Augmenting Parallel Coordinates Plots With Color-Coded Stacked Histograms

Jinwook Bok, Bohyoung Kim, and Jinwook Seo

Abstract—We introduce Parallel Histogram Plot (PHP), a technique that overcomes the innate limitations of parallel coordinates plot (PCP) by attaching stacked-bar histograms with discrete color schemes to PCP. The color-coded histograms enable users to see an overview of the whole data without cluttering or scalability issues. Each rectangle in the PHP histograms is color coded according to the data ranking by a selected attribute. This color-coding scheme allows users to visually examine relationships between attributes, even between those that are displayed far apart, without repositioning or reordering axes. We adopt the Visual Information Seeking Mantra so that the polylines of the original PCP can be used to show details of a small number of selected items when the cluttering problem subsides. We also design interactions, such as a focus+context technique, to help users investigate small regions of interest in a space-efficient manner. We provide a real-world example in which PHP is effectively utilized compared with other visualizations, and we perform a controlled user study to evaluate the performance of PHP in helping users estimate the correlation between attributes. The results demonstrate that the performance of PHP was consistent in the estimation of correlations between two attributes regardless of the distance between them.

Index Terms—Parallel coordinates plots, parallel histogram plots, color-coded stacked histogram

1 Introduction

Parallel coordinates plot (PCP) [17] is a visualization technique that arranges multiple attributes parallel to each other in a 2D plane. Clusters of data items and relations between attributes, including correlations, can be perceived by the patterns of lines in PCP. This pattern recognition becomes harder, however, when lines overlap more with each other as the number of items and attributes increases. Furthermore, relationships between attributes are difficult, if not impossible, to infer from visual patterns in PCP when the axes are not adjacent. Many approaches have been proposed to deal with these limitations, e.g., the overplotting of lines or the ordering of axes [16]; however, there are still many challenges that researchers have to face when visualizing data with PCP because of the innate limitations of the original PCP. Sometimes, the limitations have been resolved by sacrificing the original structure of PCP, which significantly weakens its perceptual advantages.

In this paper, we introduce Parallel Histogram Plot (PHP), a visualization technique that deals with the innate limitations of PCP while preserving its perceptual advantages and characteristics. Following the Visual Information-Seeking Mantra [39], we augment the original polylines of PCP with color-coded stacked bar histograms. Attached to each axis of the original PCP layout, the histograms provide a scalable overview by showing the distribution of data items of each attribute. Polylines of PCP are used in the later stages of the Visual Information Seeking process, when the cluttering problem is less severe after less important items have been filtered out. Colors applied to the stacked bars of histograms are determined by a user-selected attribute. Visual comparison of the color distributions on histograms for multiple attributes reveals relationships between the attributes without cluttering or overlapping of lines as in PCP. Relationships between distant attributes that are hard, if not impossible, to grasp in the original PCP can be readily perceived in PHP through the visual comparison of color distributions for the attributes. Improving upon our previous work [2], we also designed interaction idioms for PHP to help users investigate the details of histograms in a limited screen space.

We illustrate a use case with a real-world dataset that shows the advantages of PHP over other visualizations (i.e., PCP, SPLOM, and Angular Histograms (AH) [11]) in discovering hidden patterns of items. We show that the color coded histograms in PHP can complement the original PCP by addressing the challenge of overplotting of lines, thus providing a more scalable overview in the context of the Visual Information Seeking Mantra. We also performed a controlled user study to evaluate how PHP helped users estimate the correlation between two attributes compared with PCP and AH. It was empirically demonstrated that color-coded histograms enable PHP to consistently show satisfying performance in correlation estimation regardless of the distance between attributes, providing a new way of dealing with one of the innate limitations of the original PCP.

We begin by reviewing related work on techniques for enhancing the original PCP, especially those utilizing histograms. Next, we explain how we designed PHP along with our design rationale. We then present the visual information-

- *Jinwook Bok and Jinwook Seo are with the Seoul National University, Seoul 08826, South Korea. E-mail: bok@hcil.snu.ac.kr, jseo@snu.ac.kr.*
- *Bohyoung Kim is with the Hankuk University of Foreign Studies, Seoul 08826, South Korea. E-mail: bkim@hufs.ac.kr.*

Manuscript received 31 Dec. 2019; revised 14 Oct. 2020; accepted 11 Nov. 2020. Date of publication 0 . 0000; date of current version 0 . 0000. (Corresponding author: Bohyoung Kim and Jinwook Seo.) Recommended for acceptance by T. Dwyer. Digital Object Identifier no. 10.1109/TVCG.2020.3038446

seeking process with PHP, focusing on user interactions for exploring data in PHP. After presenting a use case and empirical evaluation results, we discuss the implications of our experimental results, along with the limitations of PHP and potential future work to overcome them.

2 Related Work

This section introduces previous research upon which we built our visualization and interaction idioms in this study. We first summarize previous approaches to enhancing the performance of PCP. Then, we introduce Attribute and Influence Explorer [40], [41] which inspired our work through the way it utilizes histograms of stacked elements.

2.1 Approaches to Enhancing PCP

Many efforts have been devoted to enhancing the performance of PCP. They have focused mainly on making it more scalable by reducing visual clutter. These efforts can be grouped into four categories according to the types of techniques employed: reduction, transformation, integration, and interaction.

Reducing the Number of Items displayed on PCP is a popular approach. Such a reduction results in fewer polylines, thus leading to less cluttering. Reducing the number of items to show is done mostly by aggregating (or compressing) the data. Because it is subject to information loss, maintaining the characteristics of the original data as much as possible has been the main concern of this approach. Various data reduction methods have been introduced, including clustering [10], [22], binning [31], sampling [7], and image-space methods based on image processing algorithms [1]. The number of dimensions can also be reduced by using dimension hierarchy [46], by measuring the distance between attributes [18], and by contracting adjacent axes [30]. Finally, without any form of data reduction, the ordering of the dimensions can be changed for a more orderly overview [6], [26], [27], [33], [48]. Reordering attributes is critical in PCP because it is impossible to investigate relationships between nonadjacent attributes.

Transforming the Components (polylines and axes) of PCP is another well-known approach. Bezier curves can replace the polylines of PCP, which are more appropriate for bundling and thus more suitable for presenting clusters of items [15], [32], [45]. Other methods of transforming the polylines include density fields [14], polygons [28], bands [25], and layers of consistency maps [29]. These substituted visual elements can serve a specific purpose better compared with the original polylines. For example, in the method of Parallel Sets [25], the lines are replaced with bands to effectively visualize categorical variables. The axes of PCP are also targets for transformation. The arrangement of the axes can go beyond the 1D linear one. They can be arranged on a 2D plane [3] or in a 3D space [5], [20] to enable users to examine relationships among multiple attributes. The axes can even be transformed into curves [34] to show the data in a polar coordinate system, or be tilted by the tension of lines in PCP [44]. When applying these approaches, we should consider the trade-off, in that we could lose the strong perceptual advantage of the original visual encoding of PCP in correlation estimation by (line-crossing) pattern recognition.

Integrating Other Visualizations with PCP can facilitate the visual information-seeking process by revealing different facets of the data that are difficult to grasp only from the patterns of polylines. Scatterplots that display the relationship between two attributes are popular visualizations for such a purpose [35], [47]. Stacked bar charts [19] and histograms [11] attached to an axis of the PCP can show the distribution of each attribute. Other visualization idioms can also be integrated with PCP, including star glyphs [8], box plots [19], [24], spherical coordinate systems [43] and MDS plots [12]. In PHP, we integrate color-coded histograms with PCP to visualize the overview of multiple attributes in a scalable manner. Color-encoding acts as an important channel that reveals the relationship between attributes, even when they are not adjacent to each other.

There have been a few attempts to use color as an auxiliary channel for delivering additional information, such as the distribution of values of an attribute. In Value-cell bar charts [23], bars are split into multiple cells that correspond to one or more individual values. The cells are color coded by the sum of the values. From the color distribution made by each of the cells, the distribution of the values in each of the bars can be inferred. Janetzko *et al.* [19] utilized color-coded stacked bar charts on the axes of PCP to show clusters generated by K-means clustering. Geng *et al.* [11] used color in Angular Histograms for redundantly encoding the height of the tilted bars to help users perceive the height more accurately. In contrast to these approaches, our method uses the color channel for showing the linear relationships between attributes. Using colors in histograms attached to axes makes it possible for users to grasp linear relationships between (even distant) attributes through implicit connections made by perceptually matching colors. Unlike [23], which uses colors to reveal the distribution of values within a single bar chart, PHP uses colors to reveal the relationship between multiple distributions or attributes. This approach enables the analysis of data with multiple attributes. Compared with [19], in which colors are mapped onto the groups generated by a clustering algorithm, our approach is more universally applicable and provides the direct relationships between attributes. In addition, compared with [11], in which histograms must be tilted, our approach preserves the original shape of the histograms to prevent users from getting confused by the distortion. Also, the color channel is used as a pivotal channel that reveals the relationships between attributes in PHP, rather than as a redundant channel as in [11].

Interaction Techniques also help users find information with PCP by facilitating the exploration process. Lack of interaction in PCP is known to discourage users from drawing information from the visualization [21]. The Angular Brushing technique enables users to filter data by the value of the angle between the line and the axis in PCP [13]. Roberts *et al.* designed a sketch-based brushing for high-dimensional pattern searches and a data-dependent smart brushing based on metadata [37]. When a visualization is integrated into PCP, novel interactions are often designed to make it work harmoniously with PCP. For instance, OPCP, a visualization technique that integrates a scatterplot-based visualization into PCP, has a dedicated interaction named O-brushing for facilitating pattern selection in complex data

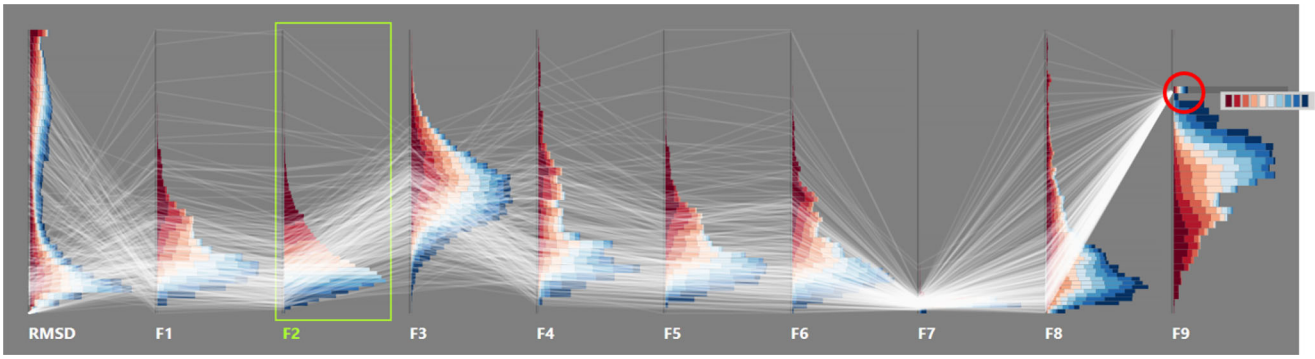


Fig. 1. Parallel Histogram Plots (PHP) used to draw the CASP dataset [42]. The attribute F2 is selected for color coding. From the color distribution, it can be deduced that F2 is positively correlated with F1, F5, and F6 and negatively correlated with F9. In addition, the data items in the upper-right region (red circle) of the F9 histogram are selected and thus displayed as polylines. The widget on the F9 histogram helps with clicking tiny bars on the histogram.

[35]. In PHP, we also designed interaction techniques that aid users in investigating small regions of a histogram that are too small for details to be seen — e.g., two-level semantic zooming that can enlarge a small selected region of the histogram while maintaining the overall layout of PHP.

2.2 Attribute and Influence Explorer

Attribute Explorer [41] and Influence Explorer [40] are tools for exploring data with multiple attributes using histograms. In both tools, each attribute is represented as a histogram, and they are drawn parallel to each other. The histograms of Attribute or Influence Explorer are constructed as a stack of 'lightbulbs', each lightbulb representing a single data element. When query conditions are set, only the lightbulbs that satisfy the conditions are lit (in Influence Explorer, in addition to being lit/not lit, each lightbulb has a brightness level that is proportional to the number of conditions it satisfies). Users can make comparisons across dimensions by observing the distributions of lit and unlit lightbulbs, or of selected and unselected lightbulbs. When they choose a lightbulb on an attribute, corresponding lightbulbs on other attributes are connected by a polyline in the same way as in PCP.

The histograms of PHP are built upon a similar metaphor, while making it more scalable and informative. In PHP, a single stacked bar that corresponds to an Explorer's lightbulb represents an aggregated group of items. Unlike the lightbulbs, the bars have an additional length property to show the size of the corresponding group. Moreover, each bar has a distinct color to show the similarities or differences between groups. As in Attribute and Influence Explorer, the distribution of the selected items/groups can be compared with the overall distribution. The length of the bars changes according to the number of selected items, which can be compared with the full histogram rendered in the background, as in Fig. 6a. In Fig. 6a, the overall distribution in dark gray can be compared with the distribution of the selected items rendered in colors.

3 Design of PHP

This section explains the design concepts behind PHP (Fig. 1) . We first introduce our design rationale and then explain how the color-coded histograms of PHP are constructed.

3.1 Design Rationale

Our approach focuses on overcoming the limitations of PCP while maintaining its original advantages. PHP is designed to deal with two critical limitations of PCP: (L1) cluttering of polylines and (L2) the difficulty in estimating relationships between nonadjacent axes.

L1 – Cluttering of Polylines

The polyline encoding of PCP helps users recognize clusters/outliers and estimate correlations from visual patterns made from the line crossings. However, such encoding inherently suffers from a scalability issue, in that polylines clutter the view with too many overlapping lines. The problem becomes worse as the data size becomes larger.

L2 – Difficulty in Estimating Relationships Between Non-Adjacent Axes

PCP utilizes a linear layout to display multiple attributes. The linear layout is easy to understand and allows multiple attributes to be displayed in a relatively small area. However, the layout makes it challenging to interpret the relationship between attributes that are not adjacent. Thus, finding an effective order of attributes in PCP has been an important research topic.

To achieve our design goal of overcoming the two main limitations of PCP while preserving its advantages, we attach a histogram to each PCP axis. Histograms can reveal the distribution of data items on each attribute in a scalable manner, irrespective of the data size. It is also a relatively simple visualization that does not require any drastic modification of the original layout of PCP, fulfilling our objective of maintaining the innate advantages of PCP. However, histograms have the limitation that they cannot show any relationships between attributes [11]. To resolve this limitation, we adopt color as a crucial visual channel for expressing the relationships between attributes. We order the data items by a user-selected attribute and split the data items into groups according to the order while ensuring that each group has a similar number of items. We represent each group as a bar and assign a unique color to each group. We then build histograms by stacking color-coded bars, with each bar representing a group. By comparing color distributions on attributes in PHP, users can estimate the relationships not only between adjacent attributes but also between distant ones even without direct connections. The indirect connection provided by colors in PCP is free from cluttering

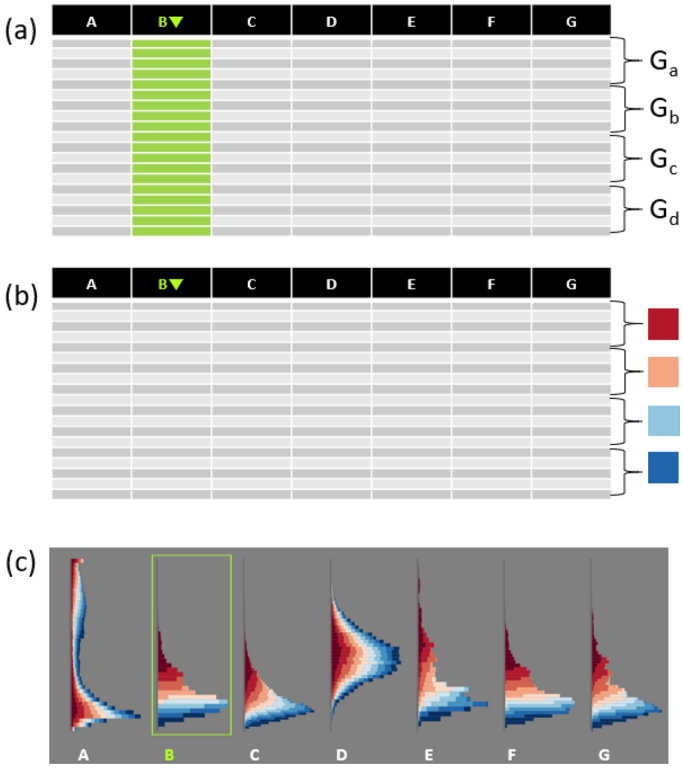


Fig. 2. How PHP is constructed from data. (a) Data items are sorted by a user-selected attribute (attribute B), and items are grouped ($G_a - G_d$) according to the sorted order, i.e., the rank data items of the selected attribute. (b) A unique color is applied to each group, with a diverging colormap: a reddish color for higher ranks and a blueish color for lower ranks. (c) Stacked histograms are rendered in the applied color.

and less influenced by the distance between the attributes. With the adoption of color encoding, the ordering of axes becomes much less important, as the relationship between attributes can be perceived by matching colors even when they are distant from each other. The next section and Fig. 2 show how PHP is built from data.

Following the Visual Information-Seeking Mantra [39], we utilize the original polylines of PCP along with the color-coded histograms. The colored histograms are good at displaying an overview of the data and the relationships between attributes; however, they are not effective in helping users estimate the exact value of each data item. Meanwhile, polylines excel in helping users grasp the value of each data item at an attribute, but they easily suffer from cluttering when there are many of them. Thus, we combine these two components so that they complement each other. In the beginning, color-coded histograms show an overview of the data. After zooming and filtering out less important/relevant data items in the histograms, the polylines show details of a small group of data items selected from the histograms, enabling users to take advantage of the original PCP design.

3.2 Construction of Color-Coded Histograms

3.2.1 Grouping Data Items

To construct the color-coded histograms of PHP, we first split the data into equally sized groups according to a user-selected attribute (Fig. 2a). Instead of using the original data value of the selected attribute to derive groups, we use ranking as the criterion to create groups. First, data items are sorted by the user-selected attribute, and the data items are grouped according to the ranking. Unequally sized groups could occur because we make sure that data items with the same value for the selected attribute are placed together when splitting groups. This prevents data items that have the same value for the selected attribute from being inconsistently placed in different groups. Grouping by ranking mitigates the effects of outliers and skewed distributions in the color mapping, which will be applied to each group in the next step.

3.2.2 Applying Colors to Groups

Second, we apply a unique color to each group of the split data using a discrete, diverging color scheme (Fig. 2b). A discrete color scheme is used, so that a color in the color scheme is assigned to a group. The color scheme is designed to distinguish between groups and to show the differences (or similarities) between them. We adopt this scheme to emphasize low- and high-ranked groups with more saturated colors because they are usually more valuable in the data analysis. In this paper, we use a ten-level blue-red diverging color scheme acquired from ColorBrewer2 [4] (low to high rankings from blue to red). We chose 10 as the number of colors to be rendered, which is close to the number of colors that a human can distinguish simultaneously [19].

3.2.3 Building Stacked-Bar Histograms

Finally, using the preprocessed data (grouped by ranking and then color mapped), we draw a histogram that represents the distribution of each attribute on the corresponding PCP axis (Fig. 2c). The histogram is constructed in the same way as a stacked bar chart is drawn, with each group in its unique color. When groups are stacked, the order of the color-coded elements must agree with the order of the colors in the color scheme so that elements in the same color are merged. This helps users perceive patterns from the color distribution. Users can also recognize the relationship between the selected attribute and others by perceiving the distribution patterns of colors across the histograms. The stacking of elements in PHP makes the layout similar to that of Attribute and Influence Explorer, with a histogram consisting of stacks of lightbulbs that represent individual data items. In PHP, however, a single stacked element represents a group of data, and its length is proportional to the number of data items belonging to the group in the corresponding attribute.

4 Visual Information Seeking With PHP

Using the visual encoding idioms of PHP [1], users can recognize important features in the data, including the distributions of attributes, correlations between attributes, and outliers. We also design interaction idioms combined with the visual encoding idioms, including two-level zooming and ghost bars for more scalable and space-efficient

1. A demo version of PHP is available at https://bokjinwook.github.io/ParallelHistogramPlots/index.html

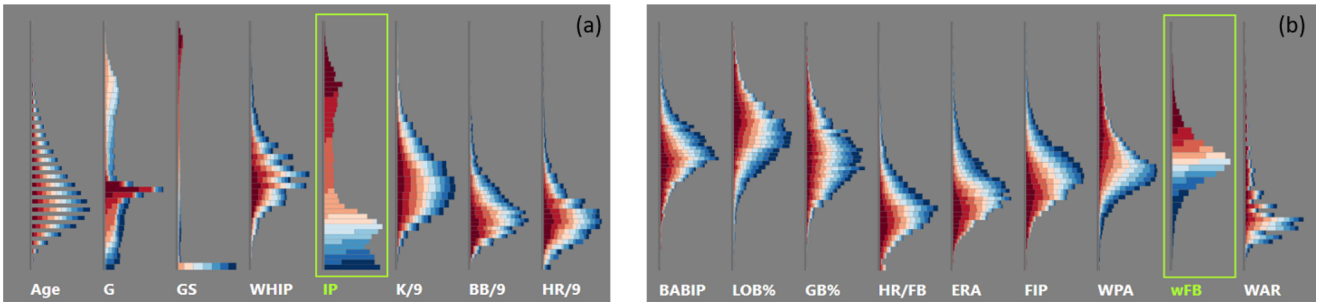


Fig. 3. The Baseball dataset [9] rendered in PHP. The attribute selected for color coding (i.e., the pivot attribute) is bordered by a green rectangle ((a) IP and (b) wFB). Relationships between the pivot attribute and all the others can be observed from how the colors are distributed.

exploration. In this section, we use 16 years of accumulated statistics data of baseball pitchers in Major League Baseball, acquired from FanGraphs [9], as the dataset for ease of explanation. The dataset contains 7,673 items representing each player's record of a year, with 17 sampled attributes reflecting the players' performance.

4.1 Interpreting PHP

PHP utilizes color coding derived from a single user-selected attribute, the so-called pivot attribute (Fig. 3). In PHP, selecting the pivot attribute is a crucial step in revealing features in the data. Visually comparing color distributions on histograms can reveal relationships between the pivot attribute and all the other attributes. Users can steer their data exploration by changing the pivot attribute to seek relationships between attributes with different aspects. This methodology can be used effectively in situations in which only some of the attributes in the dataset are familiar. Users can start their exploration by first selecting a familiar attribute as the pivot attribute and then expand their knowledge from the known to the unknown attributes by analyzing their relationships.

Correlations between the pivot attribute and other attributes of interest can be estimated just by visually comparing the color distributions of the corresponding histograms. For example, in Fig. 4, the attribute X is selected as the pivot attribute, with the color changing from blue to red from bottom to top. The histogram for the attribute Positive shows a color distribution very similar to that of the attribute X, implying a strong positive correlation between these two attributes. But the histogram of the attribute Negative shows a color distribution inverted from that of the attribute X, implying a strong negative correlation between these two attributes. By the same principle, in Fig. 3b, it is easy to recognize that WPA and LOB percent are positively correlated with the pivot attribute wFB. Meanwhile, it can also be easily recognized that BABIP, HR/FB, ERA, and FIP are negatively correlated with wFB, as such visual recognition is not affected by the distance from the pivot attribute.

Users can also recognize clusters of items that have similar color patterns, or outliers that do not follow the major color patterns around them in the histograms. Similar colors gathered in a small region indicate that the data items sharing similar properties are clustered in that region. In Fig. 3a, in which the histograms are color coded by the attribute IP, it can easily be observed that data items with high IP values are tightly gathered in the lower middle of the attribute G. Meanwhile, salient colors indicate that there exist data items outside the overall distribution of those that surround them, suggesting outliers. Fig. 5 displays a magnified histogram of WAR from Fig. 3b. Some data items in the green box have salient blue colors that are different from the overall red surroundings. Compared with most of the items nearby, these outliers have much lower rankings of the pivot attribute wFB, as is apparent from their distinct colors.

Like many other PCP-based visualizations, PHP supports common PCP-based interactions, including selecting items by the range of an attribute's value with brushing and changing the order of axes. In PHP, ordering axes is less important, as the relationship between attributes can be

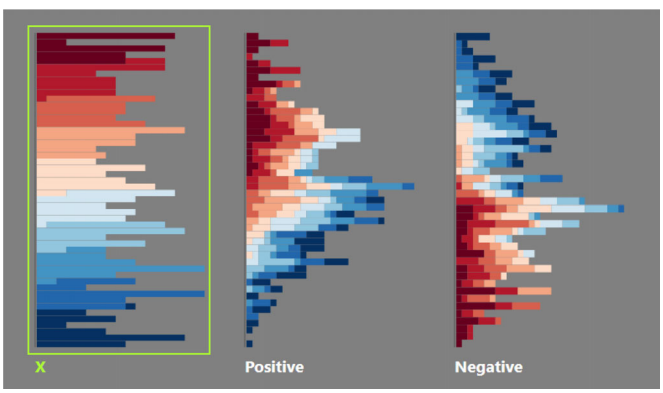


Fig. 4. Example of positive and negative correlations displayed in PHP. The attributes, Positive and Negative have positive (+0.8) and negative (-0.8) correlations with attribute X, respectively.

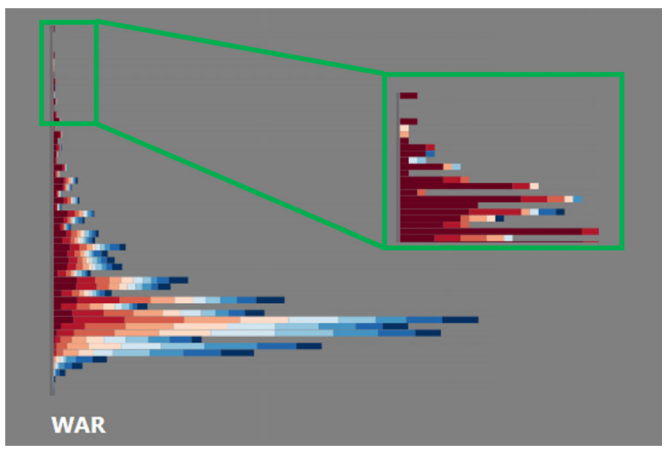


Fig. 5. Histogram of WAR from Fig. 3b. The area inside the green box is magnified for visibility. In the region of the green box, it can be observed that some blue items are distant from the overall red data items.

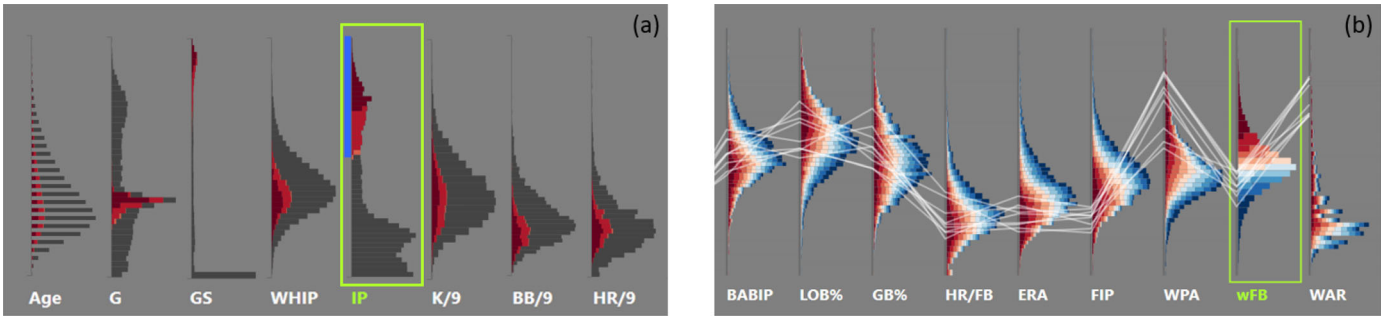


Fig. 6. Displaying selected items in PHP. (a) Data items with high IP values are selected from Fig. 3. The distribution of selected items can be compared with the overall distribution, revealing that the selected items are gathered in the lower-middle narrow region of G. (b) The distinct blue items among the dominant red items in the WAR histogram are selected from Fig. 5 to be displayed by polylines. The polylines reveal detailed characteristics of the selected data items.

observed just by color even when they are far away. Nonetheless, PHP enables users to sort the attributes by correlation or similarity for a more efficient analysis of the data. PHP also supports common interactions/modifications related to histograms, such as selecting a range of data or bars of interest in the histograms or changing the number of bins. In PHP, we adopt the analytical strategy of comparing the distributions between selected and unselected data of Attribute and Influence Explorer [40], [41]. When items of interest are selected, the color-coded stacked bar histogram for the selected data is shown in the foreground, while the original histograms for all the data are shown in grayscale in the background of PHP (Fig. 6a). In this way, the characteristics of the selected data can be examined in the context of the whole data.

Following the Visual Information-Seeking Mantra, we enable users to harmoniously use the original PCP in PHP, in situations in which the power of the pattern recognition in the original PCP can shine, i.e., exploring a smaller group of selected items with less clutter. When a user selects a bar for a group of items of interest by hovering over or clicking on it, the corresponding data items are shown as polylines as in the original PCP (Fig. 6b). This interaction can help users make a connection between histograms and PCP lines. Users can show or hide either the histograms or the polylines to avoid potential visual interference between the lines and the bars. In Fig. 6b, the outliers selected from Fig. 5 are displayed as polylines. Detailed information about the selected data can be revealed from the polylines. The polylines show that the selected outliers tend to share similarities, and that they have relatively low wFB; high WPA; and low FIP, ERA, and HR/FB.

4.2 Tools for Zooming in on Small Bars

While clicking and hovering interactions on histograms are simple and intuitive actions, issues arise when users must interact with small components in the visualization, which are hard to see and select. Each bar of a histogram in PHP consists of stacks of color-coded bars. Among the stacked bars, there can be small bars that are hard to interact with. We designed interaction techniques to overcome such problems.

4.2.1 Two-Level Zooming

One of the main issues with searching for information in histograms of stacked bars is the difficulty of noticing bins that are rendered too small owing to a skewed distribution, outliers, etc. To support observing small bins and bars, we introduce a two-level zooming interaction technique. First-level zooming, named focus+context zooming, widens the gap between axes to assign more space to an axis of interest while preserving the contexts around it in a shrunk space. In PHP, a histogram is horizontally attached to an axis, and thus occupies the space between two adjacent axes. Widening the gap between axes by dragging an axis gives more space to the histogram shown in the gap, which can reveal more details about the histogram (i.e., small bars getting bigger) (Fig. 7a). As in the focus+context technique used in Table Lens [36], users can horizontally increase the size of histograms to see more details while maintaining contextual information about all the data in the visualization.

Focus+context zooming still has limitations if the distribution of a histogram is skewed too much. In such a case, a very large space is required for the histogram to see the details of tiny bars, which sacrifices the space for other histograms. Considering this problem, we complement first-level zooming with another space-efficient, second-level zooming technique called clamp zooming. Clamp zooming is a 'within-area' zooming technique. Without changing the allocated space for a histogram, it horizontally stretches each bar inside the histogram with the same magnification, being maxed out when reaching the maximum length. When the bars reach the maximum length, they are gray colored to distinguish them from other, smaller, not-maxed-out bars, which helps users focus on the smaller bars (Fig. 7b). This clamp zooming helps users investigate the long tail of a skewed distribution. In Fig. 7b, outliers that had to be enlarged extensively in Fig. 5 (distinct blue items in the upper region of the WAR histogram) can be seen in a relatively smaller space.

When clamp zooming maxes out all bars, the original colors of the bars are restored, and the histograms transform into a heatmap visualization (Fig. 7c). This heatmap visualization is useful in extreme conditions, such as when the space allocated to a histogram is so small that the colormap from the histograms is hard to perceive. It can also be used to display a relatively high number of attributes within a limited screen size.

4.2.2 Ghost Bars for Invisibly Small Elements

When scaling histogram bars to fit the allocated space between axes, it is often inevitable that some bars cannot be

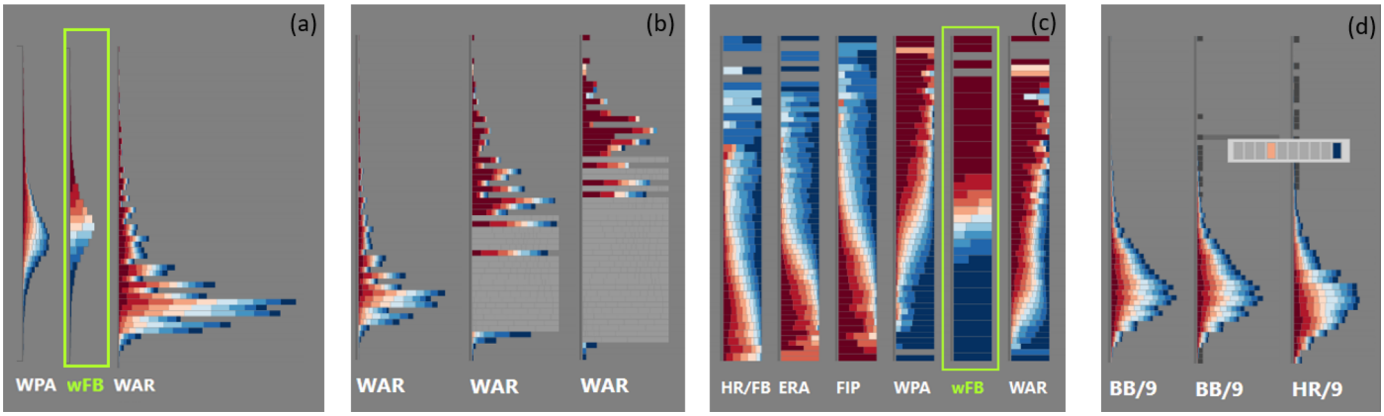


Fig. 7. Various interactions of PHP designed to deal with small components. (a) Focus+context zooming enables users to enlarge histograms of interest (the histograms of WAR are enlarged when the axis is dragged). (b) Clamp zooming helps in observing small elements in a space-efficient manner (from left to right, the histograms of WAR are increasingly clamp zoomed). (c) When clamp zooming maxes out all bars, the histograms turn into a heatmap-like visualization that can display the color distribution in a minimal space. (d) Ghost bars in the right two histograms of BB/9 and HR/9 help identify small bars that are unseen in the leftmost normal histogram of BB/9. The UI widget of pop-up color patches helps users click on tiny bars.

shown because their heights become smaller than one pixel. In Angular Histograms [11], another histogram-based PCP visualization, this problem is resolved by using an additional visual encoding idiom named Attribute Curves. Attribute Curves provides a clue that some data elements exist in a bin. But this approach takes additional space along with the original visualization. We adopt a more space-efficient approach by using a visual cue within the visualization that implies the existence of small bins, named ghost bars. The ghost bar technique shows a small but noticeable gray bar for originally invisible bins, whose length is too short to be shown on the current scale (Fig. 7d). From the gray bars, users can determine whether any bins are unseen because of their small size. The ghost bars are colored this way so that they can be distinguished from the normal histogram bars. Furthermore, they can be untoggled when they are not needed to prevent confusion.

4.2.3 *Support for Selecting Tiny Bins*

Finally, we provide a UI widget to help users select small bins in a histogram bar. When users click on the empty area right next to a (small) bar in a histogram, a widget pops up and shows a color panel, in which the colors used in the bar are shown as patches (Fig. 7d). In contrast to trying to directly click on the small bar in the histogram, which may be challenging, the color patches on the pop-up can be easily and more accurately clicked. Sometimes the pop-up widget can show colors that are invisible in the original small bar, which correspond to the invisibly small bars in the current scale.

5 Comparison With Other Visualizations

In this section, we demonstrate the efficacy and utility of PHP compared with other similar visualizations, e.g. the original PCP, AH [11], and scatterplot matrices (SPLOMs). For comparison, we used the protein tertiary structure dataset [42], which consists of 45,730 items with 10 attributes (i.e., the physicochemical properties of proteins). This dataset is part of the CASP (Critical Assessment of Techniques for Protein Structure Prediction) dataset, which contains various properties of a protein's structure.

As can be observed in Fig. 8a, the two main limitations of PCP previously discussed (L1 and L2) prevail in PCP. Because there are many items in the dataset, the overlapping of polylines is too severe in the original PCP, even though the lines are rendered translucent to mitigate the overlap. AH and PHP (Figs. 8b and 8c) both mitigate the cluttering issue using histograms. AH utilizes a vector-based approach for each bar of the histogram, with an additional attribute of direction determined by the mean angle of the polylines in the corresponding bin. Owing to this direction attribute, the histograms are tilted, likely making

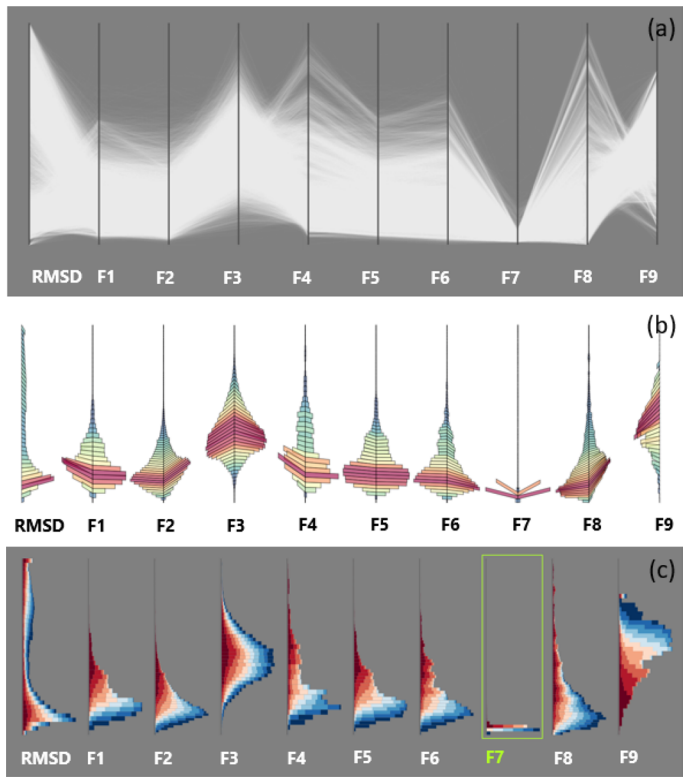


Fig. 8. CASP dataset rendered in (a) PCP, (b) AH, and (c) PHP. F7 is set as the pivot attribute in PHP. In PHP, the relationship between F7 and other attributes can be discovered by how the colors spread in the histograms, whereas in the other visualizations such discovery is hindered by the skewed distribution of F7.

it hard to derive their exact distribution [11]. To deal with this limitation, AH utilizes colors as an additional channel to show the length of each histogram's bars [11]. In contrast, PHP does not distort the distribution and utilizes color as a channel to show the relationship between the pivot and other attributes. The use of the color channel makes it possible to identify the relationship between non-adjacent attributes, in contrast to AH, in which determining the correlation depends heavily on the ordering of axes, as in other PCP-based visualizations [11]. In PHP, users can find information in a more time-efficient manner because there is no need for reordering the attributes to deduce the relationships between them.

PHP can display more attributes in a limited space than AH. PHP renders one histogram for each attribute, but AH renders two histograms for each attribute (excluding the first and last ones), taking roughly double the amount of space to render equally sized histograms. Thus, each histogram of PHP is rendered about two times larger than a histogram of AH. In larger histograms, users can observe subtle patterns or smaller bins more accurately, as well as being able to see whether the difference in histograms is tilted or not. This space efficiency also becomes an issue in SPLOMs when visualizing multiple attributes (Fig. 8d). When visualizing data with n attributes, in SPLOMs $n \times n$ scatterplots are rendered, compared with n histograms in PHP. SPLOMs become highly congested with scatterplots as the number of attributes increases, and each scatterplot becomes smaller, making it harder to observe relationships between attributes. When visualizing multiple attributes, the space efficiency of a visualization is important because it is directly related to how many attributes can be displayed in a limited screen space—i.e., the scalability of a visualization by the number of attributes. Compared to other visualizations, PHP can visualize the relationship between multiple attributes in a more space-efficient manner, benefiting users who aim to find information across multiple attributes.

In the protein dataset, the attribute F7 is radically skewed toward the bottom side of the axis. This skewness affects the performance of PCP-based visualizations. In Figs. 8a and 8b, the nearby lines and histogram bars in PCP and AH, respectively, are drastically slanted toward the lower direction. In contrast, PHP is more resilient to this skewness issue, as the shape of an attribute's distribution is independent of other attributes, unlike in PCP and AH (Fig. 8c). Moreover, correlations between the skewed F7 and other attributes can be observed by selecting F7 as the pivot attribute. In this case, the color encoding is determined by the ranking of F7, so the color distributions of all histograms show the relationships between F7 and the other attributes, not affected by the skewness of F7. In Fig. 8c, F1, F2, F4, F5, and F6 have positive correlations, F3 and F8 do not have a particularly positive or negative correlation, and F9 has a negative correlation with the skewed F7. PHP requires only selecting the skewed attribute as the pivot attribute, whereas other visualizations need further processing of the data (enlarging the visualization, filtering out outliers, logarithmic scaling, etc.) to observe more information.

Utilizing colors in PHP enables the discovery of interesting patterns. In Fig. 1, the notable red colors in the upper region of the attribute F9 (circled in red) indicate that the data items in that region do not follow the negative correlation between F2 and F9. The same information is almost impossible to obtain from AH or PCP because the attributes F2 and F9 are not adjacent to each other. While a SPLOM can show all pairwise relationships at once, it is also hard to find patterns in SPLOMs (Fig. 9) because each scatterplot is not rendered large enough owing to the number of attributes, and such interesting data items do not stand out as colors as in PHP. A focus+context technique or a simple interaction, such as selecting a scatterplot of interest to be shown as an enlarged inset, could be employed in SPLOM to mitigate this problem. In PHP, such a small group of interesting data items can be selected and displayed as polylines, as in the polylines of Fig. 1. Because only a small portion (about 1 percent of the data) is selected, the items can be displayed without cluttering. Characteristics of the selected items can be observed from the polylines, with the selected items seeming to show a negative correlation between RMSD and F1.

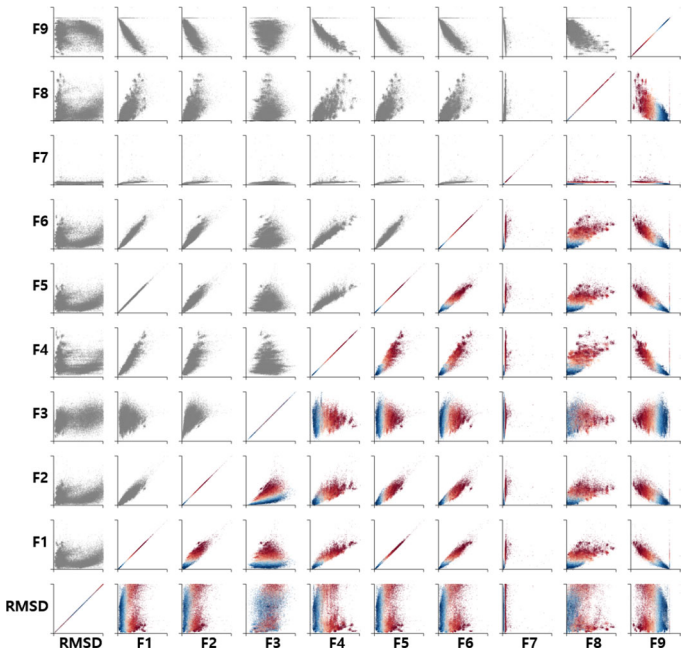


Fig. 9. CASP dataset rendered as a scatterplot matrix (SPLOM) with the colors of PHP (F7 is the pivot attribute) applied to the scatterplots in the lower-right triangle of the matrix. The colors enable additional discovery in how the items spread out in the context of F7, e.g., the items with high values for F7 (red) gather in distinct regions (right or left regions) in the scatterplots between RMSD and other attributes.

The color mapping used in PHP can also be applied to other visualizations to improve the information-seeking process. One example of this is shown in Fig. 9, in which the color mapping of PHP (F7 is set as the pivot attribute) is applied to scatterplots in the lower-right triangle of SPLOM. From how the colors spread out in individual scatterplots, additional information related to the pivot attribute can be inferred. For example, from the scatterplot between F4 and F9, it can be observed that items with high-value items of F7 are spread mostly in the upper-left region, while items with lower values of F7 are spread mostly in the lower-right region. This indicates a positive correlation between F4 and F7 and a negative correlation between F9 and F7, which can

likewise also be discovered in PHP. Also, in most of the scatterplots of RMSD and other attributes, it can be observed that items with a high value of F7 (red) are gathered in distinct regions, either the right or the left region (i.e., having high or low values of the corresponding attribute). However, the scatterplot between F3 and RMSD shows a distinct pattern, with items with a high value of F7 being gathered in the middle region of F3 and the upper and lower regions of RMSD.

6 User Study

We conducted a controlled user study to assess the performance of PHP in terms of correlation coefficient retrieval. The user study consisted of two within-subject tasks. In the first task, we compared the performance on correlation retrieval between two attributes. In the second task, distance between two attributes was added as a factor to measure how the PCP-based visualizations perform when retrieving the correlation between non-adjacent attributes. The ordering of the two tasks was fixed for all the participants: The first task was performed before the second task.

We selected three visualizations to be compared with PHP: PCP, scatterplot, and AH [11]. PCP was selected as a baseline condition to show the level of improvement of our design. While PHP is an improved version of PCP, its visual cues used to judge the correlation are different (color pattern in PHP versus line crossing in PCP). We intended to measure the effect of such a difference in visual encoding. Scatterplots were selected because they are commonly used and known to be the best method for visually analyzing the relationship between two attributes. They were used as another baseline for comparison with other techniques. We chose AH among various other improvements of PCP considering that, like PHP, it uses histograms to deal with scalability. Other approaches that use histograms [19], [40], [41] were also considered but were discarded because the visual property of the histograms of these methods does not support correlation retrieval task and requires interactions to derive any correlation between attributes.

6.1 Design

For the experiment, we recruited 36 participants from a university's online community (25 males, 11 females; aged 21-33 [mean $\pm$ SD: 25.6 $\pm$ 2.6]). Participants were screened according to two conditions: (1) participants should be familiar with the statistical terms used throughout the experiment (e.g., Pearson correlation coefficient), and (2) participants should not be colorblind. On average, the user study lasted about 60 minutes. The participants were paid about 10 dollars for their participation. A 27-inch LG monitor (27MP48HQ) was used to display the visualizations for all conditions.

Before performing the tasks, the participants received instructions for each visualization. The instructions included how the visualization is constructed from raw data, and how to interpret the patterns in the visualization to retrieve the correlation. For all visualizations used in all tasks, the interactions were disabled; only the visual encodings were utilized to retrieve the correlation coefficient.

6.1.1 First Task: Two Attributes

In the first task, users were asked to estimate the correlation coefficient (ranging from -1 to 1, with an interval of 0.1) between the two attributes displayed. In PHP, the leftmost attribute was set as the pivot attribute. We recorded the time and error rate of the responses. All responses in the experiment were self-paced, and users typed in their responses.

Two within-user factors were utilized in the experiment: (1) type of the visualization to be displayed (PCP, scatterplot, PHP and AH, with the ordering being determined by a Latin square (4 levels)) and (2) the correlation coefficient set of the data (4 levels). The set of correlation coefficients were defined to have 4 levels: $\pm$[0.9, 0.8, 0.7, 0.6] and $\pm$[0.5, 0.4, 0.3, 0.2, 0.1]. Each set will be referred to HP, HN, LP, and LN, representing high positive, high negative, low positive, and low negative coefficients, respectively.

In this task, we used randomly generated data from a normal distribution with a fixed size of 1,000 items with two attributes for each correlation coefficient set. A coefficient value was randomly chosen from a predetermined set of correlation coefficients. A pivot attribute was first generated with a normal distribution. Then, the other attribute was generated to follow the chosen coefficient value with the pivot attribute. The actual correlation coefficient of the generated data (pivot and other attributes) was slightly different from the chosen coefficient as noise was added during data generation; however, we ensured that this difference did not exceed 0.025. For each combination of visualization method (4 levels) and correlation coefficient set (4 levels), the coefficient estimation experiment was repeated 5 times. Thus, a total of $4 \times 4 \times 5 = 80$ responses was collected.

Prior to the main task, training sessions were given to the participants. The training session had the same conditions as the main task, but the response was not recorded, and the participants could check the answer and train themselves. A training session consisted of 12 responses, and users could request more training sessions if needed. On average, users performed around 2 to 3 training sessions per visualization.

6.1.2 Second Task: Multiple Attributes

In the second task, 4 attributes were displayed in one of the three visualizations (PCP, AH, and PHP). The users were asked to estimate the correlation coefficient (ranging from -1 to 1, with an interval of 0.1) between the leftmost attribute and one of the other selected attributes. In this task, we did not include scatterplots for comparison because their methodology of displaying multiple attributes (scatterplot matrices) greatly differs from other PCP-based visualizations (PCP, AH, and PHP). We recorded the time and error rate of the responses. All responses in the experiment were self-paced, and users typed in their responses.

Three within-user factors were utilized in the experiment: (1) the type of visualization (PCP, AH, and PHP) (3 levels, with the ordering determined by a Latin square), (2) the correlation set of the target attribute (4 levels [HN, LN, LP, and HP], the same as in the first task), and (3) the position of the target attribute (3 positions excluding the

leftmost; the leftmost attribute will be referred to as the pivot attribute, and each position of the target attribute will be referred to as the first, second, and third positions from the left).

In this task, we used randomly generated data from a normal distribution with a fixed size of 1,000 items with 4 attributes as in the first task. A pivot attribute was first generated with a normal distribution. Then, the other three attributes were generated according to the pivot value. When the attribute was not the target of correlation retrieval, it was generated to have a random correlation (between -1 and 1) with the pivot attribute. When the attribute was the target of retrieval, the data was generated in the same way as the target attribute in the first task. For each combination of the target data (3 levels) and correlation coefficient set (4 levels), the coefficient estimation experiment was repeated 3 times. Thus, a total of $3 \times 4 \times 3 = 36$ responses were collected per visualization, and thus a total of $36 \times 3 = 108$ responses was collected in the task.

Prior to the main task, training sessions were given to the participants. The training sessions had the same conditions as the main task, but the response was not recorded and participants could check the answer and train themselves. A training session consisted of 12 responses, and users could request more training sessions if needed. On average users performed around 1 to 2 training sessions per visualization.

6.2 Results

In both tasks, we recorded the task completion time (i.e., the time between the appearance of a visualization and the user's answer in milliseconds) and the error rate of each user's answers (i.e., the absolute difference between the user's response and the chosen coefficient).

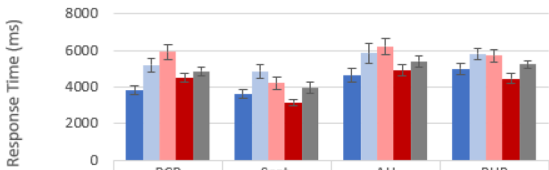
6.2.1 *First Task: Two Attributes*

The task completion time and error rate were analyzed using a 4×4 (4 visualization methods $\times$ 4 correlation coefficient sets) repeated measures ANOVA. Bonferroni's pairwise comparison was used for all post hoc tests.

Task Completion Time. Fig. 10a shows the task completion time of all correlation coefficient sets for each visualization method. There was a significant main effect by visualization type ($F_{3,35} = 13.207$, $p < .001$). Post hoc tests revealed that the task completion time of scatterplots (mean $\pm$ SD: 3,970 $\pm$ 265 ms) was significantly lower than the task completion times of all other conditions (PCP: 4,875 $\pm$ 271; AH: 5,411 $\pm$ 354; PHP: 5,255 $\pm$ 266). We also found a significant main effect by correlation coefficient set ($F_{3,35} = 35.310$, $p < .001$), with post hoc tests showing that the participants responded to the HN (4,276 $\pm$ 229) and HP (4,268 $\pm$ 200) conditions significantly faster than to the LP (5,439 $\pm$ 323) and LN (5,527 $\pm$ 300) conditions. This indicates that the participants took less time to respond to more strong patterns with positive/negative correlations.

There was also an interaction effect between visualization type and correlation coefficient set ($F_{9,35} = 3.312$, $p = .001$). For further analysis, we performed a one-way repeated measures ANOVA (4 correlation coefficient sets) for each visualization method. The result of the pairwise comparison of the four correlation sets are shown in Fig. 10b. Each

(a)



	PCP	Scat	AH	PHP
HN	3835	3624	4641	5005
LN	5215	4872	5850	5818
LP	5939	4222	6233	5715
HP	4512	3164	4918	4480
Overall [Vis]	4875	3970	5411	5255

(b)

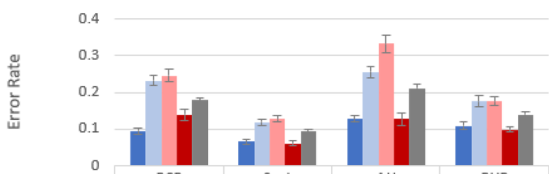
	HN-LN	HN-LP	HN-HP	LN-LP	LN-HP	LP-HP
PCP	*	*	*		*	*
Scat	*		*	*	*	*
AH	*	*				*
PHP	*	*			*	*

Fig. 10. Results regarding task completion time in the first task. (a) Results by visualization method and correlation coefficient set. Error bars indicate the standard deviation of the measured mean. (b) Significance of the difference between correlation coefficient sets for each visualization. An asterisk (*) in the table indicates that the pairwise difference is significant ($p < .05$).

visualization showed a slightly different trend. In PCP, HN outperformed all other conditions, mostly because the crossing patterns were most distinct in that condition. On the other hand, because there are no crossing patterns in PHP, such a trend did not appear for PHP.

Error Rate. Fig. 11a shows the error rate of all correlation coefficient sets for each visualization method. There was a main effect by visualization type with regard to the accuracy of the responses ($F_{3,35} = 46.618$, $p < .001$). From post hoc tests, it was found that the error rate of scatterplots (mean $\pm$ SD: 0.093 $\pm$ 0.005) was significantly lower than the error rates of all other conditions (PCP: 0.178 $\pm$ 0.007; AH: 0.211 $\pm$ 0.011; PHP: 0.140 $\pm$ 0.007). In addition, the error rate

(a)



	PCP	Scat	AH	PHP
HN	0.094	0.066	0.127	0.108
LN	0.232	0.118	0.255	0.176
LP	0.246	0.128	0.333	0.176
HP	0.139	0.06	0.127	0.098
Overall [Vis]	0.178	0.093	0.211	0.14

(b)

	HN-LN	HN-LP	HN-HP	LN-LP	LN-HP	LP-HP
PCP	*	*			*	*
Scat	*	*			*	*
AH	*	*		*	*	*
PHP	*	*			*	*

Fig. 11. Results regarding error rate in the first task. (a) Result by visualization method and correlation coefficient set. Error bars indicate the standard deviation of the measured mean. (b) Significance of the difference between correlation coefficient sets for each visualization. An asterisk (*) in the table indicates that the pairwise difference is significant ($p < .05$).

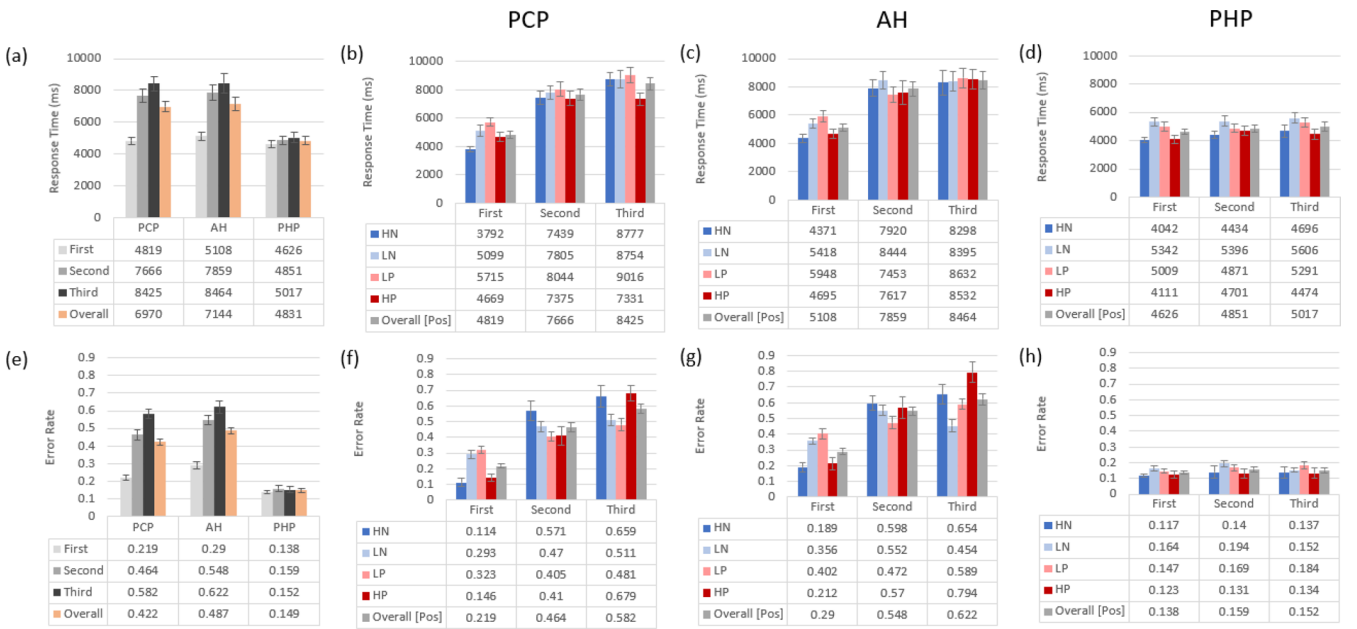


Fig. 12. Performance evaluation results of the second task. Error bars indicate the standard deviation of the measured mean. (a) Response time of the visualization by position of the target attribute. (b)-(d) Response time of each visualization by position of the target attribute and correlation coefficient set ((b) PCP, (c) AH, and (d) PHP). (e) Error rate of the visualization by position of the target attribute. (f)-(h) Error rate of each visualization by position of the target attribute and correlation coefficient set ((f) PCP, (g) AH, and (h) PHP).

of PHP was significantly lower than the error rates of PCP and AH. There was also a significant main effect by correlation coefficient set ($F_{3,35}$ = 100.049, $p < .001$). Post hoc tests indicated that the error rates in the HN (0.099 $\pm$ 0.004) and HP (0.106 $\pm$ 0.006) conditions were significantly lower than those of the LN (0.195 $\pm$ 0.007) and LP (0.221 $\pm$ 0.009) conditions. Furthermore, LN showed a significantly lower error rate than LP.

An interaction effect between visualization type and correlation coefficient set was observed ($F_{9,35}$ = 7.385, $p < .001$). For further analysis, we performed a one-way repeated measures ANOVA (4 correlation coefficient sets) for each visualization method. The result of the pairwise comparison of the four correlation sets is shown in Fig. 11b. Only in AH did LP significantly underperform LN, whereas the other visualizations did not show such notable differences.

6.2.2 Second Task: Multiple Attributes

The task completion time and error rate were analyzed using a 3 $\times$ 4 $\times$ 3 (3 visualization methods $\times$ 4 correlation coefficient sets $\times$ 3 positions of target attribute) repeated measures ANOVA. Bonferroni's pairwise comparison was used for all post hoc tests.

Task Completion Time. There was a significant main effect by visualization type ($F_{2,35}$ = 23.702, $p < .001$), with post hoc tests showing that the task completion time of PHP (mean $\pm$ SD: 4,831 $\pm$ 244 ms) was significantly lower than the task completion times of the other two methods (PCP: 6,970 $\pm$ 331; AH: 7,144 $\pm$ 419) (Fig. 12a). We also observed a main effect by correlation coefficient set ($F_{3,35}$ = 11.656, $p < .001$). The response was significantly faster in the highly correlated conditions (HN: 5,974 $\pm$ 267; HP: 5,945 $\pm$ 293) than in the other two conditions (LN: 6,675 $\pm$ 276; LP: 6,664 $\pm$ 287). Position of target attribute also showed a significant main effect ($F_{2,35}$ = 64.835, $p < .001$). The pairwise differences in task completion time between any two positions were all significant, while the response time increased as the distance between the pivot and the target attribute became bigger (first: 4,851 $\pm$ 178; second: 6,792 $\pm$ 294; third: 7,302 $\pm$ 372.

Interaction effects were also observed. Visualization type and position of target attribute showed a significant interaction effect ($F_{4,35}$ = 20.798, $p < .001$), as did correlation set and position ($F_{6,35}$ = 2.995, p = .008). For further analysis of the interaction effects, we performed a 4 $\times$ 3 (4 correlation coefficient sets $\times$ 3 positions of target attribute) repeated measures ANOVA for each visualization. As shown in Figs. 12b, 12c, 12d , in PCP and AH, correlation coefficient set and position of target attribute both showed a significant main effect in addition to the interaction effect between them. Meanwhile, in PHP, only correlation coefficient set showed a significant main effect, whereas task completion time was not affected by position of target attribute.

Error Rate. There was a significant main effect by visualization type ($F_{2,35}$ = 144.112, $p < .001$). Post hoc analysis revealed that the error rate of PHP (mean $\pm$ SD: 0.149 $\pm$ 0.011) was significantly lower than the error rates of the other two visualizations (PCP: 0.422 $\pm$ 0.018; AH: 0.487 $\pm$ 0.018) while PCP significantly outperformed AH (Fig. 12e). Position of target attribute also had a significant main effect ($F_{2,35}$ = 122.976, $p < .001$). According to the post hoc analysis, all position pairs showed a significant difference, while the error rate increased as the distance between the pivot and target attributes increased (first: 0.215 $\pm$ 0.010; second: 0.390 $\pm$ 0.014; third: 0.452 $\pm$ 0.016). No significant main effect by correlation coefficient set was observed ($F_{3,35}$ = .029, p = .993).

Multiple interaction effects were also observed. Interaction effects between visualization type and position of target

attribute ($F_{4,35}$ = 24.503, $p < .001$), between correlation coefficient set and position of target attribute ($F_{6,35}$ = 15.351, $p < .001$), and between all of the three within variables ($F_{12,35}$ = 4.827, $p < .001$). For analysis of the interaction effects, we performed a 4 × 3 (4 correlation coefficient sets × 3 positions of target attribute) repeated measures ANOVA for each visualization. As shown in Figs. 12f, 12g, 12h , in PCP and AH, we observed a significant main effect by position of target attribute and the interaction effect between it and correlation coefficient set. By contrast, in PHP, only correlation coefficient set showed a significant main effect, implying that position of the target attribute did not play a significant role in the performance regarding accuracy.

7 Discussion

The first task shows that in terms of the accuracy of the responses , PHP outperforms PCP and AH, but PHP is outperformed by scatterplots in the correlation coefficient estimation task. We suspect that the performance difference comes mainly from the innate difference in the effectiveness of the visual encodings, i.e., crossing patterns in PCP and AH, color in PHP, and position in scatterplot. Other factors could have affected the performance. While training could offset the effect, the well-known scatterplot might have advantages over the other unfamiliar visualizations. Fatigue from the first task may have negatively affected the performance in the second task, in addition to the second task being relatively more complicated than the first task. There was mostly no tradeoff between response time and error rate (faster performance does not increase the error rate). One exception to this was a faster response time in scatterplots under positive correlation conditions compared with negative conditions. In scatterplots, the response time of HP was faster than that of HN, and LP was faster than LN. But there was no significant difference in the performance between the two pairs. Although we have no empirical evidence, we suspect that the difference in response time is caused by participants' being more familiar with scatterplots with positive correlations. We think a more thorough analysis of this issue can be a potentially interesting future topic.

Results of the second task show that the positioning of attributes in PHP does not influence the performance of the correlation retrieval task, unlike other conditions in which the performance severely decreases when the target and pivot attributes are not adjacent. The empirical results imply that PHP mitigates one of the two main innate limitations of PCP we previously stressed—i.e., the difficulty in estimating relationships between non-adjacent axes. AH, which that also utilizes histograms to deal with scalability did not outperform PCP and performed worse than PHP. Crossing and cluttering of bars remain in AH, even though histograms are used to deal with the scalability issue of PCP, implying that AH does not fully overcome the first limitation of PCP we mentioned —i.e., the cluttering of polylines caused by multiple crossings. Compared with AH, PHP utilizes a totally different visual channel, i.e., color, to deal with the cluttering problem, and thus it is free from the cluttering by crossing line patterns. We expect that when the number of items further increases, AH will perform better than PCP because of the effectiveness of histograms in dealing with scalability.

When analyzing multidimensional data, it is a common approach to start by inspecting each attribute individually (1D) and then continue by examining the relationships between two or more attributes in order to obtain insights in higher dimensions [38]. PHP supports this data exploration process. Each histogram in PHP shows the distribution of one dimension, which is hard to see in PCP or SPLOMs. In PHP, users can select a pivot attribute and observe all the data from the perspective of that attribute using the attribute's colormap. After studying the 1D histograms, users can explore the relationships between two or more attributes using the color mapping applied to all other histograms. The implicit connection via colormapping reveals relationships between the pivot attribute and other attributes. Users can move on to select another attribute as a pivot, group and reorder similar attributes for higher dimensional analysis, or zoom in further to inspect a small group of items of interest in an attribute.

Throughout the paper, we fixed various parameters that could affect the performance of the visualization—e.g., the set of colors of the color scheme, the number of colors used in the color scheme, and the number of bins of the histograms. Measuring the effects of changing these parameters could be an interesting future research direction. Throughout the paper and user study, we used a blue-red color scheme for PHP. Studying how a different color scheme might affect task performance in correlation estimation could also be interesting. In addition, the number of distinct colors was fixed at a relatively small value (10) throughout the paper. The number of discriminable hues mapped onto small, separated regions is known to be moderate, i.e., fewer than 10. While using relatively few colors can still help users grasp the overall trend in the data, it could potentially oversimplify the information in the data, hindering the discovery of more diverse and precise patterns of colors in the visualization. However, such a detailed exploration is possible with the original PCP visual encoding, i.e., polyline representation. Investigating the effect of the number of colors in terms of perceiving a data distribution is an appealing future research topic. Increasing the number of colors could reveal different structures in the data, but it could become harder to discern different colors, and individual bars might become too small to interact with.

The number of bins affects the shape of a histogram, which is related to how the colormap is rendered. We expect that changing the number of bins should not greatly influence users' task performance in estimating correlation, as they examine the overall color distribution. However, since the shape of the colormap changes, it could influence some tasks, such as finding outliers or a group of similar items. Since categorical attributes do not carry any ordering information, our rank-based approach cannot be directly applied to categorical attributes. It would be interesting to study how to harmoniously combine the ranking channel and the identity channel in using color mappings for multidimensional data analysis. Also, while we proposed various approaches to dealing with skewed histograms, such as using colors based on ranking or utilizing two-level zooming interactions, they all require some level of user input.

Combining the proposed approaches with other analytic methods (e.g., log transformation) that deals with the skewedness of a distribution would be an interesting direction. Finally, PHP can be integrated with other related visualizations similarly to how PCP has been integrated with other visualizations (e.g., scatterplots).

8 Conclusion

We introduced PHP, a novel visualization technique designed to overcome the innate limitations of PCP. PHP utilizes color-coded, stacked-bar histograms to show the relationships between attributes without the issue of cluttering and regardless of the distance between the attributes. With PHP, users can discover interesting items using colored stacked-bar histograms: Similar colors gathered in a small region suggest clusters of data items that follow a certain trend, and salient colors from the overall color distribution suggest outliers. In addition, PHP provides interactions to help users investigate the details of histograms in a limited screen space: two-level zooming (i.e., focus+context zooming and clamp zooming), ghost bars, and a UI widget of the color panel. Following the Visual Information-Seeking Mantra, polylines are used to display the details of focused data, while color-coded histograms provide the overview. We demonstrated how PHP can be used on a real-world dataset in a use case. We also tested the performance of PHP in correlation coefficient estimation tasks. The results showed that PHP correlation estimates were consistent regardless of the distance between attributes.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) under Grant Nos. NRF-2019R1A2C2089062 and NRF-2019R1A2C1088900). The research facilities for this study was provided by ICT at Seoul National University. The authors would like to thank all the participants of the experiments.

References

[1] A. O. Artero, M. C. F. de Oliveira, and H. Levkowitz, "Uncovering clusters in crowded parallel coordinates visualizations," in *Proc. IEEE Symp. Inf. Vis.*, 2004, pp. 81–88.

[2] J. Bok, B. Kim, and J. Seo, "Scaling up parallel coordinates plots with color-coded stacked histograms," *IEEE VIS*, 2018. [Online]. Available: http://hcil.snu.ac.kr/system/publications/pdfs/000/000/122/original/final2.pdf

[3] J. H. T. Claessen and J. J. van Wijk, "Flexible linked axes for multivariate data visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 12, pp. 2310–2316, Dec. 2011.

[4] ColorBrewer. 2018, [Online]. Available: http://colorbrewer2.com

[5] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott, "ImAxes: Immersive axes as embodied affordances for interactive multivariate data visualisation," in *Proc. 30th Annu. ACM Symp. User Interface Softw. Technol.*, 2017, pp. 71–83. [Online]. Available: http://doi.acm.org/10.1145/3126594.3126613

[6] A. Dasgupta and R. Kosara, "Pargnostics: Screen-space metrics for parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 6, pp. 1017–1026, Nov./Dec. 2010.

[7] G. Ellis and A. Dix, "Enabling automatic clutter reduction in parallel coordinate plots," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 717–724, Sep./Oct. 2006.

[8] E. Fanea, S. Carpendale, and T. Isenberg, "An interactive 3D integration of parallel coordinates and star glyphs," in *Proc. IEEE Symp. Inf. Vis.*, 2005, pp. 149–156.

[9] Fangraphs baseball. 2018, [Online]. Available: https://fangraphs.com

[10] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner, "Hierarchical parallel coordinates for exploration of large datasets," in *Proc. Vis.*, 1999, pp. 43–508.

[11] Z. Geng, Z. Peng, R. S. Laramee, J. C. Roberts, and R. Walker, "Angular Histograms: Frequency-based visualizations for large, high dimensional data," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 12, pp. 2572–2580, Dec. 2011.

[12] H. Guo, H. Xiao, and X. Yuan, "Scalable multivariate volume visualization and analysis based on dimension projection and parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 18, no. 9, pp. 1397–1410, Sep. 2012.

[13] H. Hauser, F. Ledermann, and H. Doleisch, "Angular brushing of extended parallel coordinates," in *Proc. IEEE Symp. Inf. Vis.*, 2002, pp. 127–130.

[14] J. Heinrich and D. Weiskopf, "Continuous parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 15, no. 6, pp. 1531–1538, Nov./Dec. 2009.

[15] J. Heinrich, Y. Luo, A. E. Kirkpatrick, and D. Weiskopf, "Evaluation of a bundling technique for parallel coordinates," in *Proc. Int. Conf. Comput. Graphics Theory Appl. (IVAPP-2012)*, pp. 594–602, doi: 10.5220/0003821205940602.

[16] J. Heinrich and D. Weiskopf, "State of the art of parallel coordinates," in *Proc. Eurographics*, 2013, pp. 95–116.

[17] A. Inselberg and B. Dimsdale, "Parallel coordinates for visualizing multi-dimensional geometry," in *Computer Graphics 1987*. Berlin, Germany: Springer, 1987, pp. 25–44.

[18] T. Itoh, A. Kumar, K. Klein, and J. Kim, "High-dimensional data visualization by interactive construction of low-dimensional parallel coordinate plots," *J. Vis. Lang. Comput.*, vol. 43, pp. 1–13, 2017.

[19] H. Janetzko, M. Stein, D. Sacha, and T. Schreck, "Enhancing parallel coordinates: Statistical visualizations for analyzing soccer data," *Electron. Imag.*, vol. 2016, no. 1, pp. 1–8, 2016.

[20] J. Johansson, M. Cooper, and M. Jern, "3-dimensional display for clustered multi-relational parallel coordinates," in *Proc. 9th Int. Conf. Inf. Vis.*, 2005, pp. 188–193.

[21] J. Johansson and C. Forsell, "Evaluation of parallel coordinates: Overview, categorization and guidelines for future research," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 1, pp. 579–588, Jan. 2016.

[22] J. Johansson, P. Ljung, M. Jern, and M. Cooper, "Revealing structure in visualizations of dense 2D and 3D parallel coordinates," *Inf. Vis.*, vol. 5, no. 2, pp. 125–136, Jun. 2006.

[23] D. A. Keim, M. C. Hao, U. Dayal, and M. Lyons, "Value-cell bar charts for visualizing large transaction data sets," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 4, pp. 822–833, Jul./Aug. 2007.

[24] H. Kobayashi, T. Furukawa, and K. Misue, "Parallel box: Visually comparable representation for multivariate data analysis," in *Proc. 18th Int. Conf. Inf. Vis.*, 2014, pp. 183–188.

[25] R. Kosara, F. Bendix, and H. Hauser, "Parallel sets: Interactive exploration and visual analysis of categorical data," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 4, pp. 558–568, Jul./Aug. 2006.

[26] T. V. Long, "A new metric on parallel coordinates and its application for high-dimensional data visualization," in *Proc. Int. Conf. Adv. Technol. Commun.*, 2015, pp. 297–301.

[27] L. F. Lu, M. L. Huang, and T. Huang, "A new axes re-ordering method in parallel coordinates visualization," in *Proc. 11th Int. Conf. Mach. Learn. Appl.*, 2012, vol. 2, pp. 252–257.

[28] K. T. McDonnell and K. Mueller, "Illustrative parallel coordinates," *Comput. Graph. Forum*, vol. 27, no. 3, pp. 1031–1038, 2008.

[29] H. Nguyen and P. Rosen, "DSPCP: A data scalable approach for identifying relationships in parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 3, pp. 1301–1315, Mar. 2018.

[30] K. Nohno, H. Wu, K. Watanabe, S. Takahashi, and I. Fujishiro, "Spectral-based contractible parallel coordinates," in *Proc. 18th Int. Conf. Inf. Vis.*, 2014, pp. 7–12.

[31] M. Novotny and H. Hauser, "Outlier-preserving focus+context visualization in parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 893–900, Sep./Oct. 2006.

[32] G. Palmas, M. Bachynskyi, A. Oulasvirta, H. P. Seidel, and T. Weinkauf, "An edge-bundling layout for interactive parallel coordinates," in *Proc. IEEE Pacific Vis. Symp.*, 2014, pp. 57–64.

[33] W. Peng, M. O. Ward, and E. A. Rundensteiner, "Clutter reduction in multi-dimensional data visualization using dimension reordering," in *Proc. IEEE Symp. Inf. Vis.*, 2004, pp. 89–96.

[34] H. Qu *et al.*, "Visual analysis of the air pollution problem in Hong Kong," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 6, pp. 1408–1415, Nov./Dec. 2007.

[35] R. G. Raidou, M. Eisemann, M. Breeuwer, E. Eisemann, and A. Vilanova, "Orientation-enhanced parallel coordinate plots," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 1, pp. 589–598, Jan. 2016.
[36] R. Rao and S. K. Card, "The table lens: Merging graphical and symbolic representations in an interactive focus + context visualization for tabular information," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 1994, pp. 318–322.
[37] R. C. Roberts, R. S. Laramee, G. A. Smith, P. Brookes, and T. D'Cruze, "Smart brushing for parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 3, pp. 1575–1590, Mar. 2019.
[38] J. Seo and B. Shneiderman, "A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections," in *Proc. IEEE Symp. Inf. Vis.*, 2004, pp. 65–72.
[39] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proc. IEEE Symp. Vis. Lang.*, 1996, pp. 336–343.
[40] L. Tweedie, B. Spence, H. Dawkes, and H. Su, "The influence explorer," in *Proc. Conf. Companion Hum. Factors Comput. Syst.*, 1995, pp. 129–130.
[41] L. Tweedie, B. Spence, D. Williams, and R. Bhogal, "The attribute explorer," in *Proc. Conf. Companion Hum. Factors Comput. Syst.*, 1994, pp. 435–436.
[42] UCI Machine Learning Repository. 2018. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure
[43] J. Walker, Z. Geng, M. Jones, and R. S. Laramee, "Visualization of large, time-dependent, abstract data with integrated spherical and parallel coordinates," in *Proc. Eurographics Assoc.*, 2012, pp. 43–47.
[44] R. Walker, P. A. Legg, S. Pop, Z. Geng, R. S. Laramee, and J. C. Roberts, "Force-directed parallel coordinates," in *Proc. 17th Int. Conf. Inf. Vis.*, 2013, pp. 36–44.
[45] J. Wang, X. Liu, H. Shen, and G. Lin, "Multi-resolution climate ensemble parameter analysis with nested parallel coordinates plots," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 81–90, Jan. 2017.
[46] 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 *Proc. IEEE Symp. Inf. Vis.*, 2003, pp. 105–112.
[47] X. Yuan, P. Guo, H. Xiao, H. Zhou, and H. Qu, "Scattering points in parallel coordinates," *IEEE Trans. Vis. Comput. Graphics*, vol. 15, no. 6, pp. 1001–1008, Nov./Dec. 2009.
[48] Z. Zhang, K. T. McDonnell, and K. Mueller, "A network-based interface for the exploration of high-dimensional data spaces," in *Proc. IEEE Pacific Vis. Symp.*, 2012, pp. 17–24.

Jinwook Bok received the BS degree in computer science and engineering from Seoul National University, Seoul, Korea, in 2017. He is currently working toward the PhD degree with the Department of Computer Science and Engineering, Seoul National University, Seoul, Korea. His research interests include HCI and multidimensional data visualization.

Bohyoung Kim received the BS and MS degrees in computer science and the PhD degree in computer science and engineering from Seoul National University, Seoul, Korea, in 1995, 1997, and 2001, respectively. She is currently an assistant professor with the Department of Biomedical Engineering, Hankuk University of Foreign Studies, Korea. Her research interests include computer graphics, volume visualization, medical imaging, and information visualization.

Jinwook Seo received the PhD degree in computer science from the University of Maryland at College Park, in 2005. He is currently a professor with the Department of Computer Science and Engineering, Seoul National University, where he is also the director of the Human-Computer Interaction Laboratory. His research interests include HCI, information visualization, and biomedical informatics.