. Fisherkeller, J. H. Friedman, and J. W. Tukey. PRIM-9: An interactive multi-dimensional data display and analysis system. In *Proceedings of ACM Pacific*, pages 140–145, 1975. Cited on pages 4 and 31.

Y.-H. Fua, M. Ward, and E. Rundensteiner. Navigating hierarchies with structure-based brushes. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 58–64, 1999a. Cited on page 33.

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*, pages 43–50, 1999b. Cited on pages 25, 33, 34, 36, 37, and 42.

Y.-H. Fua, M. Ward, and E. Rundensteiner. Structure-based brushes: A mechanism for navigating hierarchically organized data and information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):150–159, 2000. Cited on page 33.

N. Gehlenborg and B. Wong. Points of view: Power of the plane. *Nature Methods*, 9(10):935–935, 2012. Cited on page 37.

Z. Geng, Z. Peng, R. Laramee, R. Walker, and J. Roberts. Angular histograms: Frequency-based visualizations for large, high dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2572–2580, 2011. Cited on pages 28 and 45.

A. Goel, C. Baker, C. Shaffer, B. Grossman, R. Haftka, W. Mason, and L. Watson. VizCraft: A multidimensional visualization tool for aircraft configuration design. In *Proceedings of the IEEE Conference on Visualization*, pages 425–555, 1999. Cited on page 45.

M. Graham and J. Kennedy. Using curves to enhance parallel coordinate visualisations. In *Proceedings of the Seventh International Conference on Information Visualisation*, pages 10–16, 2003. Cited on pages 24 and 79.

S. Grottel, G. Reina, and T. Ertl. Optimized data transfer for time-dependent, GPU-based glyphs. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 65–72, 2009. Cited on page 114.

S. Grottel, G. Reina, C. Dachsbacher, and T. Ertl. Coherent culling and shading for large molecular dynamics visualization. *Computer Graphics Forum*, 29(3): 953–962, 2010. Cited on page 114.

D. Guo. Coordinating computational and visual approaches for interactive feature selection and multivariate clustering. *Information Visualization*, 2(4): 232–246, 2003. Cited on page 40.

H. Guo, Z. Wang, B. Yu, H. Zhao, and X. Yuan. TripVista: Triple perspective visual trajectory analytics and its application on microscopic traffic data at a road intersection. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 163–170, 2011a. Cited on page 45.

H. Guo, H. Xiao, and X. Yuan. Multi-dimensional transfer function design based on flexible dimension projection embedded in parallel coordinates. In *Proceedings of the IEEE Pacific Visualization Symposium*, 2011b. Cited on pages 45, 49, and 135.

R. B. Haber and D. A. McNabb. Visualization idioms: A conceptual model for scientific visualization systems. In G. Nielson, B. Shriver, and L. Rosenblum, editors, *Visualization in Scientific Computing*, pages 74–93. IEEE Computer Society Press, Los Alamitos, CA, USA, 1990. Cited on pages 3 and 21.

J. Han and N. Cercone. RuleViz: A model for visualizing knowledge discovery process. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 244–253, 2000. Cited on page 25.

M. Harrower and C. Brewer. ColorBrewer.org: An online tool for selecting colour schemes for maps. *The Cartographic Journal*, 40(1):27–37, 2003. Cited on page 91.

J. Hartigan. Printer graphics for clustering. *Journal of Statistical Computation and Simulation*, 4(3):187–213, 1975. Cited on pages 6, 31, and 99.

J. Hasenauer, J. Heinrich, M. Doszczak, P. Scheurich, D. Weiskopf, and F. Allgöwer. Visualization methods and support vector machines as tools for determining markers in models of heterogeneous populations: Proapoptotic signaling as a case study. In *Proceedings of the Workshop on Computational Systems Biology*, pages 61–64, 2011a.

J. Hasenauer, S. Waldherr, M. Doszczak, N. Radde, P. Scheurich, and F. Allgöwer. Identification of models of heterogeneous cell populations from population snapshot data. *BMC Bioinformatics*, 12(125):1–15, 2011b. Cited on pages 123 and 124.

J. Hasenauer, S. Waldherr, M. Doszczak, N. Radde, P. Scheurich, and F. Allgower. Analysis of heterogeneous cell populations: a density-based modeling and identification framework. *Journal of Process Control*, 21(10):1417–1425, 2011c. Cited on page 123.

J. Hasenauer, J. Heinrich, M. Doszczak, P. Scheurich, D. Weiskopf, and F. Allgöwer. A visual analytics approach for models of heterogeneous cell populations. *EURASIP Journal on Bioinformatics and Systems Biology*, 2012(1):1–13, 2012. Cited on pages 12, 36, 45, 128, 129, and 130.

H. Hauser, F. Ledermann, and H. Doleisch. Angular brushing of extended parallel coordinates. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 127–130, 2002. Cited on pages 28, 33, and 134.

J. Heinrich and D. Weiskopf. Continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1531–1538, 2009. Cited on pages 9, 10, 11, 12, 15, 45, and 73.

J. Heinrich and D. Weiskopf. State of the art of parallel coordinates. In *STAR Proceedings of Eurographics 2013*, pages 95–116. Eurographics Association, 2013. Cited on page 10.

J. Heinrich, S. Bachthaler, and D. Weiskopf. Progressive splatting of continuous scatterplots and parallel coordinates. *Computer Graphics Forum*, 30(3):653–662, 2011a. Cited on pages 11, 12, and 19.

J. Heinrich, Y. Luo, A. E. Kirkpatrick, H. Zhang, and D. Weiskopf. Evaluation of a bundling technique for parallel coordinates. Technical Report Computer Science 2011/08, University of Stuttgart, Faculty of Computer Science, Electrical Engineering, and Information Technology, Germany, University of Stuttgart, Institute of Visualization and Interactive Systems, 2011b. Cited on pages 11, 12, 25, and 80.

J. Heinrich, Y. Luo, A. E. Kirkpatrick, 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*, pages 594–602, 2012a. Cited on pages 11, 12, 24, 25, 30, 36, 42, and 80.

J. Heinrich, J. Stasko, and D. Weiskopf. The parallel coordinates matrix. In *EuroVis–Short Papers*, pages 37–41, 2012b. Cited on pages 11, 12, and 39.

H. V. Henderson and P. F. Velleman. Building multiple regression models interactively. *Biometrics*, 37(2):391–411, 1981. Cited on page 7.

B. Hibbard. Top ten visualization problems. In *Proceedings of SIGGRAPH 99*, Annual Conference Series, pages 21–22, 1999. Cited on pages 5, 9, and 136.

P. Hoffman, G. Grinstein, K. Marx, I. Grosse, and E. Stanley. DNA visual and analytic data mining. In *Proceedings of the IEEE Conference on Visualization*, pages 437–441, 1997. Cited on pages 136 and 137.

D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006. Cited on pages 24, 79, and 81.

D. Holten and J. J. V. Wijk. Evaluation of cluster identification performance for different PCP variants. *Computer Graphics Forum*, 29(3):793–802, 2010. Cited on pages 22, 24, 26, 28, 30, 79, 80, 81, and 84.

C. B. Hurley. Clustering visualizations of multidimensional data. *Journal of Computational and Graphical Statistics*, 13(4):788–806, 2004. Cited on pages 8, 40, and 101.

C. B. Hurley and R. W. Oldford. Pairwise display of high-dimensional information via Eulerian tours and Hamiltonian decompositions. *Journal of Computational and Graphical Statistics*, 19:861–886, 2010. Cited on pages 39, 40, 101, 103, and 104.

A. Inselberg. N-dimensional graphics. Technical Report G320-2711, IBM, 1981. Cited on page 7.

A. Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(4): 69–91, 1985. Cited on pages 7, 8, 18, 25, and 38.

A. Inselberg. Multidimensional detective. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 100–107, 1997. Cited on page 8.

A. Inselberg. Conflict detection and planar resolution for air traffic control. In *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pages 1200–1205, 2001. Cited on pages 43 and 45.

A. Inselberg. *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. Springer, New York, 2009. Cited on pages 8, 13, 15, 16, 18, 19, 21, 22, 31, 36, 104, and 105.

A. Inselberg and T. Avidan. The automated multidimensional detective. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 112–119, 1999. Cited on page 40.

A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *Proceedings of the IEEE Conference on Visualization*, pages 361–378, 1990. Cited on page 18.

A. Inselberg, A. Chatterjee, and B. Dimsdale. System using parallel coordinates for automated line detection in noisy images. Patent, 1997, US 5631982, U.S. Classification: 382/168; 382/225; 382/281, International Classification: G06K 936; G06K 900; G06F 1714, Assignee: International Business Machines Corporation. Cited on pages 22, 30, and 34.

J. Johansson and M. Cooper. A screen space quality method for data abstraction. *Computer Graphics Forum*, 27(3):1039–1046, 2008. Cited on page 30.

J. Johansson, M. Cooper, and M. Jern. 3-dimensional display for clustered multi-relational parallel coordinates. In *Proceedings of the Ninth International Conference on Information Visualisation*, pages 188–193, 2005a. Cited on pages 28 and 101.

J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 125–132, 2005b. Cited on page 26.

J. Johansson, P. Ljung, and M. Cooper. Depth cues and density in temporal parallel coordinates. In *Proceedings of the Eurographics/IEEE-VGTC Symposium on Visualization*, pages 35–42, 2007. Cited on pages 25, 43, 56, 57, 59, and 62.

J. Johansson, C. Forsell, M. Lind, and M. Cooper. Perceiving patterns in parallel coordinates: Determining thresholds for identification of relationships. *Information Visualization*, 7(2):152–162, 2008. Cited on page 30.

S. Johansson, K. Knaving, A. Lane, M. Jern, and J. Johansson. Interactive exploration of ingredient mixtures using multiple coordinated views. In *Proceedings of the 13th International Conference on Information Visualisation*, pages 210–218, 2009. Cited on page 40.

C. Johnson. Top scientific visualization research problems. *IEEE Computer Graphics and Applications*, 24(4):13–17, 2004. Cited on pages 5, 9, and 136.

D. Keefe, M. Ewert, W. Ribarsky, and R. Chang. Interactive coordinated multiple-view visualization of biomechanical motion data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1383–1390, 2009. Cited on page 45.

D. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002. Cited on pages 5 and 8.

D. Keim and H.-P. Kriegel. Visualization techniques for mining large databases: A comparison. *IEEE Transactions on Knowledge and Data Engineering*, 8(6): 923–938, 1996. Cited on page 6.

D. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Challenges in visual data analysis. In *Proceedings of the Tenth International Conference on Information Visualisation*, pages 9–16, 2006. Cited on pages 2, 5, and 136.

D. A. Keim, F. Mansmann, D. Oelke, and H. Ziegler. Visual analytics: Combining automated discovery with interactive visualizations. In *Discovery Science*, number 5255 in Lecture Notes in Computer Science, pages 2–14. Springer, Berlin Heidelberg, 2008. Cited on page 4.

H. Kim, I. Lee, J. Cho, and J. Moon. Visualization of network components for attack analysis. In *Proceedings of the IEEE Symposium on Computational Intelligence in Cyber Security*, pages 1–8, 2009. Cited on page 46.

J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002. Cited on pages 49 and 75.

J. Kniss, S. Premoze, M. Ikits, A. Lefohn, C. Hansen, and E. Praun. Gaussian transfer functions for multi-field volume visualization. In *Proceedings of the IEEE Conference on Visualization*, pages 497–504, 2003. Cited on page 142.

R. Kosara, F. Bendix, and H. Hauser. Parallel sets: Interactive exploration and visual analysis of categorical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):558–568, 2006. Cited on page 25.

X. Kuang, H. Zhang, S. Zhao, and M. J. McGuffin. Tracing tuples across dimensions: A comparison of scatterplots and parallel coordinate plots. *Computer Graphics Forum*, 31(3):1365–1374, 2012. Cited on pages 8 and 30.

M. Lanzenberger, S. Miksch, and M. Pohl. Exploring highly structured data: A comparative study of stardinates and parallel coordinates. In *Proceedings of the Ninth International Conference on Information Visualisation*, pages 312–320, 2005. Cited on page 30.

D. Laur and P. Hanrahan. Hierarchical splatting: A progressive refinement algorithm for volume rendering. *Computer Graphics*, 25(4):285–288, 1991. Cited on page 67.

H.-Y. Lee, H.-L. Ong, E.-W. Toh, and S.-K. Chan. A multi-dimensional data visualization tool for knowledge discovery in databases. In *Proceedings of the Nineteenth Annual International Computer Software and Applications Conference*, pages 26–31, 1995. Cited on page 8.

D. Lehmann and H. Theisel. Discontinuities in continuous scatter plots. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1291–1300, 2010. Cited on page 49.

D. Lehmann and H. Theisel. Features in continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1912–1921, 2011. Cited on pages 30, 45, and 135.

J. Li, J. van Wijk, and J.-B. Martens. Evaluation of symbol contrast in scatterplots. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 97–104, 2009. Cited on pages 80, 81, 89, and 93.

J. Li, J.-B. Martens, and J. J. van Wijk. Judging correlation from scatterplots and parallel coordinate plots. *Information Visualization*, 9(1):13–30, 2010. Cited on pages 30 and 80.

M. Lind, J. Johansson, and M. Cooper. Many-to-many relational parallel coordinates displays. In *Proceedings of the 13th International Conference on Information Visualisation*, pages 25–31, 2009. Cited on pages 28, 40, and 101.

R. Maciejewski, I. Woo, W. Chen, and D. S. Ebert. Structuring feature space: A non-parametric method for volumetric transfer function generation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1473–1480, 2009. Cited on page 49.

A. R. Martin and M. O. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proceedings of the IEEE Conference on Visualization*, pages 271–278, 1995. Cited on pages 31, 33, and 36.

K. Matkovic, M. Jelovic, J. Juric, Z. Konyha, and D. Gracanin. Interactive visual analysis and exploration of injection systems simulations. In *Proceedings of the IEEE Conference on Visualization*, pages 391–398, 2005. Cited on page 45.

Maurice d'Ocagne. *Coordonnées Parallèles et Axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèlles*. Gauthier-Villars, Paris, France, 1885. Cited on page 38.

K. McDonnell and K. Mueller. Illustrative parallel coordinates. *Computer Graphics Forum*, 27(3):1031–1038, 2008. Cited on pages 24, 25, 36, 79, 81, and 82.

F. McGee and J. Dingliana. An empirical study on the impact of edge bundling on user comprehension of graphs. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 620–627, 2012. Cited on page 82.

M. Meyer, T. Munzner, A. Depace, H. Pfister, and S. Member. MulteeSum: A tool for comparative spatial and temporal gene expression data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):908–917, 2010a. Cited on page 45.

M. Meyer, B. Wong, M. Styczynski, T. Munzner, and H. Pfister. Pathline: A tool for comparative functional genomics. *Computer Graphics Forum*, 29(3): 1043–1052, 2010b. Cited on page 45.

J. J. Miller and E. J. Wegman. Construction of line densities for parallel coordinate plots. In A. Buja and P. A. Tukey, editors, *Computing and Graphics in Statistics*, volume 36, pages 107–123. Springer-Verlag, New York, 1991. Cited on pages 15, 19, 27, 48, and 135.

R. Moustafa. Parallel coordinate and parallel coordinate density plots. *Wiley Interdisciplinary Reviews: Computational Statistics*, 3(2):134–148, 2011. Cited on pages 8, 20, 25, 28, and 42.

R. Moustafa and E. J. Wegman. On some generalization to parallel coordinate plots. In *Seeing a Million – A Data Visualization Workshop*, pages 41–48, 2002. Cited on pages 24 and 79.

R. E. A. Moustafa. QGPCP: Quantized generalized parallel coordinate plots for large multivariate data visualization. *Journal of Computational and Graphical Statistics*, 18(1):32–51, 2009. Cited on pages 20 and 28.

P. Muigg, M. Hadwiger, H. Doleisch, and E. Gröller. Visual coherence for large-scale line-plot visualizations. *Computer Graphics Forum*, 30(3):643–652, 2011. Cited on pages 28 and 45.

S. Nagaraj and V. Natarajan. Relation-aware isosurface extraction in multifield data. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):182–191, 2011. Cited on page 49.

H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM (Society for Industrial and Applied Mathematics), Philadelphia, PA, USA, 1992. Cited on pages 52 and 73.

M. Novotny. Visually effective information visualization of large data. In *Proceedings of the 8th Central European Seminar on Computer Graphics*, pages 41–48, 2004. Cited on page 26.

M. Novotny and H. Hauser. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):893–900, 2006. Cited on pages 15, 25, 26, and 40.

K. Pearson. Contributions to the mathematical theory of evolution. II. skew variation in homogeneous material. *Philosophical Transactions of the Royal Society of London*, 186:343–414, 1895. Cited on page 26.

W. Peng, M. Ward, and E. Rundensteiner. Clutter reduction in multi-dimensional data visualization using dimension reordering. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 89–96, 2004. Cited on page 40.

T. Porter and T. Duff. Compositing digital images. *Computer Graphics (Proceedings of SIGGRAPH)*, 18(3):253–259, 1984. Cited on page 73.

K. Pradhan, D. Bartz, and K. Mueller. SignatureSpace: A multidimensional, exploratory approach for the analysis of volume data. Technical report, University of Tuebingen, 2005. Cited on pages 45 and 49.

N. Promrit, A. Mingkhwan, S. Simcharoen, and N. Namvong. Multi-dimensional visualization for network forensic analysis. In *Proceedings of the 7th International Conference on Networked Computing*, pages 68–73, 2011. Cited on page 46.

H. Qu, W.-Y. Chan, A. Xu, K.-L. Chung, K.-H. Lau, and P. Guo. Visual analysis of the air pollution problem in Hong Kong. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1408–1415, 2007. Cited on pages 28, 39, and 40.

R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2012, ISBN 3-900051-07-0. Cited on pages 117 and 136.

E. Ramos and D. Donoho. The 1983 ASA data exposition dataset: Cars. http://lib.stat.cmu.edu/datasets/cars/cars.data, 1983, accessed 2013/02/19. Cited on pages 80, 95, and 100.

P. Riehmann, J. Opolka, and B. Froehlich. The product explorer: Decision making with ease. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 423–432, 2012. Cited on page 7.

J. Rodrigues, A. Traina, and C. Traina. Frequency plot and relevance plot to enhance visual data exploration. In *Proceedings of the XVI Brazilian Symposium on Computer Graphics and Image Processing*, pages 117–124, 2003. Cited on page 26.

R. Rosenbaum, J. Zhi, and B. Hamann. Progressive parallel coordinates. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 25–32, 2012. Cited on pages 36, 37, and 67.

O. Rübel, G. H. Weber, S. V. E. Keränen, C. C. Fowlkes, C. L. L. Hendriks, L. Simirenko, N. Y. Shah, M. B. Eisen, M. D. Biggin, H. Hagen, D. Sudar, J. Malik, D. W. Knowles, and B. Hamann. PointCloudXplore: Visual analysis of 3D gene expression data using physical views and parallel coordinates. In *Proceedings of the Eurographics/ IEEE-VGTC Symposium on Visualization*, pages 203–210, 2006. Cited on pages 28, 29, and 45.

C. E. Scheidegger, J. Schreiner, B. Duffy, H. Carr, and C. T. Silva. Revisiting histograms and isosurface statistics. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1659–1666, 2008. Cited on pages 9 and 49.

W. Schroeder, K. Martin, and B. Lorensen. *Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*. Kitware, Clifton Park, NY, USA, 2006. Cited on pages 49 and 136.

D. Scott and S. Sain. Multidimensional density estimation. *Handbook of Statistics*, 24(2004):229–261, 2005. Cited on page 26.

K. Shedden and S. Cooper. Analysis of cell-cycle gene expression in saccharomyces cerevisiae using microarrays and multiple synchronization methods. *Nucleic Acids Research*, 30(13):2920–2929, 2002. Cited on pages 117 and 118.

P. Shirley and A. Tuchman. A polygonal approximation to direct scalar volume rendering. In *Proceedings of the Workshop on Volume Visualization*, pages 63–70, 1990. Cited on page 53.

B. Shneiderman. Dynamic queries for visual information seeking. *IEEE Software*, 11(6):70–77, 1994. Cited on page 36.

B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343, 1996. Cited on pages 9, 14, 30, 104, and 136.

H. Siirtola. Direct manipulation of parallel coordinates. In *Proceedings of the International Conference on Information Visualisation*, pages 373–378, 2000. Cited on pages 28 and 36.

H. Siirtola and K. Raiha. Interacting with parallel coordinates. *Interacting with Computers*, 18(6):1278–1309, 2006. Cited on pages 8, 30, 31, and 36.

H. Siirtola, T. Laivo, T. Heimonen, and K. J. Raiha. Visual perception of parallel coordinate visualizations. In *Proceedings of the 13th International Conference on Information Visualisation*, pages 3–9, 2009. Cited on page 30.

M. Sips, B. Neubert, J. P. Lewis, and P. Hanrahan. Selecting good views of high-dimensional data using class consistency. *Computer Graphics Forum*, 28 (3):831–838, 2009. Cited on page 101.

A. R. Smith. A pixel is not a little square, a pixel is not a little square, a pixel is not a little square! (and a voxel is not a little cube). Technical report, Technical Memo 6, Microsoft Research, 1995. Cited on page 26.

P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9(12):3273–3297, 1998. Cited on pages 117 and 118.

C. A. Steed, J. E. Swan, T. Jankun-Kelly, and P. J. Fitzpatrick. Guided analysis of hurricane trends using statistical processes integrated with interactive parallel coordinates. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 19–26, 2009. Cited on pages 14 and 135.

S. S. Stevens. On the theory of scales of measurement. *Science*, 103(2684):677–680, 1946. Cited on page 5.

G. K. L. Tam, H. Fang, A. J. Aubrey, P. W. Grant, P. L. Rosin, D. Marshall, and M. Chen. Visualization of time-series data in parameter space for understanding facial dynamics. *Computer Graphics Forum*, 30(3):901–910, 2011. Cited on pages 14 and 135.

A. Tatu, G. Albuquerque, M. Eisemann, J. Schneidewind, H. Theisel, M. Magnor, and D. 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*, pages 59–66, 2009. Cited on pages 39 and 40.

A. Tatu, G. Albuquerque, M. Eisemann, P. Bak, H. Theisel, M. Magnor, and D. Keim. Automated analytical methods to support visual exploration of high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 17(5):584–597, 2011. Cited on page 40.

M. ten Caat and N. M. Maurits. Design and evaluation of tiled parallel coordinate visualization of multichannel EEG data. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):70–79, 2007. Cited on page 45.

M. ten Caat, N. M. Maurits, and J. Roerdink. Tiled parallel coordinates for the visualization of time-varying multichannel EEG data. In *Proceedings of the Eurographics/IEEE-VGTC Symposium on Visualization*, pages 61–68, 2005. Cited on page 45.

H. Theisel. Higher order parallel coordinates. In *Proceedings of the 5th International Workshop on Vision, Modeling, and Visualization*, pages 415–420, 2000. Cited on pages 24, 40, and 79.

R. Theron. Visual analytics of paleoceanographic conditions. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 19–26, 2006. Cited on page 43.

M. Theus. Interactive data visualization using Mondrian. *Journal of Statistical Software*, 7(11):1–9, 2003. Cited on page 28.

J. J. Thomas and K. A. Cook, editors. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Society Press, 2005. Cited on pages 4 and 117.

M. Tory, S. Potts, and T. Möller. A parallel coordinates style interface for exploratory volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 11(1):71–80, 2005. Cited on pages 45 and 49.

S. Tricaud, K. Nance, and P. Saadé. Visualizing network activity using parallel coordinates. In *Proceedings of the 44th Hawaii International Conference on System Sciences*, pages 1–8, 2011. Cited on page 46.

E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, USA, 2nd edition, 2001. Cited on pages 3 and 6.

J. W. Tukey. *Exploratory Data Analysis*. Behavioral Science: Quantitative Methods. Addison-Wesley Publishing Company, Massachusetts California London Amsterdam Ontario Sydney, 1977. Cited on page 4.

C. Turkay, P. Filzmoser, and H. Hauser. Brushing dimensions a dual visual analysis model for high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2591–2599, 2011. Cited on pages 31 and 117.

A. Unwin, C. Volinsky, and S. Winkler. Parallel coordinates for exploratory modelling analysis. *Computational Statistics & Data Analysis*, 43(4):553–564, 2003. Cited on pages 8 and 14.

C. Viau, M. J. McGuffin, Y. Chiricota, and I. Jurisica. The FlowVizMenu and parallel scatterplot matrix: Hybrid multidimensional visualizations for network exploration. *IEEE Transactions on Visualization and Computer Graphics*, 16(6): 1100–1108, 2010. Cited on pages 39 and 101.

J. Walker, Z. Geng, M. Jones, and R. S. Laramee. Visualization of large, time-dependent, abstract data with integrated spherical and parallel coordinates. In *EuroVis–Short Papers*, pages 43–47, 2012. Cited on page 28.

M. O. Ward. XmdvTool: Integrating multiple methods for visualizing multivariate data. In *Proceedings of the IEEE Conference on Visualization*, pages 326–333, 1994. Cited on page 33.

C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, San Francisco, CA, USA, 2004. Cited on pages 3 and 42.

R. Wegenkittl, H. Loffelmann, and E. Groller. Visualizing the behaviour of higher dimensional dynamical systems. In *Proceedings of the IEEE Conference on Visualization*, pages 119–125, 1997. Cited on pages 28 and 29.

E. J. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 85(411):664–675, 1990. Cited on pages 7, 8, 9, 18, 38, 39, and 101.

E. J. Wegman and Q. Luo. High dimensional clustering using parallel coordinates and the grand tour. In *Classification and Knowledge Organization*, pages 93–102. Springer, Berlin Heidelberg, Germany, 1997. Cited on pages 8 and 14.

E. J. Wegman and Q. Luo. On methods of computer graphics for visualizing densities. *Journal of Computational and Graphical Statistics*, 11(1):137–162, 2002. Cited on page 26.

L. Westover. Interactive volume rendering. In *Proceedings of the 1989 Chapel Hill Workshop on Volume Visualization*, pages 9–16, 1989. Cited on pages 66, 68, and 70.

L. Westover. Footprint evaluation for volume rendering. *Computer Graphics (Proceedings of SIGGRAPH)*, 24:367–376, 1990. Cited on page 69.

L. Wilkinson, A. Anand, and R. Grossman. Graph-theoretic scagnostics. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 157–164, 2005. Cited on page 101.

L. Wilkinson, A. Anand, and R. Grossman. High-dimensional visual analytics: Interactive exploration guided by pairwise views of point distributions. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1363–1372, 2006. Cited on pages 40 and 101.

P. C. Wong and R. D. Bergeron. 30 years of multidimensional multivariate visualization. In G. M. Nielson, H. Hagen, and H. Müller, editors, *Scientific Visualization, Overviews, Methodologies, and Techniques*, pages 3–33, Los Alamitos, CA, USA, 1994. IEEE Computer Society. Cited on pages 6 and 56.

P. C. Wong and R. D. Bergeron. Multiresolution multidimensional wavelet brushing. In *Proceedings of the IEEE Conference on Visualization*, pages 141–148, 1996. Cited on page 33.

L. Xu, T.-Y. Lee, and H.-W. Shen. An information-theoretic framework for flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16 (6):1216–1224, 2010. Cited on page 49.

J. Yang, W. Peng, M. O. Ward, and E. A. Rundensteiner. Interactive hierarchical dimension ordering, spacing and filtering for exploration of high dimensional datasets. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 105–112, 2003. Cited on page 40.

X. Yuan, P. Guo, H. Xiao, H. Zhou, and H. Qu. Scattering points in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6): 1001–1008, 2009. Cited on pages 22, 24, 79, and 101.

S. Zachow, P. Muigg, T. Hildebrandt, H. Doleisch, and H.-C. Hege. Visual exploration of nasal airflow. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1407–1414, 2009. Cited on page 45.

Z. Zhang, K. McDonnell, and K. Mueller. A network-based interface for the exploration of high-dimensional data spaces. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 17–24, 2012. Cited on pages 25, 39, and 40.

K. Zhao, B. Liu, T. Tirpak, and A. Schaller. Detecting patterns of change using enhanced parallel coordinates visualization. In *Proceedings of the Third IEEE International Conference on Data Mining*, pages 747–750, 2003. Cited on page 40.

X. Zhao and A. Kaufman. Multi-dimensional reduction and transfer function design using parallel coordinates. In *Proceedings of the IEEE/EG International Symposium on Volume Graphics*, pages 69–76, 2010. Cited on page 49.

X. Zhao and A. Kaufman. Structure revealing techniques based on parallel coordinates plot. *The Visual Computer*, 28(6):541–551, 2012. Cited on page 40.

H. Zhou, X. Yuan, H. Qu, W. Cui, and B. Chen. Visual clustering in parallel coordinates. *Computer Graphics Forum*, 27(3):1047–1054, 2008. Cited on pages 24, 25, 79, 81, and 82.

H. Zhou, W. Cui, H. Qu, Y. Wu, X. Yuan, and W. Zhuo. Splatting the lines in parallel coordinates. *Computer Graphics Forum*, 28(3):759–766, 2009. Cited on pages 28 and 66.

D. Zieker, E. Fehrenbach, J. Dietzsch, J. Fliegner, M. Waidmann, K. Nieselt, P. Gebicke-Haerter, R. Spanagel, P. Simon, A. M. Niess, and H. Northoff. cDNA microarray analysis reveals novel candidate genes expressed in human peripheral blood following exhaustive exercise. *Physiological Genomics*, 23(3): 287–294, 2005. Cited on page 119.