Available online at www.sciencedirect.com
ScienceDirect

ELSEVIER

Interacting with Computers 18 (2006) 1278–1309

Interacting with Computers

www.elsevier.com/locate/intcom

Discussion

Interacting with parallel coordinates

Harri Siirtola *, Kari-Jouko Räihä

Tampere Unit for Computer–Human Interaction (TAUCHI), Department of Computer Sciences, University of Tampere, Kanslerinrinne 1, FIN-33014 Tampere, Finland

Received 20 March 2006; accepted 20 March 2006
Available online 30 June 2006

Abstract

Parallel coordinate visualizations have a reputation of being difficult to understand, expert-only representations. We argue that this reputation may be partially unfounded, because many of the parallel coordinate browser implementations lack essential features. This paper presents a survey of current interaction techniques for parallel coordinate browsers and compares them to the visualization design guidelines in the literature. In addition, we report our experiences with parallel coordinate browser prototypes, and describe an experiment where we studied the immediate usability of parallel coordinate visualizations. In the experiment, 16 database professionals performed a set of tasks both with the SQL query language and a parallel coordinate browser. The results show that although the subjects had doubts about the general usefulness of the parallel coordinate technique, they could perform the tasks more efficiently with a parallel coordinate browser than with their familiar query language interface.

Keywords: Information visualization; Parallel coordinates; Multidimensional data; Interaction

1. Introduction

It is widely agreed that the visualization of multidimensional data is a challenging problem (e.g. Inselberg et al., 1998; Hibbard, 1999). The data is generally considered

* Corresponding author. Tel.: + 358 40 5488700.
E-mail addresses: harri.siirtola@cs.uta.fi (H. Siirtola), kari-jouko.raiha@cs.uta.fi (K.-J. Räihä).

0953-5438/$ - see front matter
doi:10.1016/j.intcom.2006.03.006

to be multidimensional if there are more than about five dimensions, and such data sets are not uncommon. There is a wealth of problems that require us to understand and study the complex relationships between high numbers of attributes, like the many problems in financial design, engineering, medical research, and biology. Popular examples of these are fraud detection, production yield optimization, efficacy of complex medical treatments, and genome research (Spence, 2001, pp. 45–51).

No single information visualization technique can be considered as the standard solution to the problem of visualizing multidimensional data, but one of the most general techniques is the parallel coordinate technique developed by Inselberg and Dimsdale (1990). Originally, the parallel coordinate technique was developed for n-dimensional geometrical computations. It has been used successfully in the development of complex algorithms and in solving multidimensional optimization problems. Besides computational geometry and algorithmics, the technique is also widely utilized in the visualization of multidimensional information.

There are many reasons for the popularity of the parallel coordinate technique in the field of information visualization. One of the advantages is the uniform treatment of multiple dimensions that is paramount in exploratory tasks. Another advantage is that the technique allows users to interact with the data in many ways. This is highly useful, as interaction is known to augment the knowledge acquisition process. This general result was first shown by Gibson (1962).

The criticism towards parallel coordinate visualizations has acknowledged the power of the technique, but regarded it quite difficult to learn and use, and judged it more suited for expert users (Shneiderman, 1998, p. 530). We argue that the problem might be in the interaction with parallel coordinate visualizations—perhaps we have not yet found the most productive and usable interaction techniques. Another point is that the steady improvement of computing hardware allows us to do more within the direct manipulation time constraints (Ware, 2004, pp. 222–223).

This article presents a survey of interaction methods found in parallel coordinate browsers, evaluates them against established user interface guidelines, and reports an empirical study in which we experimented with the immediate usability of parallel coordinate visualizations. We show the rich interaction possibilities this particular visualization technique offers, and demonstrate the importance of interactivity for the technique.

1.1. Interacting with visualizations

Information visualization tools are extensions of our cognition, allowing us to process larger and more complex data sets than would be possible without external means. Interaction is crucial in this processing because the active manipulation facilitates discovery. The perception researcher and theorist James J. Gibson demonstrated this in an experiment where the task was to match cookie-cutters by touch only (Gibson, 1962, 1983). The participants in the passive touch condition were able to recognize 49% of the shapes versus the 95% success rate in the active touch condition. Although the experiment deals mainly with haptic touch, it supports the claim that interaction is important and improves our knowledge acquisition considerably.

In the case of parallel coordinate browsers, the role of interaction is crucial: due to the strong overlapping of graphical elements, large parallel coordinate plots are difficult to understand without the chance to interact with them (Artero et al., 2004).

The significance of interaction in information visualization can be seen in the reference models for the process. In the models of Card et al. (1999, p. 17); Spence (2001, p. 13), interaction is the glue that keeps the whole process together. Perhaps the most insightful analysis of the role of interaction in information visualization is present in Ben Shneiderman's work. He recognized and named the concept of *direct manipulation* (Shneiderman, 1983) in human–computer interaction. In the context of information visualization, it means avoiding indirectness in controlling and manipulating the visualization. Instead of using controls that are separate from the visualization, the user is able to manipulate the visualization artifact itself. By presenting the data, the query, and the query result as visible and controllable objects, the interaction is easier to learn and it creates an illusion of being in direct contact with the data. The direct control with rapid and continuous feedback, incremental changes, and reversible effects will ideally make the tool 'disappear' (Shneiderman, 1998, pp. 71–72), in the same sense as playing a violin: there is nothing between you and the music. The feeling of being in control is created by systems that are able to respond within 160 ms to user actions (Ware, 2004, pp. 222–223).

A good practical guideline for creating a user interface for an interactive information visualization tool is also given by Ben Shneiderman (1996). In a form of a mantra, he recommends that an effective visualization tool should function along the following principle:

Overview first, zoom and filter, then details on demand.

The mantra recommends an approach where we first provide an overview of the whole data set, then offer tools for rapid panning and zooming over the presentation, provide data filtering to reduce the clutter, and finally, facilitate drill-down into detailed information when requested. The mantra is accompanied by a task taxonomy for information visualizations that specifies seven tasks at a high level of abstraction (Shneiderman, 1996):

Overview. Gain an overview of the entire collection.
Zoom. Zoom in on items of interest.
Filter. Filter out uninteresting items.
Details-on-demand. Select an item or group and get details when needed.
Relate. View relationship among items.
History. Keep a history of actions to support undo, replay, and progressive refinement.
Extract. Allow extraction of sub-collections and of the query parameters.

In Section 3, we will discuss how this taxonomy maps to the features of current parallel coordinate browsers, and evaluate how the interaction with parallel coordinate relates to information visualization design guidelines. Besides the mantra, there is also another guideline by Shneiderman, MAGIC (IV Society, 2005), that gives another set of recommendations at a higher level.

1.2. Related work

Alfred Inselberg began the work on parallel coordinates in 1981 while working at the IBM research laboratory and published the preliminary ideas as a technical report (Inselberg, 1981). The results were later extended in a number of publications (Inselberg, 1981, 1985, 1997a,b, 1998, 1999, 2000, 2001, 2002; Inselberg and Avidan, 1999; Inselberg and Dimsdale, 1990, 1994a,b; Inselberg et al., 1987, 1998), and led to several patents (Austel et al., 1996; Inselberg, 1989; Inselberg et al., 1992, 1997) and to a commercial parallel coordinate browser called *Parallax*.

General techniques for interacting with parallel coordinates have been discussed by Siirtola (2000a, 2003), Graham and Kennedy (2003), Johansson et al. (2004), Zhao et al. (2003), Hauser et al. (2002a,b), Hauser and Kosara (2004) and Novotny (2004).

Interacting with parallel coordinate visualizations has also been studied in specific tasks, like in early stages of data-understanding (Kandogan, 2001), trend and pattern analysis (Zhao et al., 2003), exploratory modeling analysis (Unwin et al., 2003), geographical visualization (Edsall, 2003), and subset detection (Andrienko and Andrienko, 2004). We focus here on tasks related to exploratory visualization or knowledge mining. Perhaps one of the most common tasks in information acquisition is cluster analysis, or finding out if the data set can be divided into groups of similar items. The use of parallel coordinate visualizations for this task has been considered, e.g. by Ankerst et al. (1998), Chou et al. (1999), Liu et al. (2002), Artero et al. (2004), and Inselberg in the Parallax software tool (2001).

Besides the *Parallax* system, several parallel coordinate browser implementations have been introduced in the literature: *Parallel Visual Explorer* by IBM, *parvis* (Hauser et al., 2002a; Ledermann, 2003), *Parallel Coordinate Explorer* (Siirtola, 2000a), *VizCraft* (Goel et al., 1999), two unnamed implementations by Wegman (1990), and *XmdvTool* (Rundensteiner et al., 2002). In addition, many of the visualization systems for multidimensional data include a parallel coordinate module, although they generally provide quite limited interaction methods (e.g. *Mondrian* (Theus, 2002), *XGobi* (Buja et al., 1996) and its successor *GGobi* (Swayne et al., 2001), and *Weave* (Gresh et al., 2000)). Advanced statistical software systems, like *R* (R Development Core Team, 2005), usually provide tools to produce (static) parallel coordinate plots.

1.3. Structure of the paper

This paper starts with an introduction to the concept of parallel coordinates and proceeds with a survey of current interaction methods found in parallel coordinate browsers. Section 2 presents the empirical study we conducted to explore the immediate usability of parallel coordinate visualizations with our prototype implementation. Section 5 presents conclusions and future directions.

2. Parallel coordinate plots

The fundamental idea in parallel coordinate plots is to place coordinate axes in parallel, and to present a data point as a connection line between coordinate values. This is illustrated in the case of a 2D point in Fig. 1. The orthogonal axis placement allows only two axes on a plane, but there is no upper limit, in theory, for the parallel axis configurations.

A parallel coordinate plot represents an n-dimensional data point by laying out n parallel axes equidistantly on a plane. Then an n-dimensional point $(x_1, x_2, \ldots, x_n)$ is represented as a set of line segments connecting the values $x_1, x_2, \ldots, x_n$ in the parallel axes. A single set of connected line segments representing one multidimensional data item is called a *polyline*. A 6D data item is represented as a parallel coordinate plot in Fig. 2. The method allows the user to visualize data sets with a high number of attributes, depending only on the availability of horizontal space.

The parallel coordinate plot can be characterized as a piecewise representation of a high-dimensional data space, analogous to a set of pairwise cross-tabulations of the data. If we have, for example, a 3D data point (x, y, z) as a parallel coordinate plot, we can see that each data point is actually represented as a set of connections between axes. The first pair of connections between axes X and Y represents the XY coordinate plane, and the second pair Y and Z represents the plane YZ, correspondingly. Bringing the axes X and Z next to each other would form the third XZ plane. As the order of axes does not convey any information in parallel coordinate plots, we may rearrange them freely. Fig. 3 shows the representation of two 3D data points, (1, 2, 1) and (2, 2, 3), as two parallel coordinate plots. With three axes, we have six possible axis permutations, but we need only two to represent the prevailing relations. The initial arrangement on the left in Fig. 3 also illustrates the *ambiguity problem*: the plot is identical for two sets of data points, namely, {(1, 2, 1), (2, 2, 3)} and {(1, 2, 3), (2, 2, 1)}.

The advantages of the parallel coordinate representation are obvious if we plot the 3D points (1, 2, 1) and (2, 2, 3) as projections of the 3D space. Three relevant projections and a 3D view are shown in Fig. 4. The projections require combining information from three different views to form the image in 3D space, and the non-orthogonal 3D view is difficult to perceive without the natural depth cue provided by the head-motion parallax. If the user interface allows rotating the 3D view, it will be considerably easier to form a mental image of the situation, especially if the

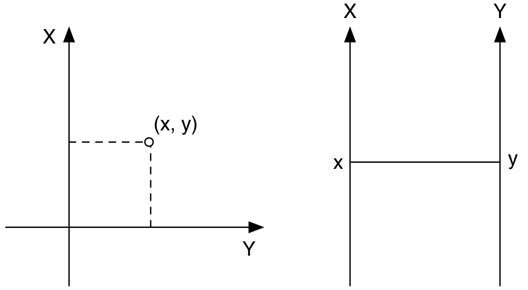


Fig. 1. A 2D data point as a parallel coordinate plot.

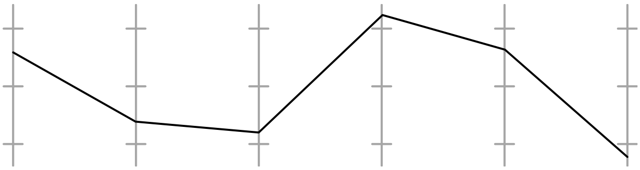


Fig. 2. A 6D data point as a parallel coordinate plot.

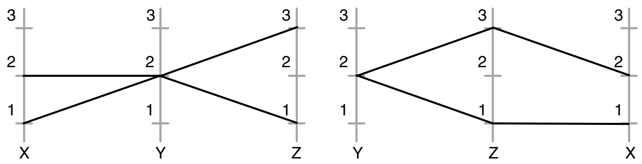


Fig. 3. Two 3D data points (1, 2, 1) and (2, 2, 3) as two parallel coordinate plots with different axis

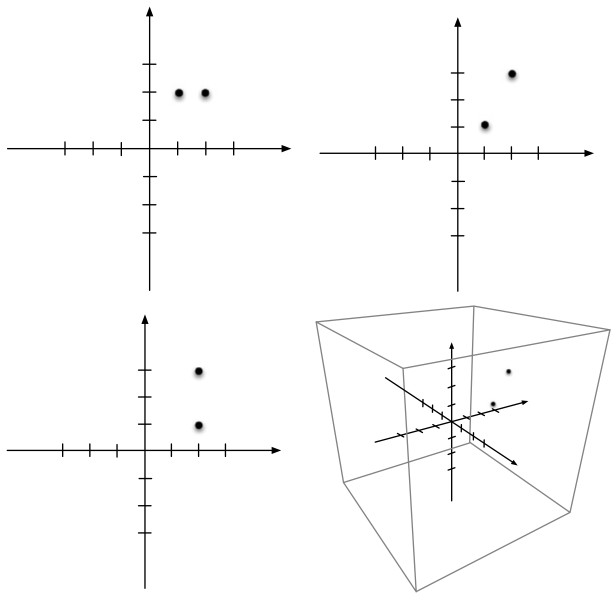


Fig. 4. Two data points (1, 2, 1) and (2, 2, 3) in 3D space. From the left to right: XY-plane, XZ-plane, YZ-plane, and a non-orthogonal 3D view.

feedback is within the direct-manipulation limits (Yuan et al., 2000). Beyond 3D, the projection approach requires us to read and combine information from a growing number of 2D views, and becomes even more difficult to use. Furthermore, the overall view is not possible in the dimensions beyond three.

Another feature related to the piecewise nature of the parallel coordinates is the *point and line duality* (Inselberg and Dimsdale, 1990). The duality is easier to discuss if we first consider the points and lines on a 2D plane. A point on a plane is represented in a parallel coordinate plot as a line that connects the coordinate pair. Likewise, the set of points that lie on the same line on a 2D plane is represented as a set of connecting lines in a parallel coordinate plot. The connection lines are either parallel or they intersect exactly in the same point. Thus, a line in a 2D space can be a point in a parallel coordinate plot and vice versa, as illustrated in Fig. 5.

It is possible to detect if the points fall on the same line in the 2D space by checking if the connection lines are parallel or form an intersection point in a parallel coordinate plot. This method works for higher dimensions also, but in that case we must check a set of intersection points. Depending on the values, the intersection point may appear between the parallel axes, on the axes, or outside a pair of axes (Fig. 6). In the last case, the intersections of line segments are not visible. The corresponding line in a 2D orthogonal space would have a positive slope in cases (a) and (e) and a negative slope in case (c). Case (b) corresponds to a vertical line and case (d) to a horizontal line. Parallel lines in the parallel coordinate visualization would correspond to a line with slope one in the 2D orthogonal space.

2.1. Parallel coordinates in information visualization

The parallel coordinate technique is a sophisticated way to do *n*-dimensional Euclidian geometry, but it can also be utilized to visualize non-numerical multidimensional data. If we allow the parallel axes to have different scales and zero points, and map the values of nominal scale attributes into small integers, the parallel coordinate plot becomes a powerful tool to visualize multidimensional information.

In the canonical situation, the axes of a parallel coordinate plot have an identical scale and the same zero point, as in Fig. 3. This allows the plot to be used directly for geometrical computations that involve the angle of line segments. However, in visual data mining and in information visualization in general, it is common to have different scales for the axes and different zero points. The most common alternatives are [*min*, *max*] scale, [0, *max*] scale, or [0, *max*] with an absolute scale (equal scale for all axes). The [*min*, *max*] scale utilizes all the available plot space in each dimension and reduces the line segment occlusion and the drawing of multiple line segments on top of each other. Using the zero-based scale on all axes makes the comparison of

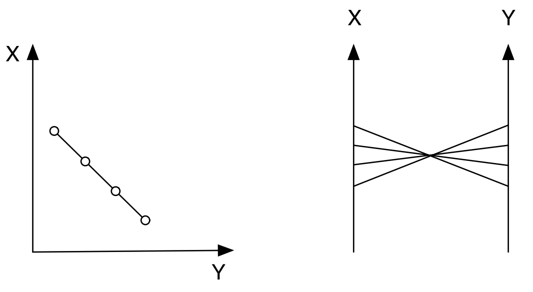


Fig. 5. The point and line duality: a set of points lying on a same line in 2D orthogonal space forms an intersection point in a parallel coordinate representation.

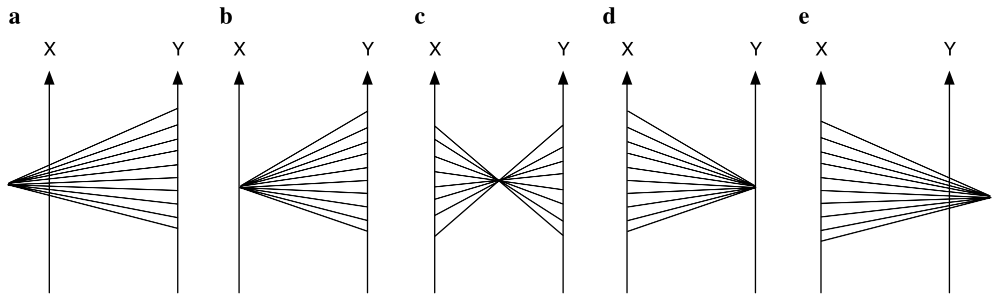


Fig. 6. The formation of the intersection point in parallel coordinates for lines in 2D space.

proportions easier even if the maximum values are individually scaled. The third alternative, using zero-based absolute scale, facilitates absolute comparisons by maintaining equal zero point and scale on all axes.

Parallel coordinate plots were originally created for the representation and exploration of high-dimensional numerical data, requiring the axes to have numeric values and an equal scale. It was later realized (Rosario et al., 2003; Siirtola, 2000a) that the technique is also useful if we allow the axes to have different scales and represent the ordinal or even nominal scale attributes in a similar manner. Mapping the values into small integers accomplishes this, permitting the attributes to be treated as numeric values. This extension makes the parallel coordinate plot a viable tool for general information visualization tasks like visual data mining (Wegman, 1999).

As an example, let us consider the parallel coordinate plot of the cars data set in Fig. 7 from the 1983 ASA Data Exposition (Ramos and Donoho, 1983). The data is about cars tested by the Consumer Reports magazine between the years 1970 and 1982 and consists of 406 cars described by nine attributes (it is often described as a 406×8 data set, but the parallel coordinate visualization treats the row identifiers like any other attributes). Five of the attributes are continuous ratio scale attributes: MPG (miles per US gallon), DISPL (engine displacement, in cubic inches), HP (engine horsepower), WEIGHT (vehicle weight, in US pounds), and ACCEL (acceleration performance from 0 to 60 MPH). Attributes CYL (engine cylinder count, one of 3, 4, 5, 6, or 8) and YEAR (manufacturing year, from 1970 to 1982) are ratio scale attributes with discrete values, and CAR (make of the car) and ORIGIN (one of ‘American’, ‘European’, or ‘Japanese’) are nominal scale attributes.

3. Interacting with parallel coordinate plots

This section presents a survey of interaction techniques for parallel coordinate browsers. In addition to the literature and freely available browser implementations, the presentation is based on several browser prototypes we have constructed, and on the experiences we have gained from user tests. The non-attributed screen pictures in this section are from our browser prototypes. One of the prototypes is available on the web for experimenting (Siirtola, 2000b).

3.1. Gaining an overview

Gaining an overview of a data set with a parallel coordinate visualization is simple, since the plot itself is an overview, albeit an abstract one. Without any interaction, it is possible to observe the characteristics of the attributes and even the relationships between attributes placed on the adjacent axes. However, although the overview includes all the data values, it contains only a subset of possible connections between the data items. The consequence is that an important relationship may remain unnoticed if the corresponding axes do not appear beside each other.

In Fig. 8, two parallel coordinate plots show a subset of attributes from the cars data set. The inverse relationship between attributes MPG and HP is obvious from

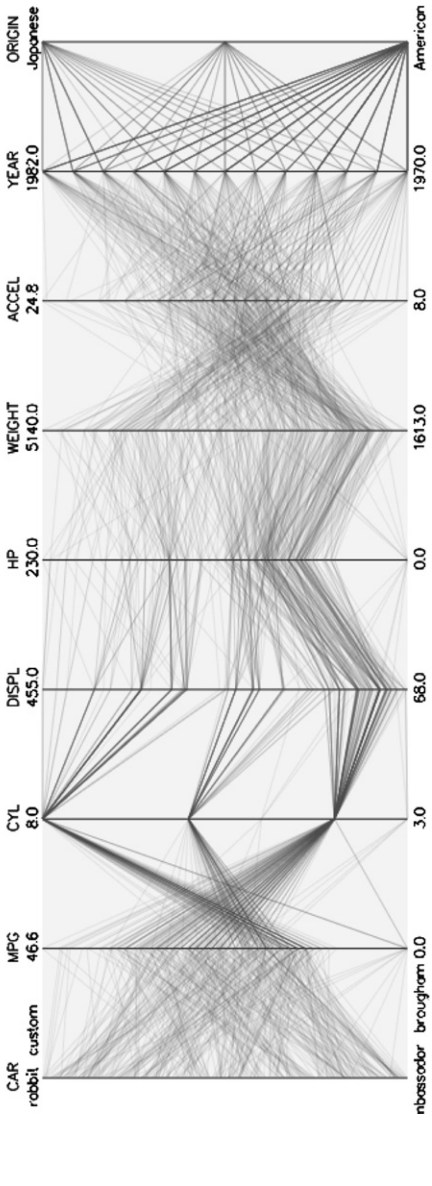


Fig. 7. A parallel coordinate plot of the cars data set.

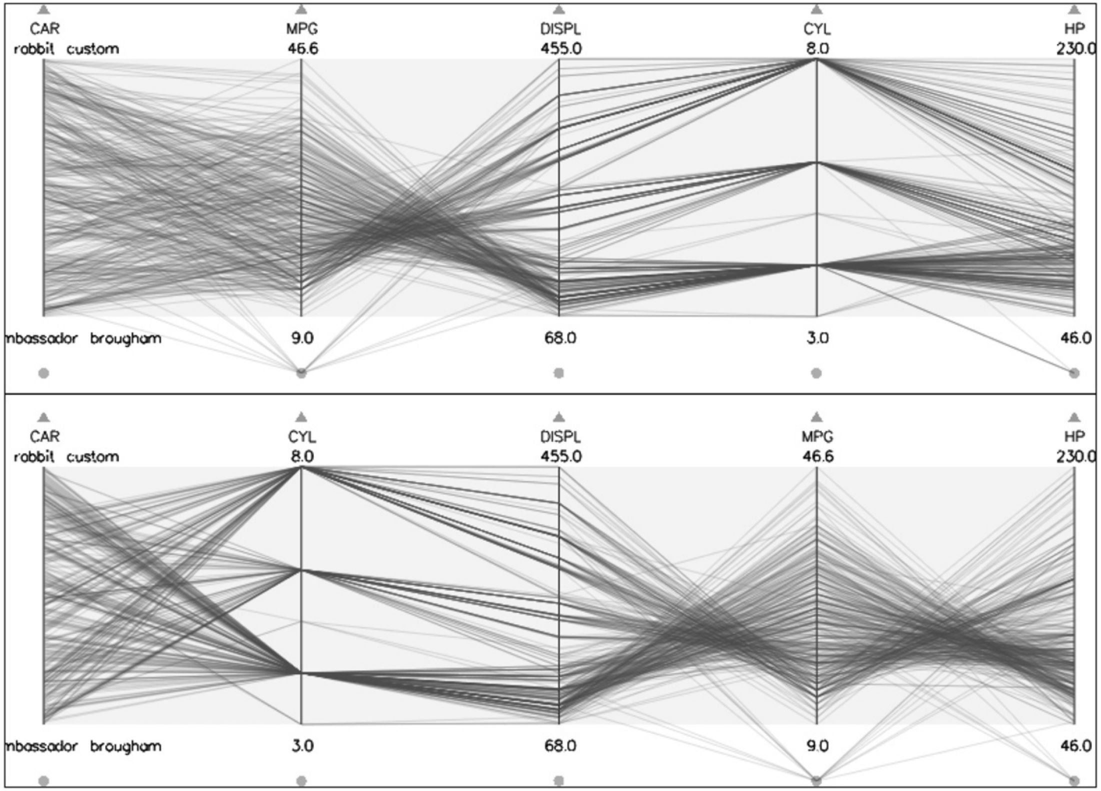


Fig. 8. Two axis arrangements for a subset of attributes. The inverse relationship between attributes MPG and HP is hidden in the upper plot, but visible in the lower one.

the lower plot, but it is hidden in the upper one. This rearrangement problem in parallel coordinate plots is discussed further in Section 3.3.

3.2. Brushing a parallel coordinate plot

The fundamental interaction technique in parallel coordinate browsers is *brushing*. It is used both in relating data items to each other and in zooming into them. In this context, brushing means selecting a set of polylines and applying a *brush operation* on them. The simplest and most common brush operation is to highlight the brushed set of polylines to facilitate comparisons between the brushed and unbrushed polylines. Other possible operations include zooming, deleting, masking, or any other operation that expects a subset of polylines as input. A special case is the brushing of a single polyline: it allows us to zoom into the actual attribute values of the data item, thus supporting the details-on-demand style of interaction. Our implementations display the data values of a single selected polyline below the corresponding axes, as in Fig. 13.

Brushing can be either a between-views or a within-view operation. In the first case, the two views are often conceptually different, such as a bar chart view and a parallel coordinate view. This kind of between-views brushing is common in software tools for exploratory dynamic analysis (e.g. XGobi, GGobi, and Mondrian). In a single view case, the same view acts both as input and output.

The actual selection of polylines can be implemented in a number of ways. The space between parallel axes does not carry meaning in the same way as, for example, the space in the 2D Euclidian coordinate plane does. If the selection is based on hitting the polylines, we are selecting connections, not actual data points in the data set. The other approach is to limit the selection into one dimension only, to the points on a parallel axis. These two approaches correspond to selecting a rectangular area and a range on an axis (Fig. 9), respectively, and the preference depends on the task. It is generally easier to highlight outliers with a 2D selection, but it is easier to select a range of values from an axis with a 1D selection.

Brushing a single axis is a single-dimensional selection, and it is not always adequate when we want to explore multidimensional data. Multidimensional brushing can be implemented by making the selections on each axis persistent and modifiable. Martin and Ward (1995) have proposed in XmdvTool a brush that is like a modifiable 'tunnel'. Polylines that fall completely within the tunnel are selected, as shown in Fig. 10.

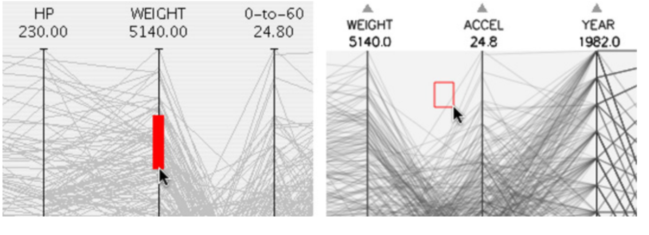


Fig. 9. Brushing a subset of polylines: a single-dimensional (axis) selection on the left, and a 2D (line) selection on the right.

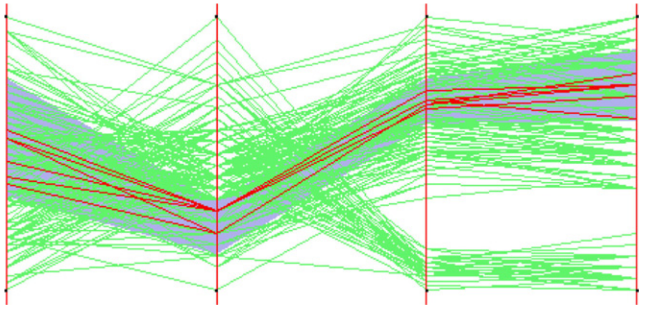


Fig. 10. A high-dimensional brush in XmdvTool (Martin and Ward, 1995). The 'tunnel' that determines the brush limits shows as a gray region underneath the polylines.

In Parallel Coordinate Explorer (PCE) (Fig. 11), the range limits are displayed next to their triangle symbols, thus allowing the display of exact values for more than one range per axis. The components of brushes can be activated and manipulated both directly from the visualization and indirectly from the lists showing the exact values.

The obvious next step after multidimensional brushes is to allow several multidimensional brushes simultaneously. With multiple brushes, it is possible to make elaborate comparisons between subsets of polylines. The multiple brushes can be differentiated by assigning distinct highlight colors to them. This introduces a new problem: how should we color the lines that are in the intersection of two brushes? In the current version of PCE, we make the transparency of polylines adjustable and change the depth order of brushes to keep the most recently accessed brush on top (Fig. 11). Another possibility is to compute a new color for the set of polylines in the intersection, but then it would be challenging to create an effective mapping.

Hauser et al. (2002a) have proposed another kind of indirect brushing operation that has been implemented in a tool called *parvis* (Ledermann, 2003). In their method, a set of polylines is brushed according to their range of slope between two axes. As seen in Fig. 12, the angular brush tool can reside anywhere between the axes, thus breaking the direct mapping between the tool and the set of polylines.

One of the desired features of information visualization tools is the ability to provide simultaneously both overview and detail. In a parallel coordinate browser, the overview is always there, and implementing a single polyline brush and selection can provide the details (Fig. 13).

Brushing a single polyline facilitates two kinds of comparisons: the single polyline compared to the whole data set, and the single polyline compared to the current subselections. Implementing the selection and highlighting of single polylines also avoids the ambiguity problem that monochromatic parallel coordinate plots have.

3.3. Rearrangement in parallel coordinate visualizations

Another important interaction technique with data that is routinely used almost everywhere is *rearrangement*. The role of rearrangement in information visualization

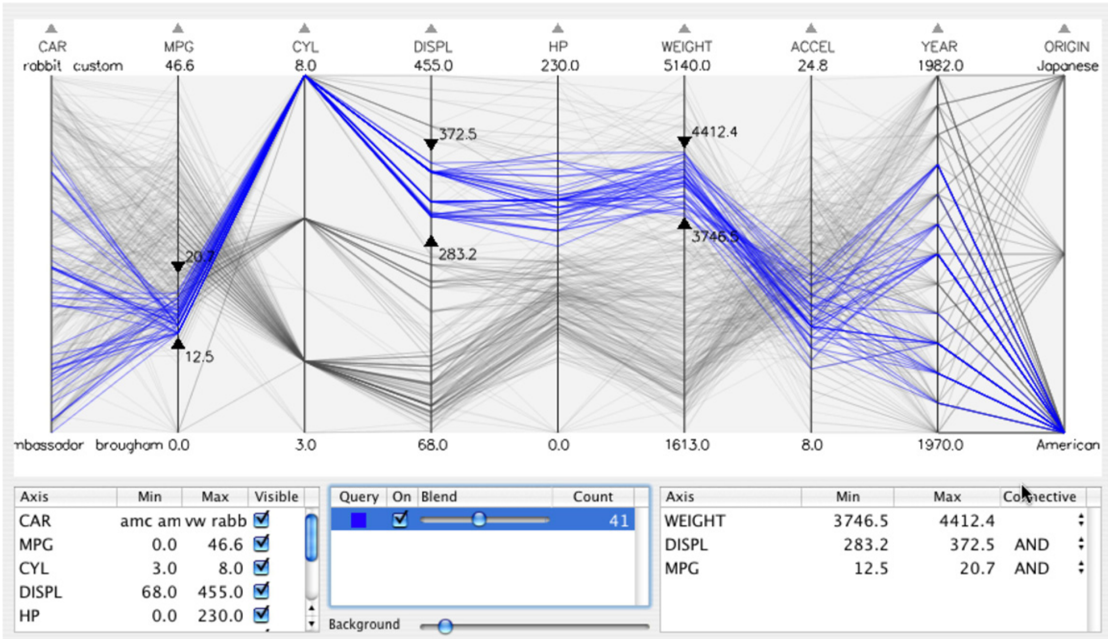


Fig. 11. The new version of the Parallel Coordinate Explorer user interface with dual representations for multidimensional brushes.

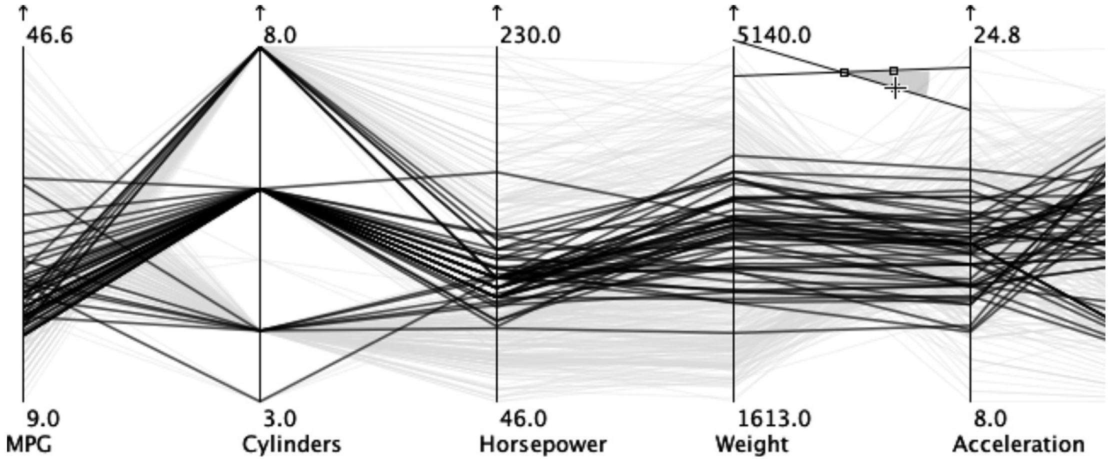


Fig. 12. Brushing polylines according to the angle of line segments between two axes: parvis (Ledermann, 2003). The angle specification tool is visible between the axes *Weight* and *Acceleration*.

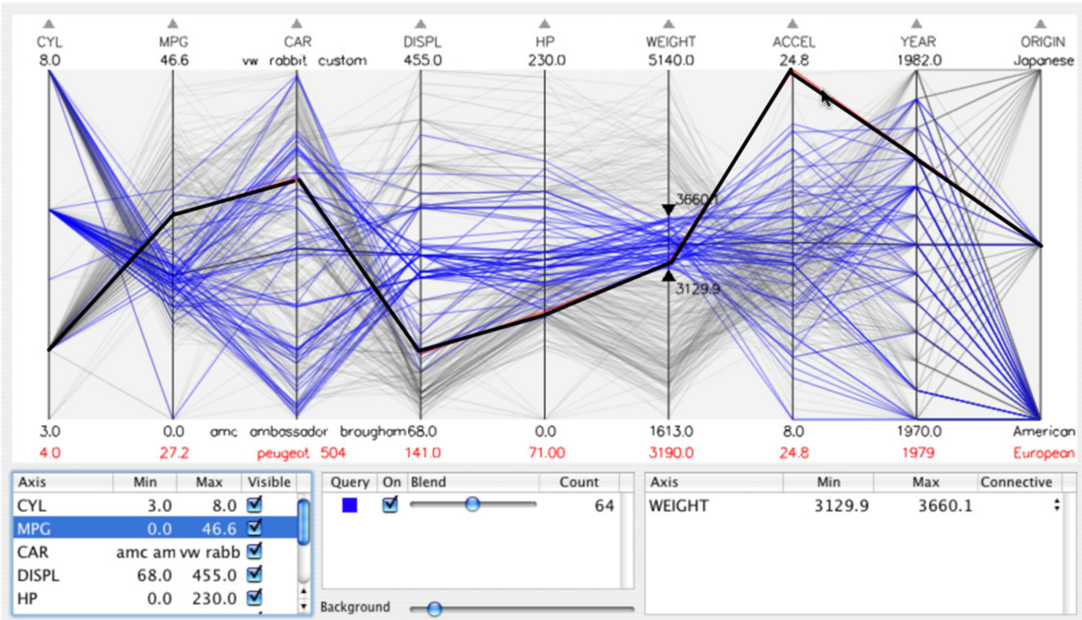


Fig. 13. Brushing a single polyline: the selected line is highlighted with black and the value of each dimension is shown below the axis.

is important, since often a mere rearrangement may give additional insight into the data (Bertin, 1981, 1983, p. 254; Spence, 2001, p. 14–15). In parallel coordinate visualizations, we may rearrange the order of axes and change their direction to produce different views into our data set. The axis rearrangement is an important operation, because the connections in the data set are visible only on the adjacent axes.

If we have an n-dimensional data tuple, we have $n!$ possible arrangements for the parallel axes. However, there is a lot of redundancy in this set, and the minimal number of arrangements to ensure the adjacency of every pair of axes is actually $\lceil (n+1)/2 \rceil$ (Wegman, 1990).

Usually, the idea in rearrangement is to bring together axes that might have an interesting relationship, and to observe the pattern formed by the connection lines. As noted earlier, the connection lines might intersect in a tight pattern that indicates a line formation on that plane. The shape and tightness of the formation is more difficult to perceive if the coordinate values grow in the same direction. Observing this phenomenon can be facilitated by allowing the inversion of the axis scale, as can be seen in Fig. 14. Inversion is also a neutral operation since the connections remain the same.

The manipulation of axes can also include operations for hiding and revealing axes. This focusing or zooming assists in concentrating on the attributes that are relevant to a task. In contrast, the brush operation described in Section 3.2 facilitates zooming into a subset of values.

3.4. Abstracting in parallel coordinate visualizations

The most characteristic issue of parallel coordinate visualizations is the *occlusion problem*. It is caused either by a large set of polylines or by a data set containing overlapping polylines with relatively few distinct data values. In the former case, the visualization will turn the parallel coordinate plot into a solid color surface

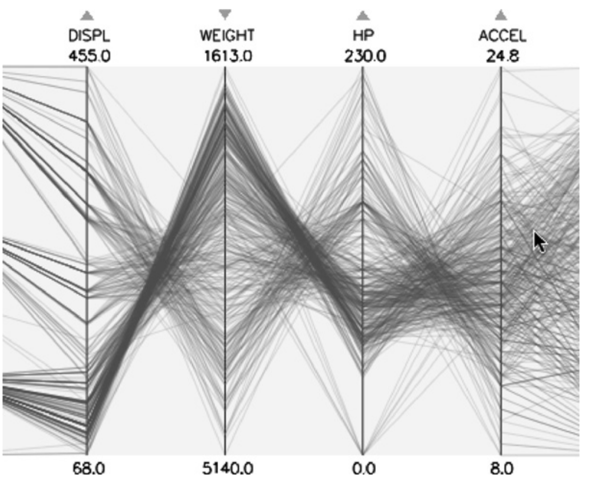


Fig. 14. Reversing the scale of axis WEIGHT to observe relationships with axes DISPL and HP.

without much detail, and in the latter case, it is impossible to differentiate the polylines from each other because of their stacking. In both situations, the visualization is rendered almost unreadable.

The stacking problem includes the ambiguity problem discussed in Section 2. In a simple situation, the ambiguity problem can be resolved by brushing and highlighting a single polyline, allowing us to see at least the topmost polyline in relation to others. Another proposed solution is to use smoothly curving line segments instead of straight ones, and thus make the continuations easier to detect (Graham and Kennedy, 2003).

Perhaps the most intuitive solution to the occlusion problem caused by a large number of polylines is to make them transparent (Wegman and Luo, 1997). When the display is generated with an appropriate alpha channel value, the effect is that the overlapping line segments appear darker, giving a natural cue of the stacking. The correct adjustment of the transparency depends on the data set and the situation, requiring the setting to be one of the user-adjustable parameters of the visualization. In our current implementation (Fig. 11), the alpha value can be adjusted separately for the background and for each query.

The alpha channel adjustment is not a sufficient abstraction for a set of polylines when the task requires more elaborate comparisons between the sets. Fua et al. (1999) proposed a hierarchical clustering method to abstract a really large set of polylines into a single, thicker line surrounded by a gradient representation of data density, or a 'variable width opacity band'. Another visual abstraction method based on clustering was proposed by Novotny (2004). His method forms a hierarchy of clusters using different algorithms at different levels to achieve the efficiency needed in a direct manipulation interface.

The abstraction of polylines can also be based on the data tableside by summarizing the data. Our earlier prototype abstracts a set of polylines by replacing it with a thick line traversing through the arithmetic mean of the values on each axis (Fig. 15). The variation of data values on each axes is indicated with boxes that are sized according to the standard deviation of data values. Variations of this idea has been implemented in several of the parallel coordinate browsers.

In our later prototypes, we decided to abandon the mean and standard deviation technique because of the sensitivity problems. For example, the arithmetic mean is seriously affected even by a single outlying data item. Often a better choice is to use the median that is still relevant even if almost half of the data items are wildly off.

In our current implementation, seen in Fig. 16, the polyline abstraction is based on Tukey's box-and-whiskers plots (Tukey, 1977, Section 2C). We indicate the median value of the selected polylines by drawing a thick black line through the median values of data items on each axis. The dark area around the median covers the range of values from the 25% quartile to the 75% quartile, called the interquartile range (*IQR*). Half of the brushed polylines are inside the *IQR*. The surrounding light area extends 1.5×*IQR* to both sides of the median, if the data items exist. Outliers beyond the 1.5×*IQR* distance from the median are not indicated in this representation.

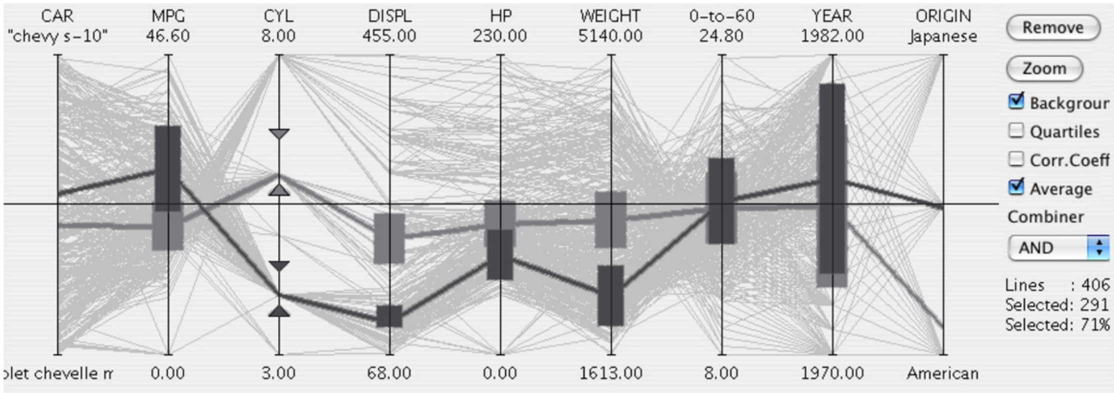


Fig. 15. Summarizing a set of polylines by presenting graphically the mean and the standard deviation of line sets. The visualization shows an overall comparison of four and six-cylinder cars.

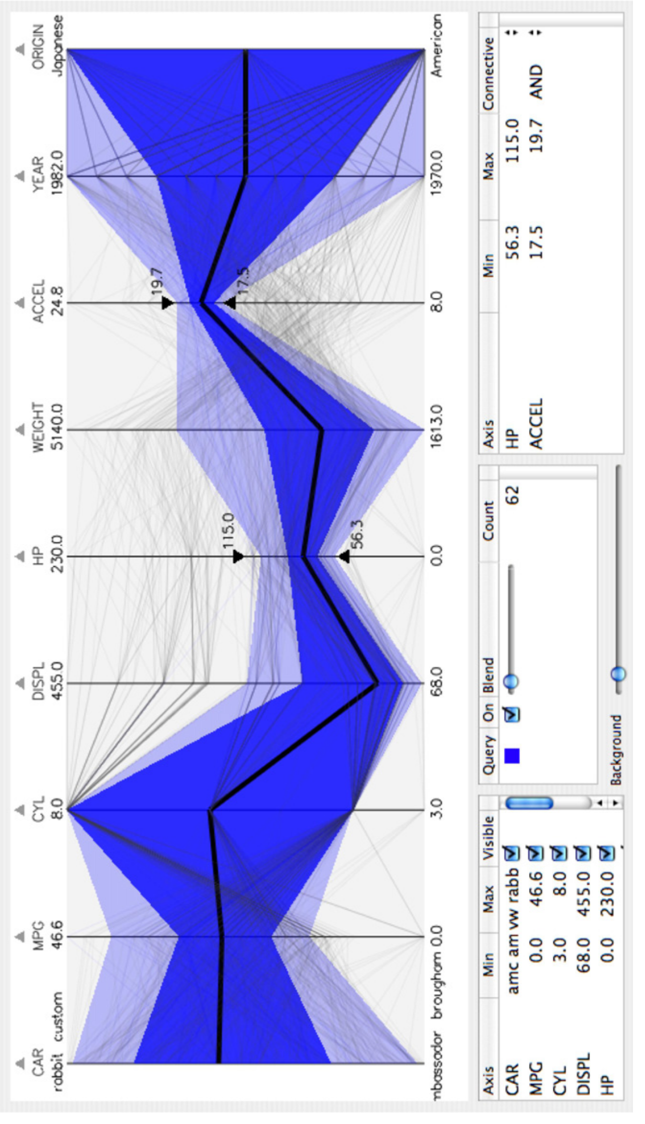


Fig. 16. A polyline abstraction based on the box-and-whiskers plot.

3.5. Missing data in parallel coordinate visualizations

Data sets are often incomplete because of various reasons. The missing data is a problem in information visualization if the visualization technique does not handle it gracefully. Many of the visualization techniques do not allow null values in the data, thus requiring the users to supply some constant that denotes a missing value. This is often misleading and may lead to wrong conclusions about the data, especially if the placeholder values are used in the calculations.

The cars data set has cars with unknown mileage and unknown horsepower. There are at least four strategies for dealing with them in the parallel coordinate visualizations:

(1) Remove the incomplete tuples from the data.
(2) Leave out the missing values from the visualization.
(3) Use a special, axis-dependent constant value to denote the missing data items.
(4) Treat the missing values as special cases with clear distinction to valid data.

In our example data set, the first alternative simply discards 14 out of the 406 cars that have incomplete data tuples. In this data set, the number of incomplete tuples is relatively low (3.5%) and the exclusion strategy might be acceptable.

The second option, leaving out the missing values from the parallel coordinate plot, leads to polylines rendered broken. Although this is a logical alternative, the discontinuity in a polyline may surprise the user and is against the underlying idea of this representation.

The third alternative is perhaps the most common approach, although there are known problems in it even in a 1D situation (Eaton et al., 2005). In this case, we could use the value zero as a reasonable marker for missing information on mileage or horsepower, since the users would probably interpret them correctly. However, introduction of zero values would change the scale of the affected axes, and make it harder to observe the data distribution. An example of this approach can be seen in Fig. 15.

The final approach was suggested by Inselberg et al. (1997) in the patent describing the parallel coordinate representation. He recommends that the default value for missing data is either 5% above or below the axis maximum or minimum, respectively. With this approach, the user can immediately determine the absence of data as the polyline traverses outside the axis range. An adaptation of this method can be seen in Fig. 8.

With so many options available, the best solution is to leave the choice to the user. For instance, if the default representation is the one shown in Fig. 8, the user could be allowed to drag the markers of missing values to a position on their axis where they interfere with the exploration as little as possible.

4. Empirical study

We have discussed several ways to support interactive exploration with parallel coordinates. How relevant are these techniques? In particular, could they alleviate the claimed complexity of parallel coordinates and make the technique a viable tool for visual data mining?

We carried out an experiment where sixteen IT professionals with considerable SQL experience solved a set of simple query tasks both with the SQL query language and a parallel coordinate browser. The subjective satisfaction, efficiency and correctness of the task execution were measured. The subjective ratings were recorded with a questionnaire and by interviewing the subjects after their test session. The subjects were first-time users of the parallel coordinate browser, and they were given 5–10 min training on using the browser. The goal of the experiment was to study the immediate usability of parallel coordinate browsing, and to establish how proficient the seasoned SQL programmers are with an alternate user interface.

The motivation for the experiment was to refute the claim that the parallel coordinate visualization is difficult to use and is suited only for the experts. We did not expect one of the interfaces to be clearly superior over the other, since it is clear that the usefulness depends on the task. Generally, it is obvious that the SQL language is superior when we need to find data items fulfilling a complex set of constraints, and that the parallel coordinate visualization is better for gaining an overview of a data set.

4.1. Participants

Sixteen volunteers (6 women and 10 men) from a local IT company took part in this study and received no monetary compensation. The only pre-screened requirement for the participants was that they had been using the database query language SQL as part of their professional duties. The mean age of the participants was 31 years, ranging from 25 to 45 with a median of 28. They were all experienced computing professionals with a mean of 11 years of daily computer use, and their professional occupation ranged from an application programmer to the lower management level. None of the participants were associated with the University of Tampere except as alumni.

4.2. Materials

All test tasks were simple queries from a single database table. The data in this table was the cars data set from the 1983 ASA Data Exposition (Ramos and Donoho, 1983), introduced in Section 2. Since all the participants had a Finnish driver's license—the obtaining of which requires taking classes about the basic mechanisms of a car—it was assumed that the domain knowledge would be adequate. The conceptual difference in expressing fuel consumption (miles per gallon versus liters per 100 km) was discussed with the participants.

A set of 16 ecologically relevant tasks for the cars data set was designed. We can describe the test tasks in terms of SQL operations as follows: three were simple SELECT-operations, another three were either multiple SELECT-operations or SELECT-operations involving sub-selections, and the rest were selections involving simple grouping. Another characterization for the tasks can be given using information levels (Bertin, 1981): three of the tasks could be classified as an elementary level question (e.g. task 6), nine of the tasks were intermediate level questions (e.g. task 9), and four of the tasks were overall questions from the data set (e.g. task 11).

The test tasks were:

(1) How many cars have a four or six cylinder engine?
(2) What is the average HP of cars having fuel economy of 40 MPG or better?
(3) What is the origin of the six cylinder cars that were manufactured in 1971?
(4) What can you say about the weight of Japanese cars compared to overall vehicle weights?
(5) Which one of the cars manufactured in 1982 has the slowest acceleration?
(6) What is the weight of a 'Saab 900'?
(7) How would you characterize the cars that weigh over 4500 pounds?
(8) How many American cars are there in the database?
(9) Which Japanese cars have the best acceleration?
(10) How would you characterize the three-cylinder cars?
(11) What else is common to the most powerful, best accelerating, and heaviest vehicles in the database?
(12) What is the most common number of cylinders for vehicles manufactured in 1973?
(13) In 1981, only one eight-cylinder car was manufactured. How does it compare to the other cars manufactured in the same year?
(14) What is the average mileage for the six-cylinder cars?
(15) Do the cars that weigh between 3000 and 3500 pounds have anything else in common?
(16) How would you characterize the acceleration performance of a 'Peugeot 505 turbo'?

The SQL tasks were performed with psql, the textual user interface for the relational database management system PostgreSQL (Open-source community, 2005). The build of PostgreSQL used in the experiment included the readline library, so the subjects could retrieve their previous inputs, edit them, and re-run them with ease. The syntax and behavior of the PostgreSQL implementation of SQL is well in accordance with the SQL standard.

The older prototype of the Parallel Coordinate Explorer (Siirtola, 2000b) was used as an alternative user interface for the data set. It is a Java applet implementation of a parallel coordinate browser and its user interface relies on direct manipulation.

4.3. Procedure

The tests were carried out in a usability laboratory with two operators. The role of the first operator was to instruct and assist the subject, and the second operator was an observer during the test and an interviewer after the test. The screen of the subject PC was videotaped along with the audio.

Each test session began with a general description of the test and its purpose, and the subjects had a chance to ask questions. Subjects were also instructed to perform the tasks as quickly as possible, but keep the correctness of the result as their top priority.

The test tasks were divided into two groups, and the subjects were randomly assigned to perform one group of tasks with the SQL interface and the other group with the parallel coordinate browser. The order of user interfaces was fixed: the SQL tasks were always performed first, because the SQL interface was already familiar to the participants. The tasks were divided into two groups in such a way that the groups contained approximately equally difficult and structurally similar sets of tasks, and the sets were counterbalanced.

The SQL tasks were preceded with a brief tutorial of the psql program and the subjects had a chance to experiment for 5–10 min with the interface and to ask clarifying questions. In addition, the subjects were told that they could use as many queries as they wished to solve a task. Although the psql program has a good online documentation, a single sheet document with the full syntax of the SELECT statement was prepared for the subjects to refer to.

The operator gave the tasks one at the time to the participant on a paper slip. The participant read the task, performed it, voiced the answer, and passed the task slip back to the moderator. The time that the participant kept possession of the task slip was recorded as the task execution time.

After the SQL task session, the subjects had a chance to take a pause if so desired. The parallel coordinate browser session was then carried out in the same manner as the preceding SQL session. The length of the experimental sessions varied from half an hour to an hour and a half. There were no time constraints for solving the tasks.

Three kinds of data were collected for analysis: response times, answers given, and the observations about the task progress. These observations included the number of queries formulated, possible problems with the user interface, and the comments that the participants made during the experiment.

4.4. Results

The designed size of the input was 16 participants×7 tasks×2 user interfaces = 224 observations. From each task, the task completion time and the given answer were recorded.

4.4.1. Task completion times

The task completion times are presented as box-and-whiskers plots (Tukey, 1977, p. 39). The 'box' extends from the 25% to the 75% quartile (*IQR*), the thicker line inside the box shows the median value, and the whiskers extend to values if they reside within 1.5×*IQR* from the median. Finally, the values not fitting into the distribution are marked with a circle.

As expected, there are tasks that are faster with the SQL interface and tasks that are faster with the parallel coordinate browser. Fig. 17 shows the overall distribution of the task completion times. The lower left corner has a display that shows the average difference in task completion times between the two interfaces. The bars

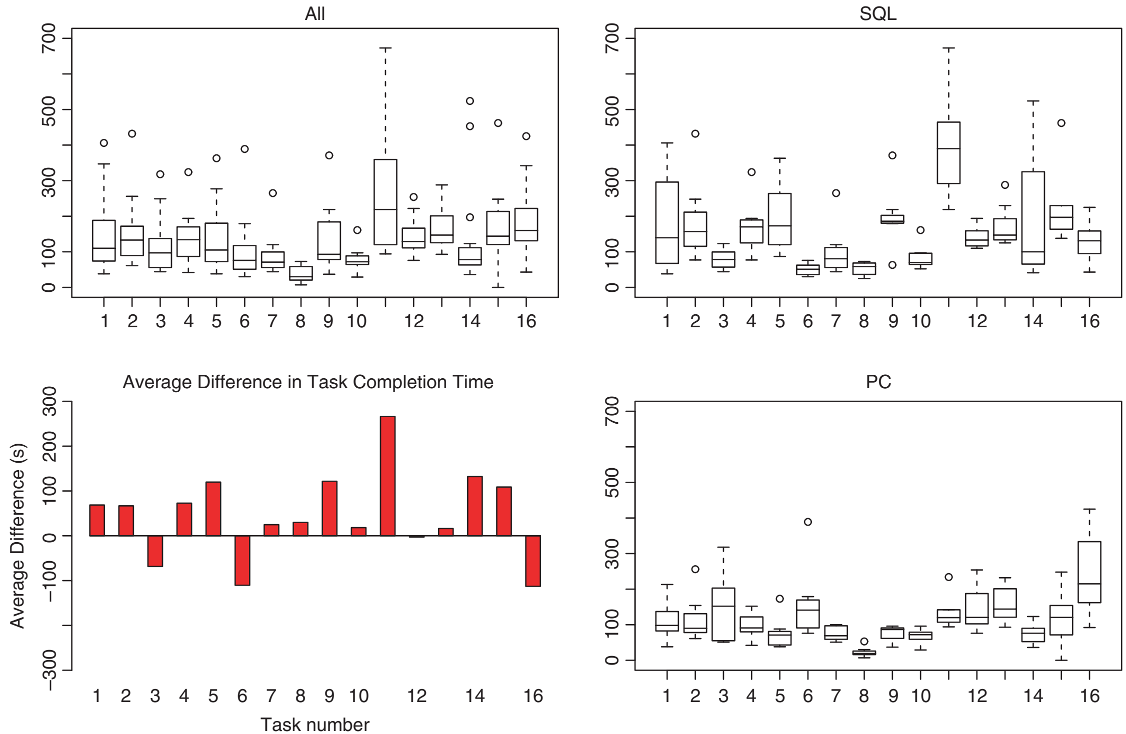


Fig. 17. Overall distribution of the task completion times. From the left to right, top to bottom: distribution of completion time per task (All), distribution of completion times for SQL, average difference in task completion times between conditions, and distribution of completion times for parallel coordinates (PC).

extending upwards present situations where the parallel coordinate browser interface was more efficient than the SQL interface, on the average. As can be seen, the SQL interface was faster in tasks 3, 6, and 16, and the parallel coordinate interface was better in tasks 1, 2, 4, 5, 9, 11, 14, and 15. In tasks 7, 8, 10, 12, and 13 the difference is negligible.

The overall grand mean of task completion time was 112 s for the parallel coordinate browser and 165 s for the SQL interface. The observed difference is also statistically significant ($t = -3.915$, $\mathrm{df} = 201.02$, $p < .001$, without assuming equal variance). The group did not have significant effect. Because of the small sample size it is not appropriate to open the analysis to the task level.

4.4.2. Correctness of retrieved information

The correctness of the retrieved information was rated as a percentage. Many of the tasks could only receive 0 or 100% (the answer was correct or incorrect), but the characterization tasks were graded on a finer scale. The overall success mean was 62% for the parallel coordinate browser and 63% for the SQL interface, but the difference was neither practically nor statistically significant.

4.4.3. Subjective ratings

The subjective satisfaction of participants was evaluated with a questionnaire and with an informal interview. The questionnaire presented the propositions listed in Appendix A to the participants, and the answer was given on a six-point Likert scale ranging from ‘Disagree completely’ to ‘Agree completely’. There was no neutral answer in the middle of the scale: the participant had to choose either ‘Disagree somewhat’ or ‘Agree somewhat’. The detailed results are represented as box-plots in Appendix A.

In the post-test interview, the subjects were encouraged to comment on the parallel coordinate technique and browsing. The most common comment was that the parallel coordinate plot looks initially complex, but the underlying idea and the interaction is easy to learn. The most frustrating feature in the user interface was the adjustment of multidimensional brushes. Adjusting the query parameter with mouse was found tedious when the desired exact value was known.

4.5. Discussion

Overall, the results suggest that the user interfaces based on the parallel coordinate visualization are not as difficult to use as generally believed. In addition, the parallel coordinate browser interface was substantially faster in solving this set of tasks, and there was no measurable difference in the accuracy of the answers.

The parallel coordinate browser used in the experiment had a flaw that could not be fixed in the middle of scheduled experiments. The flaw left the values on a nominal axis in a random order, making it very difficult to find a specific value. This affected tasks 6 and 16 where one had to find a car by its name. However, as soon as the participants realized the flaw, they used the other dimensions to reduce the set of possible polylines into smaller set that could be inspected one by one. The data analysis was made with tasks 6 and 16 included, but we discuss the results both with tasks 6 and 16 included and excluded in this section.

The time difference with these tasks and with these experienced users of the SQL query language was about 32% (43% without tasks 6 and 16) in favor of the parallel coordinate browser. These conclusions are in line with the earlier comparisons of graphical query languages and the textual query language SQL (Catarci and Santucci, 1995a,b), although the parallel coordinate approach is more general than a typical graphical query language.

The correctness of users' task performance was 62–63% in this experiment (removing tasks 6 and 16 did not have an effect). Kobsa (2001) did a comparison of three commercial multidimensional data visualization tools (Eureka, InfoZoom, and Spotfire) with three different data sets (one of them being the cars data set). In that experiment, the task completion time varied between 80 and 110 s and the task correctness varied between 68 and 75%. In comparison, it seems that a commercially developed parallel coordinate browser could be on a par with other visualization tools for multidimensional data, particularly considering the generality of the technique: if a database can be represented as a universal relation, it can be visualized as a parallel coordinate plot.

Of specific interest in our experiment was if the parallel coordinate plots have usability problems because of the abstract nature of the technique. The post-test questionnaire gives some evidence against this. Proposition 1, 'The parallel coordinate browser is easy to use', was rated between 'Agree' and 'Agree somewhat' with two exceptions. The claim that the parallel coordinate plots are difficult to read was disagreed 14 against 2, the median being 'Disagree'. Proposition number 11, 'The parallel coordinate plot looks confusing', was evaluated from 'Agree' to 'Disagree' with median 'Disagree somewhat', and this opinion was also voiced during the experiments. There was a comment that this claim is difficult to evaluate, since the parallel coordinate plots look initially confusing, but then 'you get used to them'. This can also be seen in the outcome of proposition 'It is easy to grasp the idea of parallel coordinate plots' where the median was 'Agree' with only one participant mildly on the disagreement side.

One of the themes in the questionnaire was to solicit opinions from the database professionals on the relative merits of the SQL query language and a parallel coordinate browser. The claim 'It is easier to formulate a query in SQL than in the parallel coordinate browser' got a dispersed response, but the median was on the disagreement side. Thus many of the participants saw the parallel coordinate approach as a viable alternative. The proposition 'I don't believe a graphical query language like in the parallel coordinate browser can ever replace the SQL language' got a mixed response, with five participants disagreeing and 11 agreeing, and the median being 'Agree somewhat'. The professionals also suspected that the parallel coordinate approach is not viable with more complex queries, as the 'It is cumbersome to write complex queries with the parallel coordinate browser' claim shows. The median was 'Agree', although almost half of the answers were on the disagreement side.

The ability to display the whole database as a single image was often mentioned as a strength of the parallel coordinate technique. This can also be seen from the outcome of proposition 9, ‘A parallel coordinate plot gives a good overview of the database contents’, that got rated mainly from ‘Agree’ to ‘Agree completely’.

The most common new feature request that came up during the experiments and the post-experiment interviews was the ability to edit the limits of query ranges in textual form. The participants felt that it was awkward to adjust the limit with a mouse when you know the exact value you want to set. Our more recent implementations include this feature. The queries and their limits are represented as tables below the parallel coordinate plot, and the limits of ranges can be modified both directly and indirectly (see Fig. 16).

5. Conclusion

We have presented the parallel coordinate technique for visualizing multidimensional data. Several possible ways to interact with the visualization were proposed for augmenting the knowledge acquisition process. An empirical study was carried out for exploring the learnability of the interaction techniques and for comparing interactive parallel coordinate visualizations to a standard textual query language.

Interacting with parallel coordinate visualizations can be made easier by implementing parallel coordinate browsers with better interaction tools. At present, there is no single parallel coordinate browser that would provide all the features discussed in Section 3. We do not claim that a browser should necessarily have all the features, but some of them seem to be more critical than the others: representing query constraints as persistent and modifiable objects, allowing easy axis rearrangement, and providing at least some kind of tools to fight occlusion.

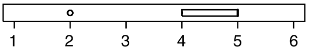
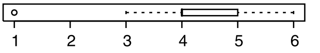
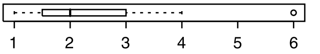
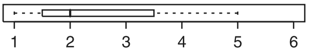
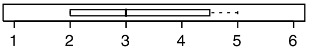
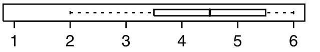
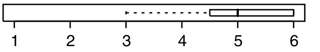
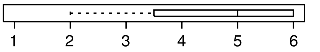
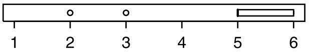
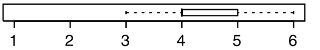
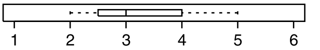
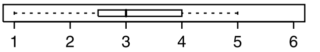
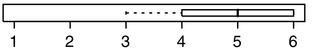
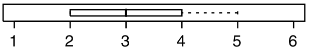
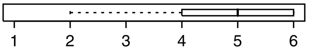
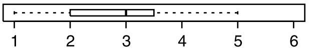
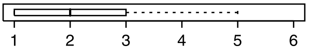
Interacting with parallel coordinate visualizations is not so difficult as is generally believed, as our empirical study suggests. The visualization looks abstract and complex, but as soon as the users have a chance to interact with it, they grasp the fundamental idea behind it. The immediate usability of parallel coordinate visualizations seems surprisingly good if we consider the challenges of a typical information acquisition or exploration task with a multidimensional data set.

There are several interesting open problems in the parallel coordinate interaction. Perhaps the most important one is the occlusion problem. Many solutions have been suggested, but very little is known of how the solutions relate to each other, or how usable they are. Another interesting issue is the direct manipulation of parallel coordinate visualizations. Choices between providing direct, indirect, or both variations of operations need further studies, as does the handling of missing data in parallel coordinate plots.

Acknowledgements

We thank Ms Kati Kivelä, BSc, for assisting in the experiments and for recruiting the participants. We also thank Mr Tomi Heimonen, MSc, for his valuable comments on the manuscript.

Appendix A. Questionnaire results

No.	Proposition	Disagree Agree
1	The parallel coordinate browser is easy to use	
2	I could use a user interface based on parallel coordinates in my work	
3	The parallel coordinate plots are difficult to read	
4	It is easy to brush a range in a parallel coordinate browser	
5	It is easier to formulate a query in SQL than in the parallel coordinate browser	
6	I would be willing to use a parallel coordinate browser in my work	
7	It is easy to grasp the idea of the parallel coordinate plots	
8	It is useful to simplify a set of polylines with a line traversing through the average values	
9	A parallel coordinate plot gives a good overview of the database contents	
10	The SQL query language is easy to learn	
11	A parallel coordinate plot looks confusing	
12	The syntax of SQL language is easy to remember	
13	It was easy to learn the use of parallel coordinate browser	
14	The parallel coordinate browser gives sufficient feedback to the user	
15	It is easy to understand a SQL query written by someone else	
16	The feedback of operations in the parallel coordinate browser is fast enough	
17	It is cumbersome to write complex queries with the parallel coordinate browser	
18	I don't believe that a graphical query language like in the parallel coordinate browser can ever replace the SQL language	
19	There is too much information in a parallel coordinate plot	
20	It is easy to get an overview of an unfamiliar database with the SQL language	

References

Andrienko, G., Andrienko, N., 2004. Parallel coordinates for exploring properties of subsets. In: Proceedings of the Second International Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV'04). IEEE Computer Society, pp. 93–104.

Ankerst, M., Berchtold, S., Keim, D.A., 1998. Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In: Proceedings of the IEEE Symposium on Information Visualization 1998 (InfoVis'98). IEEE Computer Society, pp. 52–60.

Artero, A.O., Ferreira de Oliveira, M.C., Levkowitz, H., 2004. Uncovering clusters in crowded parallel coordinates visualizations. In: Proceedings of the IEEE Symposium on Information Visualization (Info Vis'04). IEEE Computer Society, pp. 81–88.

Austel, V., Chatterjee, A., Inselberg, A., 1996. System and method for visually querying a data set exhibited in a parallel coordinate system. US Patent #5,546,516. URL: http://patft.uspto.gov/netacgi/nph-Parser?patentnumber = 5546516

Bertin, J., 1981. Graphics and Graphic Information-Processing. Walter de Gruyter, Translated by William J. Berg and Paul Scott.

Bertin, J., 1983. Semiology of Graphics. University of Wisconsin Press, Translated by William J. Berg.

Buja, A., Cook, D., Swayne, D., 1996. Interactive high-dimensional data visualization. Journal of Computational and Graphical Statistics 5 (1), 78–99.

Card, S.K., Mackinlay, J.D., Shneiderman, B., 1999. Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann.

Catarci, T., Santucci, G., 1995a. Are visual query languages easier to use than traditional ones? An experimental proof. In: In: People and Computers X, International Conference on Human–Computer Interaction (HCI95). Cambridge University Press, Cambridge, pp. 323–338.

Catarci, T., Santucci, G., 1995b. Diagrammatic vs. textual query languages: a comparative experiment. In: Proceedings of the Third IFIP 2.6 Working Conference on Visual Database Systems (VDB). Chapman & Hall, New York, pp. 69–83.

Chou, S.-Y., Lin, S.-W., Yeh, C.-S., 1999. Cluster identification with parallel coordinates. Pattern Recognition Letters 20 (6), 565–572.

Eaton, C., Plaisant, C., Drizd, T., 2005. Visualizing missing data: classification and empirical study. In: Proceedings of INTERACT 2005. LNCS 3585. Springer, Berlin, pp. 861–872.

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

Fua, Y.-H., Ward, M. O., Rundensteiner, E. A., 1999. Hierarchical parallel coordinates for exploration of large datasets. In: Proceedings of the IEEE Visualization '99 (VIS'99). IEEE Computer Society, pp. 43–50.

Gibson, J.J., 1962. Observations on active touch. Psychological Review 69 (6), 477–491.

Gibson, J.J., 1983. The Senses Considered as Perceptual Systems. Connecticut: Greenwood Press Reprint (Originally Boston: Houghton-Mifflin, 1966).

Goel, A., Baker, C., Shaffer, C.A., Grossman, B., Haftka, R.T., Mason, W.H., Watson, L.T., 1999. VizCraft: a multidimensional visualization tool for aircraft configuration design. In: Proceedings of the IEEE Visualization '99 (VIS'99). IEEE Computer Society Press, pp. 425–428.

Graham, M., Kennedy, J., 2003. Using curves to enhance parallel coordinate visualisations. In: In: Proceedings of the Seventh International Conference on Information Visualization, (IV'03). IEEE Computer Society, pp. 10–16.

Gresh, D., Rogowitz, B., Winslow, R., Scollan, D., Yung, C., 2000. WEAVE: a system for visually linking 3-D and statistical visualizations, applied to cardiac simulation and measurement data. In: Proceedings of the IEEE Visualization '00 (VIS'00). IEEE Computer Society, pp. 53–60.

Hauser, H., Kosara, R., 2004. Interactive analysis of high-dimensional data using visualization. In: Workshop on Robustness for High-dimensional Data 2004 (RobHD 2004). Extended abstract, published also as a technical report TR-VRVis-2004-004.

Hauser, H., Ledermann, F., Doleisch, H., 2002a. Angular brushing of extended parallel coordinates. In: In: Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02). IEEE Computer Society, pp. 127–131.

Hauser, H., Ledermann, F., Doleisch, H., 2002b. Angular brushing of extended parallel coordinates. Tech. Rep. TR-VRVis-2002-015, VRVis Research Center in Vienna.
Hibbard, B., 1999. Top ten visualization problems. Computer Graphics 33 (2), 21–22.
Inselberg, A., 1981. N-dimensional Graphics Part I: Lines & Hyperplanes. Tech. Rep. #G320-2711, IBM Scientific Center, Los Angeles, CA.
Inselberg, A., 1985. The plane with parallel coordinates. The Visual Computer (Special issue on Computational Geometry) 1, 69–97.
Inselberg, A., 1989. *N*-dimensional information display method for air traffic control. US Patent #4,823,272. URL: http://patft.uspto.gov/netacgi/nph-Parser?patentnumber = 4823272
Inselberg, A., 1997a. Multidimensional detective. In: Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis'97). IEEE Computer Society, pp. 100–107.
Inselberg, A., 1997b. Visualizing multidimensional geometry with applications to multivariate problems. Abstracts of Papers of the American Chemical Society 214, 202, Comp.
Inselberg, A., 1998. Visual data mining with parallel coordinates. Computational Statistics 13 (1), 47–63.
Inselberg, A., 1999. Don't panic ... just do it in parallel! Computational Statistics 14 (1), 53–77.
Inselberg A., 2000. Visualizing high dimensional datasets and multivariate relations (am-2). Tutorial notes from KDD'00: Sixth ACM SIGKDD international conference on Knowledge Discovery and Data Mining.
Inselberg, A., 2001. Automated knowledge discovery using parallel coordinates. Data Viz II Workshop. George Mason University.
Inselberg, A., 2002. Visualization and data mining of high-dimensional data. Chemometrics and Intelligent Laboratory Systems 60 (1–2), 147–159.
Inselberg, A., Avidan, T., 1999. The automated multidimensional detective. In: Proceedings of the 1999 IEEE Symposium on Information Visualization (InfoVis'99). IEEE Computer Society, pp. 112–119.
Inselberg, A., Dimsdale, B., 1990. Parallel coordinates: a tool for visualizing multidimensional geometry. In: In: Proceedings of the IEEE Visualization'90 (VIS'90). IEEE Computer Society, pp. 361–370.
Inselberg, A., Dimsdale, B., 1994a. Multidimensional lines 1: representation. SIAM Journal on Applied Mathematics 54 (2), 559–577.
Inselberg, A., Dimsdale, B., 1994b. Multidimensional lines 2: proximity and applications. SIAM Journal on Applied Mathematics 54 (2), 578–596.
Inselberg, A., Chomut, T., Reif, M., 1987. Convexity algorithms in parallel coordinates. Journal of the ACM 34 (4), 765–801.
Inselberg, A., Eickemeyer, J.S., Hurwitz, A.A., 1992. Motion constraints using particles. US Patent #5,173,861. URL: http://patft.uspto.gov/netacgi/nph-Parser?patentnumber = 5173861
Inselberg, A., Chatterjee, A., Dimsdale, B., 1997. System using parallel coordinates for automated line detection in noisy images. US Patent #5,631,982. URL: http://patft.uspto.gov/netacgi/nph-Parser?patentnumber = 5631982
Inselberg, A., Grinstein, G., Laskowsk, S., 1998. Panel: Key problems and thorny issues in multidimensional visualization. In: Proceedings of the IEEE Visualization'98 (VIS'98), pp. 505–506.
IV Society, 2005. Information Visualisation Society Newsletter, Spring 2005.
Johansson, J., Treloar, R., Jern, M., 2004. Integration of unsupervised clustering, interaction and parallel coordinates for the exploration of large multivariate data. In: In: Proceedings of the Eighth International Conference on Information Visualization (IV'04). IEEE Computer Society, pp. 52–57.
Kandogan, E., 2001. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, pp. 107–116.
Kobsa, A., 2001. An empirical comparison of three commercial information visualization systems. In: Proceedings of the IEEE Symposium on Information Visualization 2001 (InfoVis'01). IEEE Computer Society, pp. 123–130.
Ledermann, F., 2003. Parvis—a tool for parallel coordinates (PC) visualisation of multidimensional data sets. URL: http://home.subnet.at/flo/mv/parvis/
Liu, K., Zhou, D., Zhou, X., 2002. Clustering by ordering density-based subspaces. In: Second International Workshop on Visual Data Mining (VDM, in conjunction with ECML/PKDD2002).

Martin, A.R., Ward, M.O., 1995. High dimensional brushing for interactive exploration of multivariate data. In: Proceedings of the Sixth Conference on Visualization '95 (VIS'95). IEEE Computer Society, pp. 271–278.
Novotny, M., 2004. Visually effective information visualization of large data. In: Eighth Central European Seminar on Computer Graphics (CESCG'04).
Open-source Community, 2005. PostgreSQL database management system. URL: II http://www.postgresql.org/
R Development Core Team, 2005. R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL: http://www.R-project.org/
Ramos, E., Donoho, D., 1983. The 1983 ASA data exposition dataset: Cars. URL: http://lib.stat.cmu.edu/datasets/cars/data
Rosario, G.E., Rundensteiner, E.A., Brown, D.C., Ward, M.O., 2003. Mapping nominal values to numbers for effective visualization. In: Proceedings of the IEEE Symposium on Information Visualization 2003 (InfoVis'03). IEEE Computer Society, pp. 113–120.
Rundensteiner, E.A., Ward, M.O., Yang, J., Doshi, P.R., 2002. XmdvTool: visual interactive data exploration and trend discovery of high-dimensional data sets. In: In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data. ACM Press, p. 631.
Shneiderman, B., 1983. Direct manipulation: a step beyond programming languages. IEEE Computer 16 (8), 57–69.
Shneiderman, B., 1996. The eyes have it: a task by data type taxonomy for information visualizations. In: Proceedings of the 1996 IEEE Symposium on Visual Languages (VL'96). IEEE Computer Society, pp. 336.
Shneiderman, B., 1998. Designing the user interface: strategies for effective human–computer interaction, 3rd ed. Addison Wesley Longman, Reading, MA.
Siirtola, H., 2000a. Direct manipulation of parallel coordinates. In: Proceedings of the International Conference on Information Visualization (IV'2000). IEEE Computer Society, pp. 373–378.
Siirtola, H., 2000b. Parallel Coordinate Explorer. URL: http://www.cs.uta.fi/~hs/pce/
Siirtola, H., 2003. Combining parallel coordinates with the reorderable matrix. In: In: Proceedings of the International Conference on Coordinated & Multiple Views in Exploratory Visualization. IEEE Computer Society, pp. 63–74.
Spence, R., 2001. Information Visualization. ACM Press.
Swayne, D., Temple, D., Buja, A., Cook, D., 2001. GGobi: XGobi redesigned and extended. In: 33th Symposium on the Interface: Computing Science and Statistics.
Theus, M., 2002. Interactive data visualization using Mondrian. Journal of Statistical Software 7 (11).
Tukey, J., 1977. Exploratory Data Analysis. Addison-Wesley, Reading, MA.
Unwin, A., Volinsky, C., Winkler, S., 2003. Parallel coordinates for exploratory modelling analysis. Computational Statistics & Data Analysis 43 (4), 553–564.
Ware, C., 2004. Information Visualization: Perception for Design, Second ed. Morgan Kaufman, San Francisco.
Wegman, E.J., 1990. Hyperdimensional data analysis using parallel coordinates. Journal of the American Statistical Association 85 (411), 664–675.
Wegman, E.J., 1999. Data mining and visualization: some strategies. Bulletin of the International Statistical Institute Tome LVIII (Book 3), 223–226.
Wegman, E.J., Luo, Q., 1997. High dimensional clustering using parallel coordinates and the grand tour. Computing Science and Statistics 28, 352–360.
Yuan, H., Sachtler, W.L., Durlach, N., 2000. Effects of time delay on depth perception via head-motion parallax in virtual environment systems. Presence 9 (6), 638–647.
Zhao, K., Liu, B., Tirpak, T.M., Schaller, A., 2003. Detecting patterns of change using enhanced parallel coordinates visualization. In: In: Proceedings of the Third IEEE International Conference on Data Mining. IEEE Computer Society, pp. 747–750.