Revisiting Bertin Matrices: New Interactions for Crafting Tabular Visualizations

Charles Perin, Pierre Dragicevic, and Jean-Daniel Fekete, *Member, IEEE*

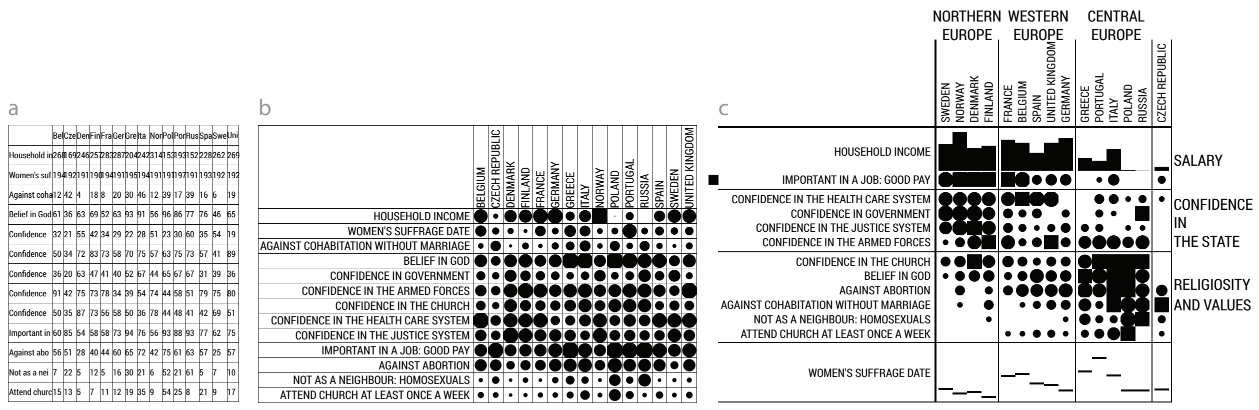

Fig. 1. A spreadsheet formatted and reordered with BERTIFIER: a) the original numerical table; b) the corresponding tabular visualization; c) the final result, reordered, formatted and annotated. The final result is ready to be exported and inserted as a figure.

Abstract— We present BERTIFIER, a web app for rapidly creating tabular visualizations from spreadsheets. BERTIFIER draws from Jacques Bertin's matrix analysis method, whose goal was to "simplify without destroying" by encoding cell values visually and grouping similar rows and columns. Although there were several attempts to bring this method to computers, no implementation exists today that is both exhaustive and accessible to a large audience. BERTIFIER remains faithful to Bertin's method while leveraging the power of today's interactive computers. Tables are formatted and manipulated through *crossets*, a new interaction technique for rapidly applying operations on rows and columns. We also introduce *visual reordering*, a semi-interactive reordering approach that lets users apply and tune automatic reordering algorithms in a WYSIWYG manner. Sessions with eight users from different backgrounds suggest that BERTIFIER has the potential to bring Bertin's method to a wider audience of both technical and non-technical users, and empower them with data analysis and communication tools that were so far only accessible to a handful of specialists.

Index Terms—Visualization, Interaction, Tabular Data, Bertin, Crossing, Crossets

1 INTRODUCTION

Most Infovis researchers know the French cartographer Jacques Bertin from his 1967 monograph *"La Sémiologie Graphique"* [8]. Less known is his later work from 1975, *"La Graphique et le traitement graphique de l'information"* [10, 11]. This book details a method for processing and communicating tabular data visually that was meant to be effective, generic and accessible to any scientist and researcher [42]. It was based on two simple ideas: *i)* encoding table cells visually, and *ii)* grouping similar rows and columns to reveal patterns. Bertin devised and refined his method after years of work with data analysts such as geographers, agricultural economists, ethnologists, and historians [42]. However, his method required a physical matrix that was only available at his lab (Figure 2) and involved tedious manipulations—often weeks of work.

Now with the widespread availability of computers, anyone can generate tabular visualizations automatically [59], and Bertin's physical matrix is little more than an intriguing historical anecdote. However, the results are rarely satisfactory without a good amount of user interaction [33, 49]. Bertin himself realized the formidable potential of interactive computers [42] and he tried to adapt his method to computers [14]. Later, some Infovis researchers tried to resurrect Bertin's matrices [46, 49]. However, none of these implementations has been widely adopted, perhaps because they only featured rudimentary interactions, were incomplete, and were not accessible to a wide audience.

Bertin's idea to make tables visual and fully reorderable remains largely underexploited. Yet we believe that such an approach can facilitate data analysis not only by scientists, but also by a much wider audience due to the strong similarity between common tables and tabular visualizations. Many people today store and manipulate data using spreadsheets, and may be interested in getting new insights on their data, effectively communicating it to others, or quickly making sense of tables they receive. Although spreadsheet tools offer basic support for tabular visualization and reordering, through conditional formatting and sorting, this support remains surprisingly poor. We address this lack of proper tools through the following contributions:

- Requirements for tabular visualization tools based on Bertin's work.
- An extensive review of existing systems for tabular visualization.
- BERTIFIER (`www.bertifier.com`), a web-based tabular visualization authoring system compatible with online spreadsheets.
- *Crossets*, a novel interaction technique for creating, manipulating and fine-tuning tabular visualizations.
- Algorithm adaptations to support user-driven matrix ordering.
- A study suggesting that users from different backgrounds can use BERTIFIER to help them understand their own tabular data.

The tool we introduce remains faithful to the spirit of Bertin's original work while fully leveraging the power of modern graphical computers. By releasing it to the public, we hope that more scientists will consider Bertin's approach as a way to explore, analyze and interpret their data, as well as to communicate their findings by visual means. We also hope that BERTIFIER will bring Bertin's methods to a wider audience, both for educational and for pragmatic purposes.

- *Charles Perin is with INRIA, Université Paris-Sud and CNRS-LIMSI. E-mail: charles.perin@inria.fr.*
- *Pierre Dragicevic is with INRIA. E-mail: pierre.dragicevic@inria.fr.*
- *Jean-Daniel Fekete is with INRIA. E-mail: jean-daniel.fekete@inria.fr.*

Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014. Date of publication 11 Aug. 2014; date of current version 9 Nov. 2014. For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org. Digital Object Identifier 10.1109/TVCG.2014.2346279

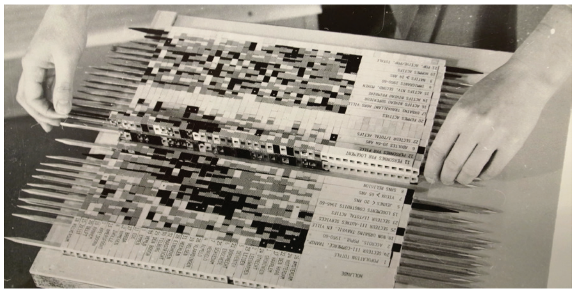

Fig. 2. A physical matrix from Bertin (Serge Bonin, Archives Nationales).

2 Background

We first explain our focus on Jacques Bertin's method. We then present a short history of tabular visualizations with a focus on this method, and finally review existing implementations and the methodological issues involved in porting this method to graphical computers.

2.1 Why Focus on Bertin's Method?

As we discuss later in this Section, Jacques Bertin is not the only one who employed and advocated tabular visualizations and matrix reordering. Moreover, his recommendations are more informed by his intuitions as a cartographer than by formal empirical evidence. We however focus on Bertin for five major reasons:

- Jacques Bertin has accumulated a unique experience and know-how by working on real data analysis problems as a consultant for about 100 practitioners over the course of his career [25, 42],
- Bertin's method is the only detailed tabular visualization analysis method that encompasses the whole analytic process, from formulating research questions to communicating insights.
- While Bertin's *Sémiologie* has had a great influence on the domain of Infovis, the full details of his matrix method are only known by a few. One of the goals of this work is to demystify Bertin's matrices and provide a tool with pedagogical and historical value.
- Sticking closely to Bertin's original work is a first step to understanding its limitations and identifying valuable improvements.
- Bertin introduced a unique visual style for presenting tables that people may want to imitate simply for its compelling value.

2.2 Before Bertin

Most of Bertin's ideas were not new. The idea of encoding cell values visually was already around in the late 19th century [58]. We refer to this technique as *tabular visualizations*, but various other names have been employed, such as *heat maps* and *shaded matrix displays* [59]. The idea of reordering rows and columns to reveal patterns also dates back to the late 19th century, when the English Egyptologist Flinders Petrie reordered strips of paper to reconstruct the chronology of excavated graves [39, 59]. Although only rows were ordered and no visual encoding was used, tabular visualizations that were reordered on both rows and columns started to appear shortly afterwards [59].

Reordering tabular visualizations on both rows and columns manually is a tedious process, and specialized devices have been designed to assist in this task. In the 1950s, the Israeli mathematician and psychologist Louis Guttman built one called the "scalogram board" for analyzing questionnaire responses [53]. The device consisted of two separate wooden boards, one for reordering rows and the other one for columns, and data items were represented by ball bearings that could be transferred between the two boards [26]. In the 1960s French sociologist Robert Pagès built a magnetic version called "le permutateur" [42]. Little information remains today about these two devices.

2.3 Bertin's Method

Despite this previous work, Bertin was the only one to provide a general and detailed method that was meant to be applicable across a range of domains [42]. He also carefully considered visual design, building on his earlier work on map and chart design: *La Sémiologie Graphique* [8].

Reorderable matrices are part of a more general method of "graphic information processing" [9, 55] that Bertin started to develop shortly after he published *La Sémiologie Graphique*. Later the method was detailed in a book on its own [10, 11] and summarized in new editions of *La Sémiologie Graphique* [13, 42]. This method stems from his realization that, while maps and charts are traditionally drawn once for all, "mobile images" can be freely manipulated to reveal patterns [10, 42]. Thus he recognized the importance of interactivity three decades before information visualization [16, 19]. Bertin identified and physically implemented five types of interactive visualizations [9]: the *family of curves* (line charts on transparent paper sheets that can be superimposed), the *image file* (stacked cards with a visual index), the *collection of maps* (rearrangeable small multiples), the *collection of ordered tables* (rearrangeable paper tables), and finally the *reorderable matrix*.

The general method consists in three stages [10, 11]: **S1** – *Convert research questions into a table*, i.e., **S1a**: frame high-level research questions, then **S1b**: compile data into a numerical table in a way that is relevant to the research questions. Then **S2** – *Construct and process the image*, i.e., **S2a**: turn the table into an image by choosing the appropriate encodings and data conditioning (i.e., data transformations), then **S2b**: manipulate this image in order to simplify without destroying and reveal hidden patterns. Finally, **S3** – *Interpret, decide and communicate answers* consists in interpreting the resulting image according to externally available information and annotating it for publication.

These three stages were carried out with the help of one of Bertin's physical interfaces, depending on the type of dataset considered. While all interfaces complement each other, the reorderable matrix is, according to Bertin, the most general method of which others are only subcases. Thus, the matrix method is the focus of our work.

Bertin designed three reorderable physical matrices he called Dominos, each with a different size and visual encoding. Figure 2 shows Domino 2, a matrix of intermediary size. A rod mechanism allowed unlocking either rows or columns for reordering. The initial stage, S1, was carried out on paper. In stage S2a, values were converted into discrete steps on a paper table (7 to 11 different steps depending on the Domino version), then the physical matrix was assembled by choosing among a collection of physical cells that encode different ranges of values. In stage S2b, the matrix was reordered. Finally, in stage S3, meaningful groups were identified and named. The result was then photographied or photocopied, and the final image was used as a figure in the scientific publication, enriched with a legend and caption.

The reordering stage S2b could take weeks because of the manipulations involved and because no systematic procedure was known. Bertin's reordering method heavily relied on visual judgment. When asked how exactly he reordered his tables, he referred to the *"painter's eye"*[1]. He however tried to give heuristics and illustrated them with several examples. He recommended the following [10]: *i)* choose a row with a particular aspect (*e. g.,* high values) and move it to an extremity of the matrix. *ii)* Move similar rows close to this reference row, and opposite rows to the bottom. This will create two opposite groups, with a third group in the middle. *iii)* Do the same for columns. *iv)* Iterate.

From Bertin's original matrix method we derive requirements that a computer implementation should provide:

R1 allow the creation of a table from raw data
R2 perform data conditioning: scale, clamp range, discretize (step), and inverse rows/columns values so they become comparable
R3 select an encoding for cell values
R4 present the table visually
R5 reorder the rows/columns to group similar items together and move apart dissimilar ones
R6 group rows/columns that form meaningful chunks
R7 annotate the matrix (name groups)
R8 finalize the results for communication / publication.

2.4 Automatic vs. Manual Approaches

Many of the previous requirements could in principle be automated, which raises the difficult question of the level of interaction needed to perform an effective analysis and to produce an effective tabular visualization. Bertin's method generally involves many different types of tasks. Some of them have a strong data analysis component (e.g., stage S2b), while others clearly require a substantial amount of decision

[1] J. Bertin, personal communication, 2008.

making. The stage S2a, for example, involves choosing appropriate visual encodings, a task for which computer automation can provide only limited assistance. The same is true for stages such S1a and S3.

What remains is the question as to whether the stage S2b (requirement R5) should be fully automated or whether humans need to be involved. For data analysis in general, the question of automation is a long-standing debate [21]. While fully automated analysis yields tremendous benefits such as speed, scientific objectivity and reproducibility [31], full automation also has a number of issues: *i)* it ignores the analyst's knowledge that cannot be formalized [57]; *ii)* it yields only minimal exposure to the data and therefore does not encourage deep understanding [22]; *iii)* it may produce errors that can be way more costly than the effects of human subjectivity [58]; and *iv)* it often produces visualizations that are suboptimal for communication. For example, cluster analysis throw out lots of data and can produce results that are controversial, subject to misinterpretation, or meaningless [32, 39, 51].

Many researchers have stressed the importance of human judgment in data analysis. Bertin was one of them: "the best graphic operations are those carried out by the decision-maker himself" [11]. But Bertin was also aware that matrix reordering could benefit a lot from automation [8], at no cost: automatic reordering is a non-destructive analysis approach, just as manual reordering. However, Bertin commented on the results of three automatic reordering algorithms and pointed out that none of them was satisfactory [13]. He concluded that automatic reordering can save time but needs to be interlaced with manual tweaking. Semi-automatic approaches to matrix reordering have been argued for since then [33, 51]. Devising semi-automatic methods however requires a good understanding of automatic methods.

2.5 Matrix Reordering Methods

Automatic matrix reordering methods have been extensively studied, although mostly in isolation by three domains: statistics, linear algebra, and graph theory. Liiv [39] provides an extensive review of seriation (another term for matrix reordering). According to him, "seriation is an exploratory combinatorial data analysis technique to reorder objects into a sequence along a one-dimensional continuum so that it best reveals regularity and patterning among the whole series." While clustering methods try to create cohesive groups according to some measure of cohesion, seriation is only concerned by finding an order, leaving operations like grouping to the interpretation of the user.

Automatic seriation of a table $T_{N,M}$ of N rows and M columns consists of finding an order for the rows and columns that optimizes and objective function $F(T)$. A naive method would try to evaluate F for all the possible permutations of rows and columns, requiring $\frac{N!\,M!}{2}$ operations, which is not practical even for small tables.

Furthermore, even the objective function F is not well understood. According to several articles, F should bring together rows and columns that are similar. This characteristic has been formalized by Robinson [44] by using the similarity matrix (*SM*) of rows (resp. columns), considered as vectors, computed using a similarity measure. According to Robinson, "the highest values in the matrix should be along the diagonal and monotonically decrease when moving away from the diagonal" [44]. There are several automatic methods to compute an order to try to achieve a "Robinsonian" *SM* from a given *SM*, but for most real tables, no permutation will lead to a truly Robinsonian *SM*. Approximations of the Robinsonian can always be computed, but there is no clear metric to decide which is best.

Still, even without a clear characterization of "good" orders, there is a rich literature on automatic methods for seriation; Liiv cites 171 articles in his review [39]. One approach is to model a numeric table as a weighted bipartite graph, the edges being the table cells connecting rows to columns, with the cell values as weights. Graph-based ordering methods can then be applied. Díaz et al. [18] present an overview of linear ordering methods for graph vertices that try to optimize 9 graph measures (*e. g.,* bandwidth or min-cut), referring to 261 articles.

Following Bertin, we want to let users interactively build a subjectively satisfying order by iteratively enhancing many partial solutions. Bertin argues for ordering rows and columns independently, which is incompatible with methods ordering both axes at the same time such as biclustering or Siirtola's approach [49]. We thus opted for using "optimal leaf ordering" [4] (*OLO*). This method starts from a hierarchical clustering of rows (resp. columns) and finds an order that is consistent with the dendrogram and optimal in the traveling salesman's sense: the sum of distances between consecutive items is minimal. This method is computationally expensive ($O(n^3)$ in time) but provides very good results with only two parameters: the distance metric and the linkage type used for the hierarchical clustering. Furthermore, it is flexible enough to accommodate enhancements for interactive use.

2.6 Modern Tabular Systems

We now turn to existing systems. Before reviewing systems inspired by Bertin's reorderable matrices, we give a brief overview of other tools for creating and manipulating tabular visualizations.

Spreadsheet tools (*e. g.,* Excel, Google Spreadsheet) are probably the most commonly used. Their primary purpose is to execute functions on cells or subsets of cells to compute derived values. They can easily be used to create tables from raw data (R1) and condition rows and columns (R2). As for R3 and R4, it is possible to use conditional formatting to encode cell values, typically through color — although Bertin strongly argues against the use of colors to encode numerical and ordinal values, since colors have no natural order [10]. Spreadsheets also include various charting tools, but these are not tabular visualizations. Concerning reordering (R5), most spreadsheet tools support manual reordering by letting users drag row or column headers, and also support sorting, a useful but limited form of automatic reordering.

While spreadsheet tools are mainly designed to create and manipulate tables, other programs are specifically meant to visualize them. Tableau [54] supports a large number of visual representations for tabular data, including tabular visualizations (R4) for which it provides several cell encodings such as color or bar charts (R3). It also supports data conditioning (R2), manual row and column reordering (R5), annotations (R7) and image export (R8). Thus, despite not being (at least explicitly) inspired from Bertin, Tableau supports a range of features recommended by Bertin. However, Tableau does not provide automatic reordering algorithms besides sorting (R5), and—like most Bertin implementations as we will see—it relies on a complex WIMP-style user interface. Several other charting and visualization systems exist that usually support image export (R8) but are often weak in interactive reordering, grouping, and annotation (R5–R7).

Several Infovis research systems implement or extend tabular visualizations. Table Lens [43] is a focus+context system for exploring large tables, where numerical rows and columns can be interactively compressed into tabular visualizations. Users can change visual encodings (R3) and sort columns (R5). FOCUS [52] is another focus+context tabular exploration system that lets users collapse identical adjacent values. Tableplot [36] is a system for visualizing factor-analytic data that extends tabular visualizations with elaborate cell encodings, including multidimensional encodings (R3), and supports automatic reordering (R5). None of these tools has grouping, annotation, or export capabilities (R6–R8), making them inappropriate for communication.

2.7 Computer adaptations of Bertin's method

The number of past attempts to adapt Bertin's method to computers is surprisingly large. While previous work only cite a few of these attempts [15, 33, 34, 46, 49], we conducted an extensive search over the course of three months and found a total of 16 tools, reported in Figure 3. These tools were built within different communities who were likely not aware of each other's work. Several tools were not available online and were obtained by contacting the authors or their past collaborators. The three tools that could not be tested at the time of this article's submission were evaluated based on publications, videos, and communication with their authors.

Figure 3 presents systems in rows and their characteristics in columns. The *features* category refines our requirements R2-R8 into testable software functionalities based on two extra sources: Bertin's later specifications for adapting his method to computers [12], and our own observations of images created by him and his collaborators. The *interaction* category assesses the user interface. Although Bertin emphasized interaction, he did not discuss how the user interface should be

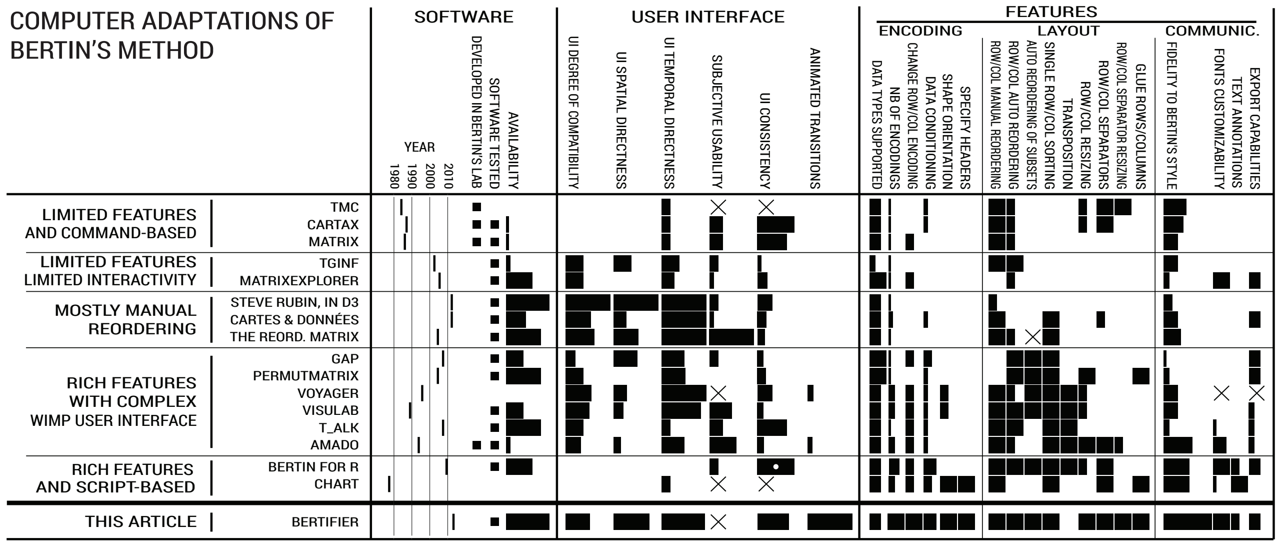

Fig. 3. Computer adaptations of Bertin's matrix method with their availability, interactivity, and features. Crosses indicates unavailable data.

designed. The tools listed employ a variety of interaction styles, ranging from fully conversational (e.g., command-based) to fully graphical (e.g., direct manipulation). Direct manipulation interfaces are generally believed to be easier to learn and to support rapid, incremental and reversible actions [48]. Thus the first three columns score interactivity by how "direct" the interactions were, based on criteria from the instrumental interaction model [6]. More details on the coding schemes are available at `www.aviz.fr/bertifier`. Based on these scores we classify the systems into five categories:

1. Limited features and command-based: TMC [37], Cartax [27], and Matrix [23] are early systems developed in Bertin's laboratory and used in French public schools. They support only a few visual encodings (R3) and layout features (R4), and limited data conditioning (R2). They support both manual and automatic reordering (R5), but some of the developers commented that automatic reordering was acting as a black box and recommended using pen, paper, pen and tape as a first step to get used to the method [28]. From the interaction standpoint, the three systems are based on keyboard input and are clearly obsolete.

2. Limited features and interactivity: these systems support R3, R4 and R5 but are lacking most features and employ indirect interaction methods (*i. e.* menus, toolbars and dialog boxes). TGINF [41] provides few features and only accepts binary values. MatrixExplorer [33] has a few more features but at the expense of a more complex user interface.

3. Mostly manual reordering: a d3 implementation [45], Cartes et Données [3], and The Reorderable Matrix [49, 51] are relatively recent implementations that support R3, R4 and R5, and let users rearrange the tabular visualization by direct manipulation. Their user interface scores high in overall "directness" but only because the system has very few features besides manual reordering.

4. Rich features with complex WIMP user interface: GAP [60], PermutMatrix [15], Voyager [17, 46], VisuLab [34], T_alK [29], and AMADO [14] generally support most of the requirements: R1, R2 to some extent, R3, R4, R5, and for PermutMatrix [15], R6. In contrast with previous systems, they provide table layout features and support the reordering of subsets, hence a better support for R4 and R5. They are still lacking in encoding flexibility (R3) and do not support text annotations (R7). They employ indirect interactions like the systems in category 2, but are generally high in temporal directness due to the tools having an immediate effect, and some use of direct manipulation.

5. Rich features and script-based user interface: Two tools use a script-based language for creating and customizing tabular visualizations. CHART [7] is the first computer implementation of Bertin's method. The paper describes a long list of features but we could not test the software. Bertin-R is a plug-in to the R statistical package [47] that provides powerful features and capitalizes on R's extensive support for data importation (R1) and transformation (R2). Both systems are close to supporting all features (R1-R7) but they employ a user interface that is indirect and dedicated to experts. The author of Bertin-R—also author of Voyager [46] in Category 4—recognized that such interfaces are incompatible with Bertin's vision of "mobile images"[2].

To summarize, there were a remarkably large number of attempts to bring Bertin's method to computers over more than 30 years, but none of the implementations really caught on. We may have a working system today had developers built on each other's accumulated experience, but most of them were not aware of all other attempts. Our survey tries to fill this gap. Overall, some early attempts—especially CHART [7] in 1977—were rich in terms of functionality, but poor in terms of interactivity. On the opposite side of the spectrum, a few recent implementations—Like Rubin's implementation in d3 [45]—provide consistent, direct and easily-discoverable interactions, but at the cost of only providing a few features. Like in many other application domains, there is a marked trade-off between the level of functionality on one side, and the level of interactivity and discoverability on the other side. But the two sides are not entirely irreconcilable. The last row shows the features of BERTIFIER, the system we propose, and suggests that a Bertin implementation can be high overall both in functionality and in interactivity. We discuss this system in the next sections.

3 BERTIFIER WALKTHROUGH

We illustrate BERTIFIER based on a fictional usage scenario. Suppose Clara is preparing a newspaper article on European values. She compiles a Google Spreadsheet where each column is a country, and each row contains information about this country. She took most of her data from a survey on European values [20] : for example, the percentage of people who believe in God, or who find that a good salary is important for a job. Although she carefully selected the data to keep the table small (15×16), it is impossible for her to see any pattern.

Clara prefers not to use traditional charts (e.g., bar graphs or scatterplots) because she wants to see the entire table. So she decides to "bertify" it. She just publishes her spreadsheet to the web, opens the BERTIFIER web page and pastes the table URL. She sees the numerical table (Figure 1a) and can transform it in a number of ways.

When Clara moves her mouse outside the table, a gray frame appears that contains formatting tools made of *crossets*: small crossing-based interactive components (Figure 4). All tools on the left apply to entire rows, while tools on the top apply to columns. Tools on the right apply to the space between rows, and tools on the bottom apply to the space between columns. When she moves her mouse on top of any tool icon, a tooltip appears to describe its function.

Tools are organized in groups. To mark the first row as a header row, Clara first expands the "Misc" group and clicks on the "H" icon

[2]G. Sawitzki, personal communication, February 11, 2014.

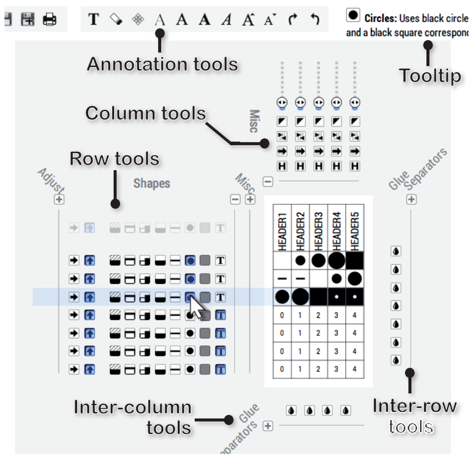

Fig. 4. The BERTIFIER interface, with some tool groups collapsed.

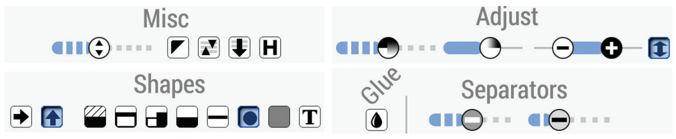

Fig. 5. All crossets for manipulating rows and columns.

adjacent to the first row (top right of Figure 5). She does the same with the left column. Then, to turn numeric values into shapes, she expands the "Shapes" group (second row of Figure 5), clicks on the circle icon next to the first row of the table to turn its values into black circles and squares. To propagate this encoding, she clicks the same icon on the next row, and drags down to the last row. This instantly turns all the table cells into circles and squares (Figure 1b).

Now Clara wants a more compact table. Each row can be resized independently using a slider placed next to it (leftmost crosset on the top of Figure 5): Moving the slider to the right increases row height while moving it to the left decreases it. She sets all rows to their minimum size by clicking on the topmost slider, dragging down until the last row (at which point all sliders are controlled concurrently), then dragging left. She does the same for columns.

Now she wants to tidy up the table. She drags vertically over all black arrow icons (top right of Figure 5), which reorders rows by visual similarity. Indicators that have a similar profile are now close together. She does the same for columns, which moves similar countries next to each other. To reduce clutter, she removes the grid by setting all white separators and black separators to their minimum value (bottom of Figure 5). Now, she can already see several country groups.

Clara continues to format the table to exhibit more patterns and convey a clearer message. This includes inverting the values of a row (importance of a good salary) to better show its correlation with another (household income), and emphasizing two rows of specific type (women's suffrage year and household income) by changing their visual encoding, increasing their height, and dragging them aside. Finally, Clara adds separators and annotations to emphasize groups. Her table is ready to be used as a figure (Figure 1c). She downloads it as a vector graphics file and inserts it in her word processor.

This figure will help Clara explain that there are roughly two opposite groups of countries: Northern Europe, with early women's suffrage, confidence in state organizations, happy with their high salaries, not very religious, and open-minded about homosexuality and abortion. On the opposite side, Central Europe countries trust the army and the Church but not their state organizations and justice system. Intriguing cases include the Russians who are very religious but do not go to church. Countries from Western Europe stand in-between. Czech Republic is an outlier: low income and low confidence in state organizations, but not religious. Clara will also comment on general trends. For example, salary tends to be more important in countries with low income. These countries also tend to be more religious and less open-minded. The women's suffrage came also rather late.

4 Design Details and Rationale

BERTIFIER is an open-source application implemented using *Javascript* and *d3*, and runs in a modern web browser. Here we describe and justify the key aspects of its design.

4.1 General Design Principles

BERTIFIER's design philosophy is to remain as faithful as possible to Bertin's original method—in particular, we made sure we could reproduce most of Bertin's examples—while exploiting the opportunities of modern technology. Bertin's recommendations are influenced by the technology available at the time: very few things were possible or even thinkable, and HCI has improved dramatically since then.

As seen previously, Bertin's method is a step-by-step approach. Since the initial stage S1—formulating research questions and compiling a data table—is already well-supported by spreadsheet tools, we focus on supporting stages S2 and S3. In principle, data conditioning and visual encodings should be chosen first, then the matrix is built, then reordered, and then annotated [10]. This sequential approach may have been driven by Bertin's devices, that did not allow otherwise. We take a different angle by making all these operations accessible at any time and from a single integrated view, without enforcing any order. Thus BERTIFIER is compatible with Bertin's step-wise approach, while supporting more flexible data exploration and visual authoring processes by minimizing premature user commitment [30].

Another design principle is the accessibility to a wide audience. This includes the choice of a stand-alone web application, support for online spreadsheet import, the design of a short video tutorial, the use of informative tooltips, and a careful choice of terminology. For example, a term like "visual encodings" is unlikely to be understood by people without Infovis background, and we therefore preferred the term "shapes" (Figure 5). Animated transitions are also provided for all operations to help users understand their effects.

Two major design problems remained to be addressed: designing tools that are simple yet powerful enough to support a wide range of operations on matrices and subsets of matrices, and providing a sensible support for human-assisted reordering. We addressed the first issue by introducing *crossets*, and the second issue by introducing the principle of *automatic visual reordering*, presented in the next sections.

4.2 Crossets

Based on recent work in HCI, we chose to design tools that minimize the indirections typically found in traditional interfaces [6] and come as close as possible to direct manipulation. While manual reordering lends itself well to direct manipulation, it is not the case for more abstract operations such as choosing an encoding, or conditioning the values of a row. We addressed this problem by introducing *crossets*. Crossets are inspired from work on painting-based and crossing-based interfaces [1, 2, 5], i.e., interfaces that let users invoke commands by crossing (or painting over) widgets instead of clicking them.

A crosset is a widget placed next to a row, a column, or next to an intersection (Figure 4). Crossets are grouped in "toolbars" that can be collapsed and expanded, and there can be as many crossets as needed. This allows users to apply a variety of operations on arbitrary rows or columns (e.g., for specifying headers or visual encodings), and on the intervening space between rows and columns (e.g., for specifying separators). The customization of individual rows and columns provides lots of flexibility. For example, Bertin produced tabular visualizations with different encodings for different rows [55]. Although these were made with another device than the reorderable matrix (the image file), being able to encode different types of dimensions in different ways can facilitate both data analysis and communication.

Crossets also allow users to apply operations to multiple rows or columns in a single gesture. For commands such as provided by the "Shapes" toolbar (Figure 5), this is done by applying the command to the first row, then crossing subsequent rows. The result is immediately shown but only confirmed at mouse release, allowing users to freely adjust their selection. Slider widgets (Figure 5) follow the same principle, except they also support the adjustment of multiple rows or columns at once, by dragging in the orthogonal direction (illustrated in

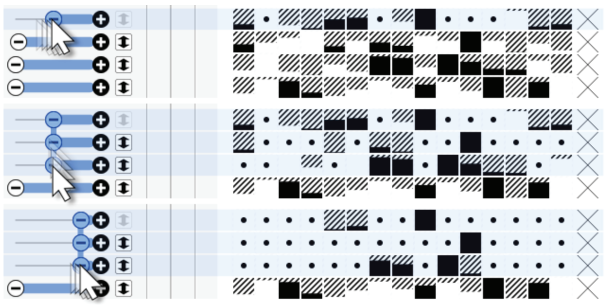
Fig. 6. Using a crosset to adjust the range of three rows at a time.

Figure 6). Compared to standard spreadsheet interactions for resizing rows or columns, the resizing crosset does not require the selection to be specified in advance. Since this operation impacts the layout of crossets, the tool only shows a partial preview. The same is true for separator adjustment tools and the automatic reordering tool.

By supporting crossing-based interactions, BERTIFIER makes it possible to quickly change arbitrary groups of adjacent rows and columns, which is useful in many cases, including for automatically reordering rows or columns within identified groups. Crossing all rows or columns allows users to apply operations to the entire table in a quasi-instantaneous manner, provided the table fits the screen. All these interactions remain consistent across all tools.

Crossets extend previous work on crossing-based interfaces in several ways [1, 2, 5]. One problem with such interfaces is that they require steering, a slow motor task [1]. Crossets do not have this problem since the command is selected on mouse press, after which the user is allowed to freely deviate from a straight trajectory. As far as we know, crossing gestures have never been applied to manipulate multiple sliders at once, and have never been used to interact with tables.

4.3 Human-Assisted Reordering

Human-Assisted reordering is supported both through manual reordering interactions and through automatic visual reordering.

4.3.1 Manual Reordering Interactions

As in previous implementations [14, 45, 46, 49], we support column and row reordering by drag and drop. This is the first level of integration between automatic and manual reordering, and allows to tweak the results of automatic reordering [33]. Following previous findings that reordering rows and columns concurrently can be confusing to users [49], we lock the reordering on the row or column based on the initial dragging direction. To help users understand changes, we perform animated transitions during the dragging operation.

Following previous recommendations [49–51], we also support dragging on sets of rows and columns. We provide a tool that lets users "glue" several rows or columns together (Figure 5 bottom right). Since we distinguish between groups used for concurrent manipulation and visual groups (i.e., separators), we avoided the ambiguous term "group". Our automatic reordering algorithm preserves rows and columns glued together by the user, thus providing a second level of integration.

4.3.2 Automatic Visual Reordering

The principle of *automatic visual reodering* is that rows and columns are reordered not according to the underlying data, but according to their visual similarity. We believe this principle is easier to understand for users not familiar with data analysis. Visual reordering is only a user interface metaphor that does not have to accurately capture what the system is doing, but which is meant to elicit a simple and "good enough" mental model of the system to allow for easy tuning.

The implementation of this metaphor is fairly simple and relies on two basic principles: *i)* the reordering algorithm should take as input the data *after* it has been conditioned and normalized (e.g., betwewn 0 and 1), and *ii)* the visual encodings used should ensure that visual differences are roughly proportional to numerical differences. From this it follows that the automatic reordering algorithm will behave *as if* it were operating visually. Next we discuss to what extent the condition *ii)* is fulfilled by the visual encodings implemented in BERTIFIER.

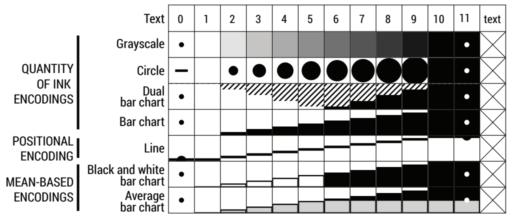

Fig. 7. Encodings in BERTIFIER. The range of each row is set to $[1, 10]$. 0 and 11 are beyond the range. Crosses encode N/A values.

4.3.3 Visual Encodings

BERTIFIER implements eight types of visual encodings (Figure 7). For the sake of generality, we consider text as being one type of encoding. Automatic reordering is however disabled on text to reinforce the visual reordering metaphor. All encodings except grayscale have been either explicitly mentioned in Bertin's book [10] or were used by him. The software can be easily extended with other types of encodings.

The black and white bar charts and average bar charts are particular in that they encode summary statistics in addition to value: values lower than the mean are shown in white or gray, while values above the mean are shown in black. The line encoding is also different in that it uses a positional encoding. These three encodings have been only used occasionally and are not implemented in any of the physical matrices.

The remaining encodings are more common and all roughly follow the rule that the quantity of ink—the average pixel darkness—is proportional to the normalized data value. We enforced this rule to make the encodings consistent with the visual reordering metaphor. For example, the dual bar chart encodes the values from white to black, with a fully hatched cell corresponding to the mean value of the row. In this case, we use a 50% hatching to respect the rule of proportionality. Thus for these four encodings, computing a numerical difference between cells is equivalent to computing the difference in their average shading. The rule of proportionality is challenging to implement for the "circle" encoding, since the circle is clipped and progressively turns into a square. We chose to replicate Bertin's original encoding although previous work described an optimal scale for symbol size discrimination [38]. Deriving the correct circle radius is a geometrical problem without any analytic solution, but we found the following good approximation:

$$r = \begin{cases} D = 2\sqrt{v/\pi} & \text{if } v \leq \frac{\pi}{4} \\ \frac{1}{2}(t + t^6)(\sqrt{2} - 1) + 1 \quad \text{with } t = \frac{D-1}{2/\pi - 1} & \text{if } v > \frac{\pi}{4} \end{cases}$$

with v being the cell value and r the (unclipped) circle radius.

4.3.4 Interactive Data Conditioning

Although users can pre-process their data in their spreadsheet as part of step $S1$, BERTIFIER provides tools for performing further data conditioning of rows on-the-fly (the "Adjust" toolbar in Figure 5). Slider crossets are provided for *i)* adjusting the data range and clipping the values accordingly, *ii)* reducing the maximum value of the data (which has the effect of making all cells look brighter or disappear), *iii)* turning values into discrete steps. In addition, users can *iv)* invert row or columns values. *i)*, *iii)* and *iv)* have been recommended by Bertin [12] and *i)*, *ii)* and *iii)* are implemented in CHART [7].

These operations change the matrix visually in a similar way as photo retouching tools. Since the automatic reordering is performed on post-processed values, these controls provide a way of fine-tuning the reordering algorithm without explicitly tuning any of its internal parameters. Specifically, rows that are made brighter will be given a lower weight by the reordering algorithm. Rows that are made entirely white will be ignored. A specific case of this is making all rows white except one, which enables a regular sorting operation.

Sometimes it is useful to provide custom ranges (*e. g.,* for applying a uniform range to all rows). To achieve this, users can specify the range of each row in the initial spreadsheet, using special header names. A crosset in BERTIFIER allows to enable or disable this custom range.

4.4 Bertifier's Reordering Algorithm

As explained in Section 2.5, we use the *OLO* [4] as basis for visual reordering. This algorithm takes a list of vectors of values as input, either rows or columns, with a distance metric—we use Euclidean or Manhattan—and returns an optimal order by minimizing the sum of distances between consecutive vectors. We provide a richer API on top of the *OLO* to take into account interactions. The API allows to specify a first and/or last vector as limits that will remain at the end(s) and the remaining vectors will be optimally ordered. Finally, one or several ranges of vectors can be protected against reordering (rows or columns that are glued). Our implementation uses 4 pre-processing steps before the standard *OLO*: 1) limits management, 2) exception managements, 3) equivalent vectors management, 4) standard reordering.

Limits are implemented by changing the distance to the first and/or last vectors. These vectors should be far away from the other vectors so that the hierarchical clustering will add them to the last cluster(s). We compute the maximum distance m in the matrix and add it to the distance of every vector to the starting vector. If there is an ending vector, we add $2 \times m$ to its distance to every other vector. Exceptions are implemented similarly: when a range $[a, b]$ is specified as an exception, all the vectors between them are removed from the list, only the first and last vectors are kept, and a distance of 0 is set between them in the distance matrix. The *OLO* then always glues them together and the indices of the vectors in-between are re-inserted afterwards.

The clustering algorithms can generate artifacts when applied with many identical vectors so we remove all but one of them before applying the algorithm and re-insert them afterwards after their representative. Note that the reordering algorithm can sometimes invert the order of vectors. When limits are specified, the pre-processing step 1) makes sure the final order keeps the limits where they were, but protected ranges can still be inverted by the algorithm. We believe this is not an issue for interactive reordering since range constraints are still honored and each group remains the same. Finally, standard sorting is just a particular case of the reordering algorithm when the list of rows contain exactly one column, and vice versa.

4.5 Support for Requirements and Limitations

Now we discuss to what extent BERTIFIER supports the requirements identified in the Background Section:

R1. Table creation is delegated to spreadsheet software, whose data can then be imported in BERTIFIER. However, modifying the table requires restarting the whole authoring process. Also, the table needs to be correctly prepared, as BERTIFIER has no transposition feature.

R2. Basic conditioning (scaling, clamping range, steps and inversion) is supported by crossets, while more elaborate processing needs to be done in R1. Adding more transformations (e.g., log) to BERTIFIER would allow to try more options with immediate visual feedback.

R3. Encoding is supported by eight different crossets, plus two crossets to change their orientation (Figure 5). Encodings support ordinal data provided it has been numerically coded, while there is no encoding yet for qualitative data—which can be split into binary dimensions [10].

R4. The tabular visualization is always visible and updated on-the-fly.

R5. Manual reordering is supported through direct manipulation and the glue crossets. Automatic reordering is supported through crossets, and can be tuned using the glue and data conditioning crossets. Implementing visual reordering based on actual visual similarity between cells (instead of ink) may yield new possibilities for algorithm tuning.

R6. Crossets can be used to create a variety of separators. However, these are fixed and not updated when reordering rows or columns.

R7. A separate toolbar allows users to add text annotations, but they are not structured and remain fixed when the table layout changes.

R8. The results can be exported in SVG for further tweaking with external authoring software. This step is often crucial, for example for resizing labels or adding legends as in Figures 3 and 8.

5 ARE SPREADSHEETS COMPATIBLE WITH BERTIFIER?

To better understand spreadsheets generated "in the wild" we collected personal and professional spreadsheets by asking people from research (*C1*, 123 spreadsheets), and administration & education (*C2*, 128 spreadsheets) to send us their spreadsheets, for a total of 14 people. We extracted 16 characteristics per table. We use the notation $Q_{\mathbf{Ci}} = \{Q1, Q2, Q3\}$ to report the quartiles of counts (25%, 50%, and 75%) of spreadsheets belonging to category *Ci*.

Spreadsheets are generally compatible with Bertin's method. Following Bertin, a table must clearly distinguish between entries and dimensions. 92.5% of our spreadsheets have explicit entries and dimensions for *C1*, and 93.6% for *C2*. Moreover, spreadsheets must contain numerical values and present a data table. 87% of *C1* spreadsheets meet this condition, and 85.2% for *C2*. The reasons for not being conform are various and include: containing multiple tables (44% of non conform spreadsheets for *C1* and 16% for *C2*); being a calendar layout (13% and 37%); being a simple list of people (0% and 26%); and being a drawing support or diagram layout (12% and 0%).

Spreadsheets are small. For numbers of entries, $Q_{\mathbf{C1}} = \{13, 21, 37\}$: 25% of spreadsheets in *C1* are shorter than 13 columns, 50% are shorter than 21 columns, and 75% are shorter than 37 columns; $Q_{\mathbf{C2}} = \{10, 22, 35\}$. For dimensions, $Q_{\mathbf{C1}} = \{6, 9, 14\}$, $Q_{\mathbf{C2}} = \{9, 11, 13\}$.

Spreadsheets mostly contain quantitative values. 64.4% of the dimensions are quantitative for *C1* and 92% for *C2*; then qualitative (14.9% and 5.3%), ordinal (10.2% and 2.5%), and text (10.5% and 0.2%).

Spreadsheets often contain missing data. 56.1% of the spreadsheets for *C1* and 20.2% of spreadsheets for *C2* contain N/As.

Spreadsheets sometimes contain color coding. Conditional formatting is used for 12% of *C1* and 35% of *C2* spreadsheets. It indicates that users are willing to make sense of their data by using visual cues.

Overall, 86% of the spreadsheets we analyzed are compatible with Bertin's method requirements. The typical spreadsheet contains a unique table with 10–37 entries and 6–14 dimensions, far below the limits of BERTIFIER; both entries and dimensions have 1–2 headers and mostly contain quantitative values.

6 QUALITATIVE USER STUDY

Figure 8 presents the results of the qualitative study in a bertified table. The rest of this Section refers to this Figure.

6.1 Procedure

We recruited participants willing to visualize a data table they were interested in. They initially sent us their table of interest. We limited table sizes to 40×15 (which represents more than 75% of personal tables according to our spreadsheets analysis) to avoid too long sessions. We selected eight participants based on their background, and favored diversity of datasets and tasks. We cleaned up some of the tables with the participant to ensure compatibility with BERTIFIER.

The session consisted of five phases: i) the participant read short instructions about the session; ii) the facilitator asked the participant to explain why the dataset has been chosen, what was already known about it, and what the hypotheses were; iii) the participant watched a six minutes video tutorial explaining how to use BERTIFIER and covering all features (the same scenario as in the walk-through Section 3); iv) the participant switched to BERTIFIER in the web browser. We encouraged participants to think aloud and they could ask questions to the facilitator, although they were not guided when using the tool. We took notes on everything the participant said, as well as questions, difficulties, behavior and insights. When participants decided that the work was done using the tool, they could open the final result in a vector graphics editor to finalize it. Finally, participants filled a questionnaire. The session ended with a short (10–15 minutes) semi-structured interview. The average session time was 75 minutes (min 60, max 130), including 29–52 minutes using BERTIFIER. Summary data is reported in Figure 8.

6.2 Participants

Participants could be roughly categorized into two groups of equal size: *Infovis/HCI* and *Other*. In the *Infovis/HCI* category, participants tended to be experts in infovis, HCI, computer science, data analysis, scientific research and spreadsheet software.

The Post-Doc (***PD***, HCI) wanted to analyze participants' answers to two online user study questionnaires. She did not analyze the data and expected to see differences between the two techniques she wanted to compare, in order to strengthen her quantitative results.

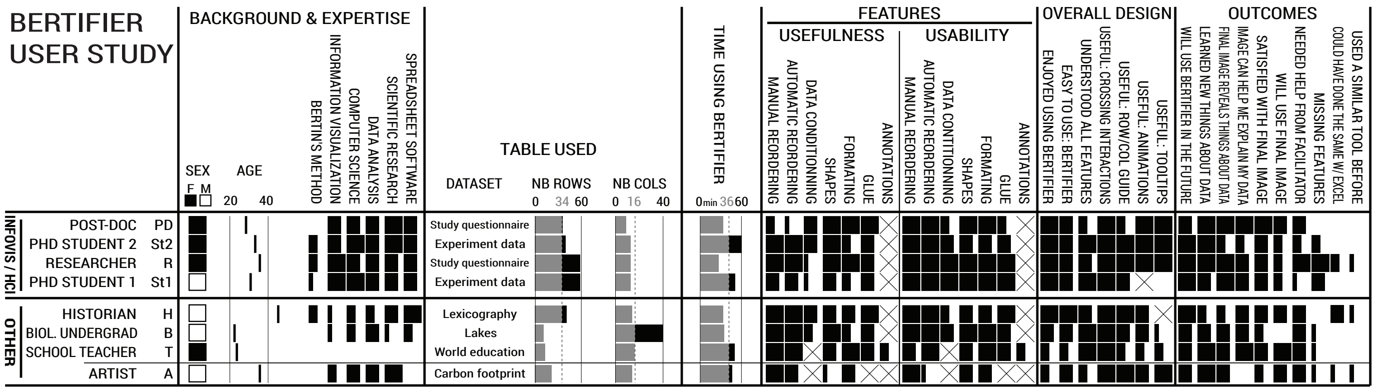

Fig. 8. Visual summary of participant demographics, datasets, time on task and questionnaire responses. Crosses indicate unused features.

The researcher (***R***, Infovis) also wanted to analyze participants answers to a user study questionnaire, for a project on glyph design [24]. She had hypotheses, but did not look at the results. She expected to discover interesting patterns that are worth statistical analysis.

The two PhD students (***St1***, Infovis, ***St2***, Infovis/Design) worked independently on the same dataset. Their table contained both qualitative (questionnaire answers) and quantitative (*e. g.,* time, counts) results from an exploratory user study on constructive visualization [35]. With 57 rows and 13 columns, their table was the largest of our evaluation. They wanted to find out if their measures were correlated with participant's demographics. They also wanted to group participants according to the way they performed the tasks. Overall they were seeking an informative overview of heterogeneous data that is difficult to understand, and wanted to report the trends in their article.

In the *Other* category, participants had little or no expertise in Infovis, and varying levels of expertise in other domains.

The Historian (***H***) wanted to visualize lexicographical data from a project he already published about the vocabulary used in judiciary texts. He knew the data very well and already analyzed it using statistical methods. His task was not exploratory but confirmatory: he wanted to check that the visualization tool will highlight the same results.

The Biologist undergrad (***B***) was interested in rivers and lakes; he provided a dataset with 40 lakes and 11 characteristics. He wanted to *"have an overview of the location where there are no large lakes and understand why."* His hypotheses were that there were only a few large lakes in the southern hemisphere, and in highly elevated countries.

The School teacher (***T***) provided a table with country areas as columns, and education indicators as rows. She presented this table in one of her classes and realized some pupils did not know how to read a table and had difficulties making sense of such heterogeneous numbers.

The Artist (***A***) provided a table with countries and environmental indicators. He had been active in an environmental association and was interested in communicating facts and convincing people. He also wanted to *"have a more global view of [his] own carbon footprint"* by making sense of those numbers.

6.3 Results

Observations, questionnaire responses (Figure 8) and interviews indicate that participants with and without Infovis background found BERTIFIER useful. All were enthusiastic during the session and rated BERTIFIER as easy to use and enjoyable. They found it useful both for exploration and communication, and for personal and professional use.

Manual reordering was the first step for two participants (*A*, *T*) who reordered rows and columns based on their knowledge and the data semantics. Sometimes, it also helped to fix the result of *Automatic reordering* if they judged it unconvincing (*St2*). Some participants used this feature to move away rows or columns that were not interesting (*St1*, *St2*), and *St2* took advantage of the animated transitions by dragging slowly a row over other ones to identify correlations visually. All participants gave the maximum score for *Manual reordering* usability and *Other* participants found it more useful than *Infovis/HCI* ones (R5).

Automatic reordering was one the most used features and generated the most insights (R5). Just after reordering, participants discovered surprising facts (*e. g., "I was not expecting to see Portugal in first position according to forest area"*, *A*), generated new questions (*e. g., "I think that if I had the age of the lakes as a dimension, it would be very interesting to see how it correlates"*, *B*), and provided meaningful arrangements of the data (*e. g., "I understand something that was hidden in my previous factorial analysis"*, *H*). All participants but *A* reordered their table locally several times by dragging over a subset of adjacent crossets. They did it when they were satisfied with some groups, but not with the rest of the table. It yielded new findings (*R*, *T*).

Data conditioning was mostly used for two reasons: first, to reorder the table according to a subset of rows only. Participants performed this task by reducing to its minimum the *Strength* of the rows they wanted to ignore, using crossing gestures (*PD*, *R*, *St1*, *B*). Second, *St1* and *St2* de-emphasized rows that they wanted to ignore or that they did not want to show in their final image by setting their strength value to 0. Several participants used the *Range* slider to change the scale of a row when an outlier compressed the other values (*St1*, *St2*, *H*, *B*). *Data conditioning* is one of the only feature lowly ranked by several participants. It suggests that R2 is reserved to advanced users. For example, *PD* and *St2* commented that it was very useful to negate several questions of her questionnaire because they were not all congruent and *B* negated a dimension to check if it was inversely correlated to another one; on the other hand, *T* did not like the negative function at all since to use it, she needed the header of the row to be negated too.

Shapes: All participants except *A* tried several encodings to get different views on their data and rated this feature as very useful (R3). Participants encoded rows differently according to their type: *PD* used grayscale to encode frequencies and barcharts elsewhere; *H* chose the black & white barchart because he was interested in perceiving average values; *B* encoded latitude and longitude using lines to compare their position in space, and circles to convey measures congruent with shape such as area and volume. Two participants (*T*, *A*) commented that squinting at the table/blurring their eyes revealed areas and were surprised to realize that they were reordering the table without paying attention to the semantic of the data, but only the shapes.

Formatting: All participants resized at least once the rows and columns to get an uniform layout using a dragging gesture. *St2* also used this feature to highlight interesting rows by increasing their height. All participants used separators during their session, both for exploration and for communication purposes. More expected use of separators included adding white spaces for readability (7/8 participants) and separating groups for communication purposes (6/8 participants).

Glue was used at least once by all participants except *A* (R6). The first gluing often happened after the participant asked how to move several rows or columns at once in order to save time (*St1*, *H*, *T*). It helped move apart final groups and focus on the rest of the table. *PD*, *B* and *H* found that the interaction with the *Glue* crosset was not obvious but that the animated feedback on the matrix was helpful. Overall, participants found this feature very useful but not straightforward.

Annotations were used by *T* only (R7). All other participants either forgot about this feature (*St2*, *H*) or said they preferred to annotate their image in a vector graphic editor (*PD*, *R*, *St1*, *B*).

6.4 Overall Benefits of Bertifier

All *Infovis/HCI* participants seemed convinced by the overall design of BERTIFIER, and reported finding it very easy to use and enjoyable. Only *H* (and to a lower degree *R*) thought he could have accomplished the same using a standard spreadsheet. Participants in the *Other* group needed more help from the facilitator compared to *Infovis/HCI*. However, no one asked for extra features. The two strongest takeaways from the experiment are about crossets and automatic reordering:

Crossets obtained the maximum score from all participants, who almost never clicked individual crossets but applied crossing gestures to series of crossets. They commented it was extremely useful (*R*, *PD*, *St1*, *A*) and powerful (*H*). No participant had difficulties understanding and reproducing the gesture they saw in the video tutorial. They were even sometimes positively surprised to see that the gesture worked on slider widgets (*R*) and for sliders that were previously set to different values (*A*). Surprisingly, all participants found the *Formatting* crossets extremely easy to use but not the most useful. We expected the converse to be true, as formatting seems important to us, but our tools have a low degree of *compatibility*, as the gesture direction to change the size of rows, columns and separators is orthogonal to its effect.

Human-assisted reordering coupled with grouping and conditioning was considered an easy and efficient way of exploring the data. *R* said that BERTIFIER was the only tool she knows with reordering capabilities : *"Usually I do the reordering by hand, and do not see a matrix visualization of it."*. *St2* said that automatic reordering allows her to do *"something [she] always wanted to do with [her] table"*. She already used spreadsheet tools for this purpose but found them tedious.

All participants learnt new things about their data, except *H* who gave a score below 4/5 for learning new things, because he already analyzed his data thoroughly. They all anticipate several situations where they would use BERTIFIER (min score is 4/5). They consider using it for exploring personal (*St1*, *B*) and heterogeneous (*PD*, *St2*) data, and for communicating findings (*St1*, *A*) and illustrating articles (*R*, *PD*, *B*, *St1*) with the ability to present all their data (*R*, *St2*). *St1* and *St2* later used BERTIFIER to craft an overview of their data and included it in a paper submission [35]. *H*, *T* and *A* also see a pedagogical value in the tool. *H* wants to integrate BERTIFIER into the Master's class he teaches as soon as it will be available on-line, both for pedagogical purposes and to provide exploratory analysis of tabular data. *T* believes that the image she created could be useful for teaching lessons when presenting a numerical table of small size is already challenging.

7 Discussion and Future Work

BERTIFIER is the first attempt to remaining faithful to Bertin's original matrix method, while leveraging the power of computers, controlled with recent methods from HCI. This design choice gives us a fresh perspective on Bertin's approach. Our user study suggests that Bertin's method is still valid and useful today, but deserves to be further refined.

Non-Sequential Analysis. Bertin was constrained by the technology available at his time, but modern computers now allow to relax some of these constraints. In particular, the method does not necessarily need to follow a linear order: conditioning (R2), choosing the encoding (R3), presenting the table, grouping and reordering (R5) can be done in any order and as many times as necessary to explore the data, generate new hypotheses, and converge towards the best visual image. BERTIFIER can still be improved in this direction, as formatting (i.e., glued groups, separators, and annotations) is not well-integrated with reordering.

Semantic Reordering. Our study stresses that Bertin's famous "Town dataset" is only an idealized example meant to illustrate the benefits of data-driven reordering. Many datasets do not yield clean clusters, and users sometimes prefer to preserve the natural hierarchy and/or ordering of dimensions in order to facilitate reading, independently from the data patterns. Many datasets are also heterogeneous, leading users to use a mix of data-driven reordering (automatic and manual) and semantic reordering (manual only). Overall, BERTIFIER's essential value is to support the presentation of tabular data in a compact and legible fashion, and automatic reordering is only one tool among others.

Visual Encodings. Not all of Bertin's original visual encodings may be optimal and easy to interpret. Many of our participants expressed doubts about the effectiveness of certain encodings, although personal preferences varied a lot. Some participants also felt the provided encodings did not meet their needs for specific data types, such as dimensions with values above and below a baseline (*e. g.,* temperatures), or dimensions with only a few possible numerical values that need to be easy to discriminate. Because BERTIFIER is open source, new encodings can easily be added, tested and empirically compared with Bertin's original encodings. A strong need was also expressed for legends, that currently need to be added in an external authoring application.

Qualitative Data. Supporting qualitative or categorical data is another key limitation of BERTIFIER. Bertin proposed categorical encodings for maps [10] but not for matrices. He typically treated qualitative dimensions with n possible values as n binary dimensions [10], but this approach does not scale to large n. One problem with categorical encoding is that it requires users to explicitly map each possible data value to its visual representation (e.g., a specific color or texture). Previous work showed that this operation can in some cases be automated [40], but users still need a way to tweak the results. Implementing such a feature poses non-trivial problems of HCI design, and may possibly require the addition of other tools than crossets.

Scalability. Like Bertin's physical matrices [10], BERTIFIER does not support very large tables. Although our previous analysis suggests that personal spreadsheets are typically small, large tables exist and can cause problems to BERTIFIER. First, they may require scrolling. Although scrolling is supported, crossets cannot be crossed beyond the viewport. One solution is to zoom out (also supported), but interaction becomes difficult as widgets become smaller. Another approach could be to let users collapse or aggregate groups of rows or columns, or (as suggested by *R*) split a table into several sub-tables to be manipulated independently. Finally, one could consider using arbitrarily large high-resolution displays. Although in principle very large tables can be shown even on a 30" screen, another bottleneck lies in performance, as UI responsiveness quickly decreases with SVG scenegraph complexity.

Participants from the user study suggested various other improvements, such as being able to rotate text headers, set cells' background color, and split the table into subparts that can be reordered separately. Many other extra features can be considered that would support more effective communication, such as adding legends to individual dimensions, labels to individual cells, and various other graphical decorations. Finally, we are considering adding support for interaction history and revisitation based on small multiples as proposed by Bret Victor [56], as well as the ability to export specific states as URLs.

8 Conclusion

We presented BERTIFIER, a tabular visualization authoring tool based on Jacques Bertin's matrix analysis method. BERTIFIER matches and extends the requirements stated and implied by Bertin. We contributed a new interaction technique—the *crosset*—that lets users quickly and easily apply commands to series of rows or columns. Crossets could be used in other Infovis systems (*e. g.,* for dynamic queries), but further evaluations are needed to better assess their strengths and weaknesses. We also introduced *visual reordering*, an approach that lets users apply and tune automatic reordering algorithms in a WYSIWYG manner. This approach fills a gap between flexible but often tedious manipulations, and fast but often unsatisfactory automatic reordering methods.

In his review of matrix reordering methods, Liiv concludes that *"seriation cannot be considered ubiquitously usable, until implemented and shipped as a standard tool in any spreadsheet and internet browser for enabling such analysis. Then one can say that seriation and matrix reordering is usable. That is the main future goal for seriation."* [39]. BERTIFIER is a significant step towards this goal, as our user study suggests it makes it possible for both scientists and a wider audience to explore, analyze and interpret their data, as well as to communicate their findings by visual means. We will continue to extend BERTIFIER based on user feedback, and further hope that its design will inspire extensions to popular commercial tools such as spreadsheet software, so that Bertin's method finally becomes accessible to a wide audience and empowers people with data analysis and communication tools that were so far only accessible to a small number of specialists.

REFERENCES

[1] J. Accot and S. Zhai. More than dotting the i's — foundations for crossing-based interfaces. In *Proc. CHI '02*, pages 73–80. ACM, 2002.

[2] G. Apitz and F. Guimbretière. Crossy: A crossing-based drawing application. In *Proc. UIST '04*, pages 3–12. ACM, 2004.

[3] ARTICQUE. Cartes et données, 2013. Available at `www.articque.com/solutions/cartes-et-donnees/`.

[4] Z. Bar-Joseph, D. K. Gifford, and T. S. Jaakkola. Fast optimal leaf ordering for hierarchical clustering. *Bioinformatics*, 17:S22–S29, 2001.

[5] P. Baudisch. Don't click, paint! using toggle maps to manipulate sets of toggle switches. In *Proc. UIST '98*, pages 65–66. ACM, 1998.

[6] M. Beaudouin-Lafon. Instrumental interaction: An interaction model for designing post-wimp user interfaces. In *Proc. CHI '00*, pages 446–453. ACM, 2000.

[7] W. H. Benson and B. Kitous. Interactive analysis and display of tabular data. *SIGGRAPH Comput. Graph.*, 11(2):48–53, July 1977.

[8] J. Bertin. *Sémiologie graphique : Les diagrammes-Les réseaux-Les cartes*. Mouton/Gauthier-Villars, 1967.

[9] J. Bertin. Graphique et mathématique: Généralisation du traitement graphique de l'information. *Annales. Économies, Sociétés, Civilisations*, 24(1):70–101, 1969.

[10] J. Bertin. *La graphique et le traitement graphique de l'information*. Nouvelle bibliothèque scientifique. Flammarion, 1975.

[11] J. Bertin. *Graphics and Graphic Information Processing*. De Gruyter, Berlin, 1981. Translation: William J. Berg, Paul Scott.

[12] J. Bertin. Cahier des charges pour un mini-ordinateur "graphique". Actes du second Colloque de micro-info-graphique de Rouen (Unpublished). Available at `http://www.aviz.fr/wiki/uploads/Bertifier/bertin-1982-cahier.pdf`, 1982.

[13] J. Bertin. *Sémiologie graphique : Les diagrammes-Les réseaux-Les cartes*. Éditions de l'Ecole des Hautes Etudes en Sciences, Paris, France, les réimpressions edition, 1999.

[14] J. Bertin and J.-H. Chauchat. Logiciel amado (analyse graphique d'une matrice de données). *CISIA, France*, 1994.

[15] G. Caraux and S. Pinloche. PermutMatrix: a graphical environment to arrange gene expression profiles in optimal linear order. *Bioinformatics*, 21(7):1280–1281, Apr. 2005.

[16] S. K. Card, J. D. Mackinlay, and B. Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[17] A. de Falguerolles, F. Friedrich, and G. Sawitzki. A tribute to J. Bertin's graphical data analysis. In *Advances in Statistical Software 6*, pages 11–20. Lucius & Lucius, 1997.

[18] J. Díaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34(3):313–356, Sept. 2002.

[19] A. Dix and G. Ellis. Starting simple: Adding value to static visualisation through simple interaction. In *Proc. AVI '98*, pages 124–134. ACM, 1998.

[20] EVS. European values study 1981–2008, longitudinal data file, 2011.

[21] J.-D. Fekete, J. J. Wijk, J. T. Stasko, and C. North. The value of information visualization. In A. Kerren, J. T. Stasko, J.-D. Fekete, and C. North, editors, *Information Visualization*, pages 1–18. Springer-Verlag, Berlin, Heidelberg, 2008.

[22] S. Few. The slow data movement: My hope for 2013. `http://www.perceptualedge.com/blog/?p=1460`. Online, accessed 2013-03-22., 2013.

[23] J. Fras, R. Gimeno, and P. Vicens. *Matrix: programme de traitement matriciel en Basic TO7*. CNDP, 1984.

[24] J. Fuchs, P. Isenberg, A. Bezerianos, F. Fischer, and E. Bertini. The influence of contour on similarity perception of star glyphs. *TVCG*, 2014. To appear.

[25] M. Gardet and Y. Irid. Archives du laboratoire de cartographie, inventaire, 2012.

[26] W. A. Gibson. A simple procedure for rearranging matrices. *Psychometrika*, 18(2):111–113, 1953.

[27] R. Gimeno and M.-F. coise Durand. Mise en place et utilisation d'un logiciel de cartographie et traitement des données dans l'enseignement de la géographie. *Revue de géographie de Lyon*, 63(2):64–68, 1988.

[28] Y. Girault and T. Rukingama. Evaluation de l'exploitation d'une visite au zoo par le traitement graphique. 1988.

[29] R. Gosselin. T_alk, 2008. Available at `http://rg75.free.fr/hyperespace.T_AlK.fr.htm`.

[30] T. R. G. Green and M. Petre. Usability analysis of visual programming environments: a 'cognitive dimensions' framework. *Journal of Visual Languages & Computing*, 7(2):131–174, 1996.

[31] B. F. Green Jr. The computer revolution in psychometrics. *Psychometrika*, 31(4):437–445, 1966.

[32] L. Guttman. What is not what in statistics. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 26(2):81–107, 1977.

[33] N. Henry and J. Fekete. Matrixexplorer: a dual-representation system to explore social networks. *TVCG*, 12(5):677–684, Sept 2006.

[34] H. Hinterberger. The visulab®: an instrument for interactive, comparative visualization. 2010.

[35] S. Huron, Y. Jansen, and S. Carpendale. Constructing visual representations: Investigating the use of tangible tokens. *TVCG*, 2014. To appear.

[36] E. Kwan, I. R. R. Lu, and M. Friendly. Tableplot: A new tool for assessing precise predictions. *Zeitschrift für Psychologie / Journal of Psychology*, 217(1):38–48, 2009.

[37] J. Le Fourn and A. Mailles. Un micro ordinateur graphique : Programmes réalisés et exemples d'application. 1984.

[38] J. Li, J.-B. Martens, and J. J. van Wijk. A model of symbol size discrimination in scatterplots. In *Proc. CHI '10*, pages 2553–2562. ACM, 2010.

[39] I. Liiv. Seriation and matrix reordering methods: An historical overview. *Stat. Anal. Data Min.*, 3(2):70–91, Apr. 2010.

[40] S. Lin, J. Fortuna, C. Kulkarni, M. Stone, and J. Heer. Selecting semantically-resonant colors for data visualization. In *Proc. EuroVis '13*, pages 401–410. Eurographics Association, 2013.

[41] P. Orús and G. Villarroya. TGINF : graphic information processing for teaching. Reims, France, 2003.

[42] G. Palsky. *L'esprit des cartes: approches historiques, sémiologiques et sociologiques en cartographie: diplôme d'habilitation à diriger des recherches*. Université de Paris XII-Val-de-Marne, 2003.

[43] 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. CHI '94*, pages 318–322. ACM, 1994.

[44] W. S. Robinson. A method for chronologically ordering archaeological deposits. *American Antiquity*, 16(4):293–301, 1951.

[45] S. Rubin. Available at `www.eecs.berkeley.edu/~srubin/toys/reorderable/`.

[46] G. Sawitzki. Extensible statistical software: On a voyage to oberon. *Journal of Computational and Graphical Statistics*, 5(3):263–283, 1996.

[47] G. Sawitzki. Bertin matrices. an R implementation, 2012.

[48] B. Shneiderman. Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69, Aug. 1983.

[49] H. Siirtola. Interaction with the reorderable matrix. In *Proc. IV '99*, pages 272–277. IEEE, 1999.

[50] H. Siirtola. Interactive cluster analysis. In *Proc. IV '04*, pages 471–476. IEEE, 2004.

[51] H. Siirtola and E. Mäkinen. Constructing and reconstructing the reorderable matrix. *Information Visualization*, 4(1):32–48, Mar. 2005.

[52] M. Spenke, C. Beilken, and T. Berlage. Focus: The interactive table for product comparison and selection. In *Proc. UIST '96*, pages 41–50. ACM, 1996.

[53] E. A. Suchman. The scalogram board technique for scale analysis. *Studies in Social Psychology in World War II*, 4:91–121, 1950.

[54] Tableau Software, 2014. `www.tableausoftware.com/`.

[55] F. Vergneault, R. Lamontagne, and J. Bertin. Traitement graphique d'une information: les marines royales de france et de grande-bretagne (1697-1747). *Annales. Économies, Sociétés, Civilisations*, 22(5):991–1004, 1967.

[56] B. Victor. Drawing dynamic visualizations. Stanford HCI Seminar. Video available at `http://vimeo.com/66085662`, 2013.

[57] P. Wegner. Why interaction is more powerful than algorithms. *Commun. ACM*, 40(5):80–91, May 1997.

[58] L. Wilkinson. Statistical methods in psychology journals: Guidelines and explanations. *American Psychologist*, 54(8):594–604, Aug. 1999.

[59] L. Wilkinson and M. Friendly. The history of the cluster heat map. *The American Statistician*, 63(2):179–184, 2009.

[60] H.-M. Wu, Y.-J. Tien, and C.-h. Chen. Gap: A graphical environment for matrix visualization and cluster analysis. *Computational Statistics & Data Analysis*, 54(3):767–778, Mar. 2010.