

Table Lens as a Tool for Making Sense of Data

Peter Pirolli and Ramana Rao

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304, USA
{pirolli, rao}@parc.xerox.com

ABSTRACT

The Table Lens is a visualization for searching for patterns and outliers in multivariate datasets. It supports a lightweight form of exploratory data analysis (EDA) by integrating a familiar organization, the table, with graphical representations and a small set of direct manipulation operators. We examine the EDA process as a special case of a generic process, which we call *sensemaking*. Using a GOMS methodology, we characterize a few central EDA tasks and compare performance of the Table Lens and one of the best of the more traditional graphical tools for EDA i.e. Splus. This analysis reveals that Table Lens is more or less on par with the power of Splus, while requiring the use of fewer specialized graphical representations. It essentially combines the graphical power of Splus with the direct manipulation and generic properties of spreadsheets and relational database front ends. We also propose a number of design refinements that are suggested by our task characterizations and analyses.

Keywords

Information visualization, multivariate visualization, database visualization, evaluation, GOMS, exploratory data analysis

INTRODUCTION

The Table Lens (Rao & Card) has been informally characterized as an effective tool for making sense of numerical and categorical data of the sort typically found in multivariate datasets. Here, we would like to refine the notion of "making sense" in a way that allows us to judge the effectiveness of tools such as the Table Lens. We would like to know if our tools maximize the support for necessary sensemaking functions while minimizing the costs of learning and use. Furthermore, we would like a characterization of the tool that exposes opportunities for design refinements and guidance in assessing them. Toward these ends, we will characterize a space of tasks

for a particular instance of sensemaking, basic Exploratory Data Analysis (EDA). Our task characterization draws on concepts and methods from studies of sensemaking and discovery as problem-solving processes as well as from cognitive modeling techniques for human computer interaction.

In the words of Tukey, the field's founder, EDA includes a variety of techniques for "looking at data to see what it seems to say." For the purposes of this paper we focus on two typical EDA tasks involving multivariate datasets: (i) assessing a batch of data (i.e. the features of each single variable) and (ii) finding lawful relations among a set of observed variables. The first task is a kind data "browsing" which is typical of early stages of exploration when general insights are sought and the second task is perhaps the canonical task of EDA.

These two tasks can be accomplished with Table Lens as well as with its cousins in two categories: more broadly-purposed productivity applications (e.g. spreadsheets and relational databases) and more specialized exploratory data analysis packages (e.g. Splus, DataDesk). Table Lens, in some sense, integrates the virtues of both categories: the direct manipulation ease of use of the mainstream products with the tuned support for exploratory statistical processes.

Using the GOMS methodology (Card, Moran, & Newell, 1983; Olson & Olson, 1990), we characterize and analyze methods for accomplishing the two above mentioned EDA tasks using the Table Lens. This analysis provides basic time estimates for task (and constituent subtask) performance. Comparison with corresponding methods for performing these tasks using Splus and Excel reveals quantitative performances differences as well as qualitative understanding of the source of these differences. Perhaps the most important consequence of this kind of analysis is that it can provide insights which can fuel further design. In our case, we propose a number of refinements to the Table Lens which are clearly indicated by our analysis.

SENSEMAKING

Sensemaking refers to activities in which external representations such as texts, tables, or figures are interpreted into semantic content and represented in some other manner. Retrospective case studies of sensemaking in information work were presented in Russell, Stefik, Pirolli, and Card (1993). A general recurring pattern of activity called a *learning loop complex* was observed. This loop involves component processes including: (a) a generation loop that involves a search for representations to capture important regularities in collected information, (b) a data coverage loop in which information is encoded into the representation forming encodons, and (c) a representation shift loop in which ill-fitting residue information triggers and guides search for a new representation, and (d) encodon consumption (formation) in which the representation drives the search for information. Here, *representations* refers collectively to internal cognitive structure as well as external resources such as tables, graphs, or hypermedia structures.

In cognitive science, the learning loop is consistent with process models of discovery that have been developed to address historical accounts of scientific breakthroughs (Langley, Simon, Bradshaw, & Zytkow, 1987) and protocol studies of people finding empirical laws (Qin & Simon, 1990). In general, these accounts can be characterized as *search processes* operating in *problem spaces*.

The notion of problem solving as a process of search in problem spaces was developed in Newell and Simon's (Newell & Simon, 1972) classic human information-processing theory. Basically, a human problem solver is viewed as an information-processing system with a problem. A *task environment* is the external environment, inclusive of the problem, in which the information processing system operates. For instance, a data analysis problem that required the use of the Table Lens would constitute a task environment. A *problem space* is a formalization of the structure of processing molded by the characteristics of the information-processing system, and more importantly, the task environment. A problem space is defined in terms of *states* of problem solving, *operators* that move the problem solving from one state to another, and *evaluation functions*. As a family, the discovery models can be characterized as *dual problem space search* models (Langley, et al., 1987) in which processing alternates between search in a problem space of possible representations and a search in a problem space that fits data to the representations.

In the case of Exploratory Data Analysis, one searches for models on one hand and searches to assess the fit of the models to the data on the other. In Tukey's (1977) classic characterization, data values can be viewed as compositions of a fitted part and a residual part:

$$\text{Data} = \text{Fit} + \text{Residual}.$$

A typical goal in EDA might be, for example, to produce an equation that predicts the values of a dependent variable based on values of one or more independent variables. The predicted values would be the fitted part of the actual values. The differences between the predicted values and the actual values are the residual values (or residuals)--the residue of the sensemaking process in EDA. Typically, an evaluation function on the quality of the EDA modeling process involves assessing the total amount of residuals, and the guiding heuristic for the search process is to minimize that value, while also keeping the complexity of the model to a minimum (for instance one might prefer equations with fewer variables and lower-order polynomials).

REPRESENTATIVE EDA TASKS

EDA involves ferreting out regularities and irregularities as well as relationships in multivariate datasets (also variously called cases by variables array and multidimensional datasets and roughly the same as a relational table). Such datasets can be considered as multiple batches (i.e. the collected values of a single variable) or multiple cases (i.e. the values for all variables of single observation). For the purposes of this paper, we focus on two typical EDA tasks: (i) assessing the properties of a batch of data including central values, extremes, dispersion, and symmetry; and (ii) discovering lawful relationships among observed variables e.g. producing an equation that predicts the values of a dependent variable based on values of one or more independent variables.

The first task is a browsing task which involves browsing the batch of values for each variable. Typical EDA displays for supporting this task include Stem and Leaf Displays, Letter Value Displays, and Boxplots. The Letter Value Display is a textual summary of the batch that typically shows the median, quarters (the values half way between median and the extremes), and the extremes. The other displays more fully depict the batch as a whole and help with assessments like the following features:

- How nearly symmetric the batch is.
- How spread out the numbers are.
- Whether a few values are far removed from the rest.
- Whether there are concentrations of data.
- Whether there are gaps in the data.

The second task typically entails an iterative process, described above in terms of the concept of a dual problem space search. In data analysis terms, given a dependent variable that the analyst is interested in modeling with an equation involving a set of independent variables, the first step is to find a candidate variable that is highly correlated to the variable. This correlation may involve "re-expression" of the variable (i.e. transforming the values

with a power function such as $1/x^2$, $1/x$, $\ln(x)$, x^2 , x^3) to linearize the relationship between the two variables. With a selected independent variable in hand, the analyst would subtract the indicated fit (obtained from a regression) from the dependent variable to obtain a residual. That is: $\text{Residual} = \text{Data} - \text{Fit}$. The process can then be repeated and additional independent variables can be considered to further explain the residual.

EDA TOOLS

The two EDA tasks---batch assessment and variable modeling---can be performed utilizing Table Lens, Splus, and Excel. Both Splus and Excel are much more richly featured than the Table Lens, and in general offer a greater variety of methods for performing these tasks. In our comparative analysis we focus on the best method offered by each tool.

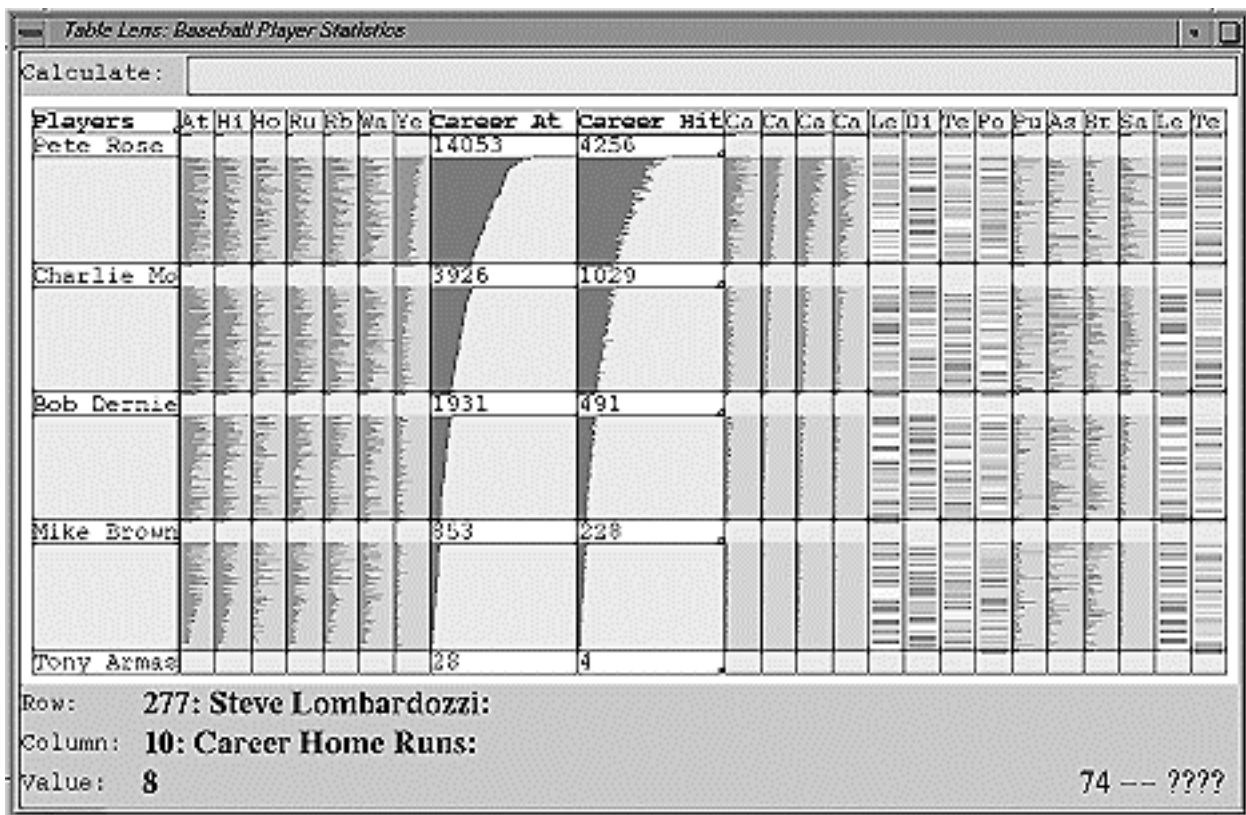


Figure 1. Table Lens visualizing a set of baseball statistics.

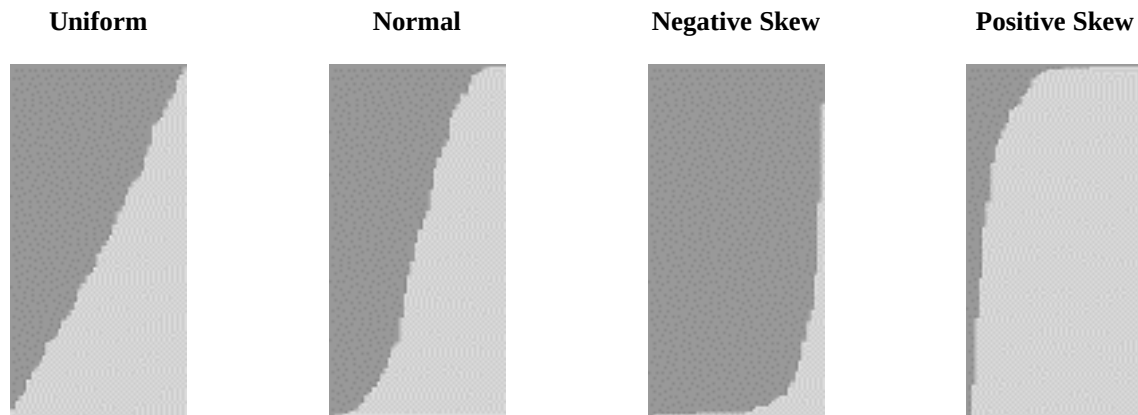


Figure 2. Prototypical distributions as they appear in the Table Lens.

Table Lens

The Table Lens (depicted in Figure 1) provides a structured graphical representation that supports browsing of the values for hundreds of cases and tens of variables on typical workstation display. Table Lens is essentially the graphical equivalent of a relational table or spreadsheet in which the rows represent cases and the columns represent variables. In addition, particular rows and columns can be

assigned variable amounts of space, which allows "opening them up" to support direct incorporation of textual representations of values.

For quantitative variables, a graphical bar is used to represent the values. The bars within each variable's column align with its left edge which can indicate a minimum value, zero, or a lower "fence" (i.e. an outlier

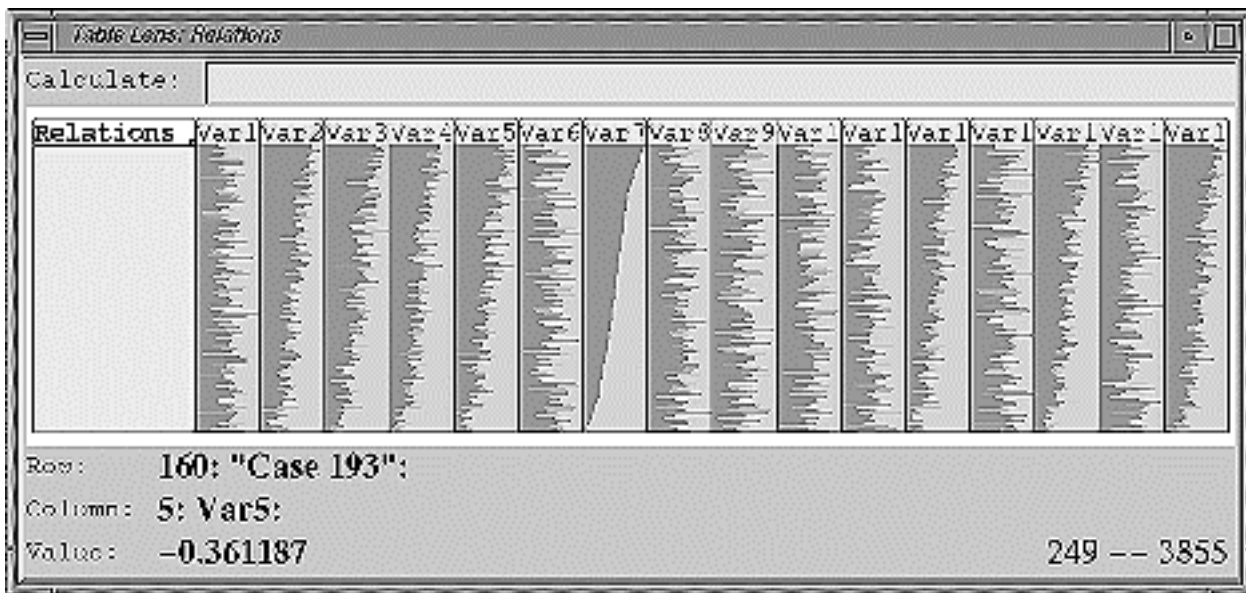


Figure 3. Correlations among column variables become apparent when one of the columns is sorted.

cutoff). The length of the bar is proportional to the relative size of the represented value. The use of graphical representations not only provides a scale advantage---since the bars can be scaled to one pixel wide without perturbing relative comparisons---but also an exploration advantage, since large numbers of tiny bars can be scanned much more quickly than a bunch of textually represented numbers.

Besides controlling which rows and columns are "focused" (i.e. given space to display textual values), there are two other major operations necessary for performing the EDA tasks: sorting a column and creating a new column computed using a formula based on other columns. Performing a "down gesture" on a column (i.e. dragging the pointer down with mouse button held down in column and then releasing) causes the cases to be sorted by the values in that column. A formula can be entered by typing in an equation involving existing columns which can be "check" gestured during formula entry or else typed in by name. When the formula is committed by the user, a new column appears immediately to the right of the rightmost referenced column.

After a column is sorted, properties of the batch of values can be estimated by graphical perception and some amount of display manipulation. Central values can be viewed by focusing on values near the center of the column. Likewise, extremes can be read by focusing on the ends of the column. Assessing dispersion is a matter of comparing those extrema. Classifying the shape and the skew is a learnable skill, which is somewhat different for a curve of batch values than it is for a histogram. Figure 2 presents four common kinds of distributions that were used in our studies. Finally, a column can be re-expressed by invoking a formula based on that single

column. Introducing a special operator facilitates many of these subtasks. This operator essentially incorporates the virtues of a five value summary (i.e. letter values) into the display by automatically focusing on the extremes, the median, and the quarters (see Figure 1). Besides revealing precisely the values for these locations, it also visually breaks up the table into quarters which aids shape and skew assessment.

Sorting is also the first step of looking for correlations among variables. After a first variable has been sorted, if another variable is correlated then its values will also appear to be sorted. For instance, Figure 3 shows a Table Lens display in which one of the columns--Column 7 from the left--has been sorted and positive correlations with variables in Columns 2, 3, 4, 5, 12, 14, and 16 are revealed. Thus looking for correlated variable is a matter of scanning across the columns to identify other columns which seem to exhibit a descending trend. New column computations can be used to fit a correlated variable to the sorted variable (or a re-expressed version of either or both of the variables) and in turn to display the residuals of that fit. The sorting and fitting process can then be iterated on the residual column.

Splus

Splus (Becker, Chambers, & Wilks, 1983) is a data analysis environment based on an interpretive programming environment model in which a variety of data manipulation and viewing techniques are integrated as a library of primitive operators. In particular, after having loaded and named a multivariate dataset in the interpreter's environment, a user can invoke a "brush tool," commonly known as a scatterplot matrix (Becker & Cleveland, 1984), which displays a matrix of all pairwise scatterplots. Optionally a histogram of each variable can be placed at the base of each column of scatterplots

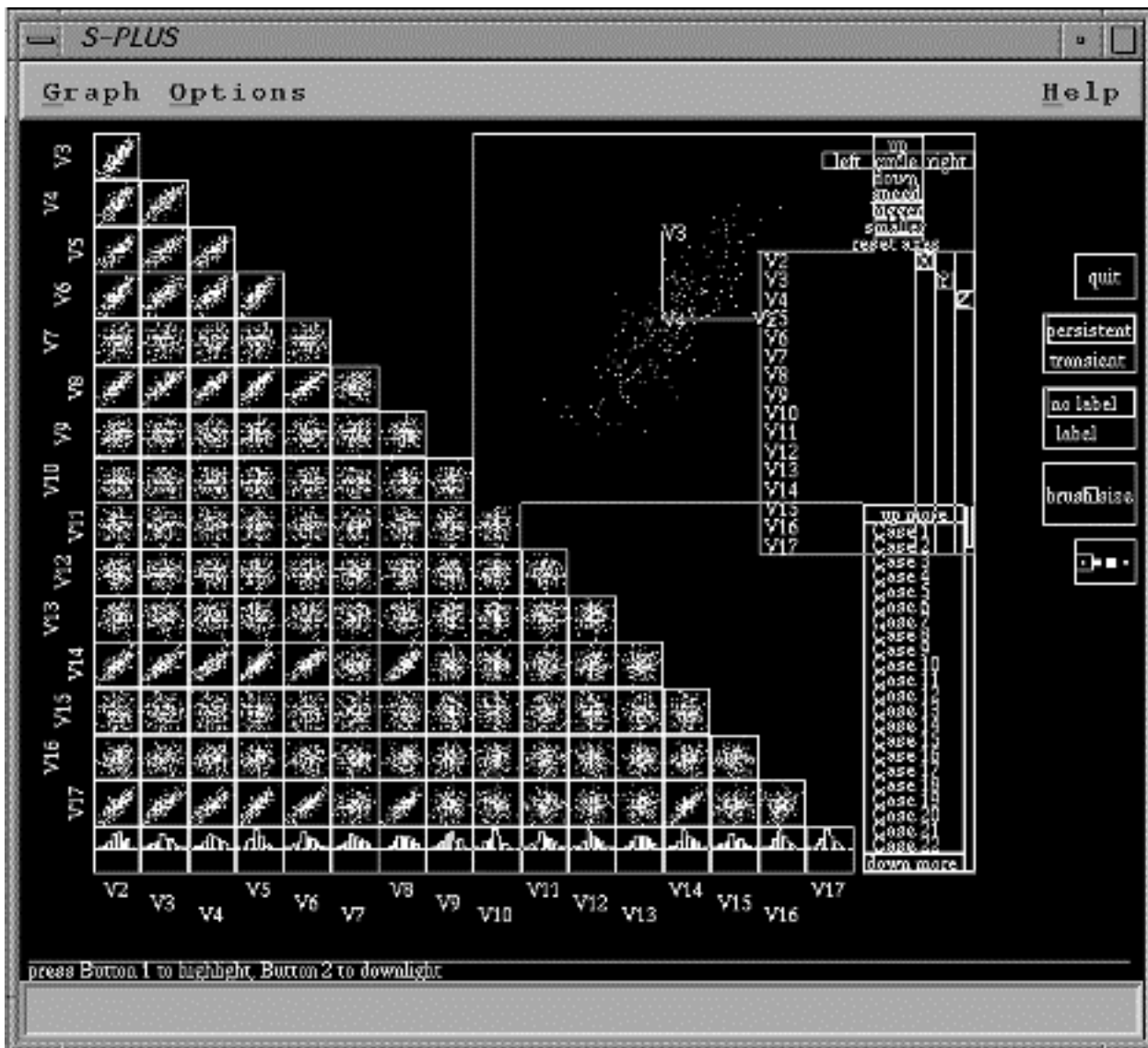


Figure 4. The Splus brush tool.

associated with that variable. Figure 4 shows a brush tool utilizing the histogram option. This view provides the strongest methods for performing both of our example tasks.

Batch assessment is performed by graphical perception of the histograms for the chosen variable displayed at the bottom of the column of matrices for that variable. Values have to be read by examining the histogram which for the case of many variables may be quite small (as in the Figure 4). An alternative method involves invoking a stem and leaf or boxplot display of the variable of interest by invoking the appropriate command in the interpreter. Re-expression is performed by invoking a transform function on the batch of data.

Cross variable correlation involves scanning the row or column associated with a chosen variable and looking for scatterplots that reveal well formed lines. With training, one can learn whether the correlation is positive or negative depending on the angle of the line and also what re-expression may be indicated by curvatures in the relationships. One difficulty for inexperienced or occasional users is that interpretation is flipped depending on whether rows or columns are scanned.

Fitting variables (possibly re-expressed) and examining residuals involves a series of commands in the interpreter that amount to more or less the same operations as required in the Table Lens column computations, though there is more explicit datasets management involved (e.g. a command line style of naming etc.)

Excel

Excel is a broad spectrum productivity application that can be used to perform a variety of tasks, many which aren't even analytical in nature. The central representation is a textual table and additional mechanisms are provided for invoking computations and graphics, and tuning the tables purpose for handling multivariate data. After a multivariate table has been loaded into Excel, a user can draw on several separate mechanisms to perform EDA tasks. Some of the elements of the task can be performed by direct manipulation of the textual display, others require either the use of the embedded formula language or of the mechanism for generating graphical views of the data. So for example, some of batch assessment---extremes, medians, quarters---can be performed by sorting by a particular column, however shape assessment is all but impossible without using a built-in statistical operator or by plotting the elements of the batch. All of the considered methods take a great deal of manipulation and knowledge to produce either summaries by computation or graphical views. Since in each case the best method involves generating one or another of the representations of Splus or Table Lens at a much greater initiation cost, we don't include Excel in our detailed analysis.

GOMS ANALYSIS OF EDA TOOLS

When faced with ill-defined sensemaking problems such as EDA we often want to explore the space of content and possible courses of actions available to us before exploiting them. In EDA one might, for instance, first identify which variables seem to be most strongly related to one another out of a large set of variables before

investing time in refining exactly how those variables are related.

Here we present assessments of some EDA methods carried out with Table Lens and Splus. These assessments concern quantitative variables only, as these can be studied in simple ways with Table Lens.

Exploration: Assessing Properties of a Batch of Variables

One basic task in EDA is to understand important features of the distribution of values of a variable over the sample of cases that are under examination. An important set of such features are (a) the central tendency (e.g., the mean, mode, or median), (b) the dispersion (e.g., the range, standard deviation, or inter-quartile range), (c) the shape (e.g., normal, skewed, uniform), and (d) extreme values or outliers (e.g., the minimum and maximum values). We analyzed a representative task of finding such features using the Table Lens and Splus. The representative task involved finding the median, inter-quartile range (IQR), judging the shape (normal, uniform, positively skewed, or negatively skewed), and finding the maximum and minimum values.

In GOMS notation, the method for performing the variable characterization task is:

Goal: Characterize-variable
Goal: Find-median
Goal: Find-IQR
Goal: Find-shape
Goal: Find-min&max.

Table 1. Relevant perceptual, cognitive, and motor time-cost parameters from the HCI literature.

Parameter	Value	Source
Visual scan to target (1° arc ≈ .25" @ 15" eye-screen distance)	4 msec/degree of visual arc	OO
Decode abbreviation	50~66 msec	OO
Mentally compare two words	47 msec	CMN
Point mouse at target of size S at distance D	$1030 + 960 \log_2(D/S + .5)$ msec	CMN
Read a word	300 msec	CMN
Mouse click	70 msec	CMN
Mouse gesture	70 msec	CMN
Keystroke	372 msec	CMN
Perceptual Judgement Time	92 msec	OO

Execute Mental Step	70 msec	OO
Retrieve from Memory	1200 msec	OO

Note: CMN = Card et al. (Card, et al., 1983), OO = Olson and Olson (Olson & Olson, 1990).

Table 2. GOMS analysis of methods to judge the shape of a distribution of the Nth column in a Table Lens display. Time estimates (in msec) are in bold.

<u>METHOD: FIND-SHAPE =</u>		
Goal: Find-shape ; of the Nth column from left in Table Lens		
Goal: Find-and-sort-column		[1893 + 101 N]
Goal: Judge-distribution-shape		[93]
Total (msec) = 1986 + 101 N		
<u>METHOD: FIND-AND-SORT-COLUMN =</u>		
Goal: Find-and-sort-column		
Goal: Match-column-variable-name ; first column		
Scan-to-column ; first column	[(3.5"/.25")•4 =	56]
Decode-abbreviation[COLUMN-NAME]		[50]
Match[COLUMN-NAME, VARIABLE-NAME]		[47]
Subtotal (msec)		153
Goal: Match-column-variable-name ; If necessary, iterate N - 1 times		
Scan-to-column ; next column	[(.25"/.25")•4 =	4]
Decode-abbreviation[COLUMN-NAME]		[50]
Match[COLUMN-NAME, VARIABLE-NAME]		[47]
Subtotal (msec)		101
Goal: Verify-column-match ; If there is a name match		
Mouse-Point [COLUMN-NAME]	[1030 + 96 log₂(2"/.25" + .5) =	1330]
Scan-to-status-bar ; at lower left of window	[(6"/.25")•4 =	96]
Read[STATUS BAR]		[300]
Match[STATUS BAR, VARIABLE-NAME]		[47]
Subtotal (msec)		1773
Flick-Down ; If match found		[70]
Total (msec) = 153 + (N - 1)101 + 1771 + 70 = 1893 + 101 N		

Table 3. Summary of time costs for Table Lens methods for assessment of properties of a batch of variables. Estimates in msec.

Method	Literature-based Estimate + Average System Response	Empirical Estimate + Average System Response
Find-median (for N th column)	8325 + 101 N	8470 + 64 N
Find-IQR (for N th column)	13811 + 101 N	13710 + 64 N
Find-shape (for N th column)	1986 + 101 N	2677 + 94 N
Find-max&min (for N th column)	8737 + 101 N	8470 + 64 N

Random-variable-walk (for V variables)	$33061 V + 202 V^2$	$33741 V + 144V^2$
Iterate-over-variables (for V variables)	$366 + 15034V$	$17710V$

Table Lens Analysis

To illustrate our method of analysis, we will expand the task analysis of one of these subgoals, Find-shape, down to basic operations involving the Table Lens, and show how we arrived at estimates of time costs from the human-computer interaction (HCI) literature. The goal of finding the shape of the distribution of a variable can be achieved in the Table Lens by (a) visually locating the column associated with the variable, (b) sorting the values in that column using a flick-down mouse gesture, and (c) making a perceptual judgment of the resulting shape. Table 1 contains relevant time parameters for perceptual, cognitive, and motor operations from a variety of literature sources. The Find-shape method is presented in Table 2, along with the submethod it uses to find and sort the desired column. Table 2 also presents an analysis of the time costs of the methods using the literature-based parameter estimates in Table 1.

As an empirical check, one of us performed self-timed speeded trials of finding and sorting variable columns in a 23 variable Table Lens dataset. Each variable was tested three times and the full sequence of trials was randomized. A regression ($R^2 = .93$) yielded empirical time cost estimates for the Find-shape method that are presented alongside the literature-based estimates in Table 3. The literature-based estimate of 101 msec per column scanned is about a 7% overestimate over the observed estimate of 94 msec per column scanned. The literature-based estimate of 1893 msec for the verification of the column match and the flick-down gesture is a 784 msec underestimate of the 2677 msec obtained empirically (about 30%). This empirical time includes, however, observer reaction time (trials were self-timed using a stopwatch).

Table 3 summarizes the literature-based and empirically estimated time costs for the Table Lens methods associated with the finding important features of a variable distribution. For each literature-based estimate, we constructed a GOMS task analysis like the examples in Table 2, and used relevant time-cost parameters from Table 1. The empirical estimates were produced from a number of small self-timed user experiments (by one of the authors). For each method or submethod, and at each parameter level (e.g., location of the target column), three to five replications were conducted. Similar mini-experiments were conducted to assess system response times—for example, to assess the time taken to sort a column. These system times were added, when

appropriate, to the literature-based and empirical estimates of human performance times for the methods.

The Find-median, Find-IQR, Find-shape, and Find-min&max analyses assume that the user starts with a focus at a central location at the top of the Table Lens display, and must scan to some particular variable whose location is unknown. The last two rows of Table 3 present analyses for two methods that would iterate through all the variables in a dataset and characterize the median, IQR, shape, and extreme values. The Random-variable-walk simply assumes that the user randomly chooses a variable and a feature to characterize at random (without replacement), performs the method, and repeats the process until all variable features have been identified. Its time cost is just the sum of the time-costs for the Find-median, Find-IQR, Find-shape, and Find-min&max methods over V variables in a dataset. The Iterate-over-variables method, assumes that the user iterates through each variable, left-to-right, finding the median, IQR, shape, and extreme values. These two methods might be considered reasonable boundary cases at the highly inefficient and highly efficient ends of the spectrum of methods for finding the relevant distributional properties of variables in a dataset. The time costs for these two methods as a function of the number of variables in the data batch are presented in Figure 5.

Overall, the absolute differences between literature-based and empirically-based estimates range from about 2% to 50%, and in general there is good agreement between the two. For current purposes, this level of agreement seems satisfactory, so we have used literature-based estimates for the remaining analyses.

Splus

Table 4 presents literature-based time-cost estimates for Splus methods analogous to those analyzed for the Table lens. The methods for Find-median, Find-IQR, and Find-max&min largely involve cycles of typing in commands to the Splus interpreter and reading off the returned results. The Find-shape method involves typing in a command to plot a histogram and then judging the shape of the distribution.

At the bottom of Table 4 are methods for iterating through all the variables in the data set. Again, similar to the Table Lens method, the random-variable-walk method involves randomly choosing one of the first four methods and applying it to a randomly chosen variable (random choices are made without replacement). The iterate-over-variables method involves invoking the brush tool, making judgments of all the variable distribution

shapes, then iteratively finding the median, IQR, and extreme values for each variable.

These two Splus methods are compared to the analogous Table Lens methods in Figure 6 (all estimates are literature-based). Figure 6 shows that time costs of the Table Lens methods bracket the time costs of the Splus methods. Most interesting is the suggestion that it is more efficient to iterate through a batch of variables in Table Lens extracting important features.

Exploration: Assessing Relations Among Variables

Another phase of EDA involves judging features about the relationships among variables. Here we examine the simple task of judging if two variables seem to be correlated. Later, we will discuss the tasks involved in determining more precisely how two variables are related.

Figure 5. Comparison of literature-based and empirically-based estimates for time-cost functions for Table Lens methods for finding important features of all variables V in a dataset.

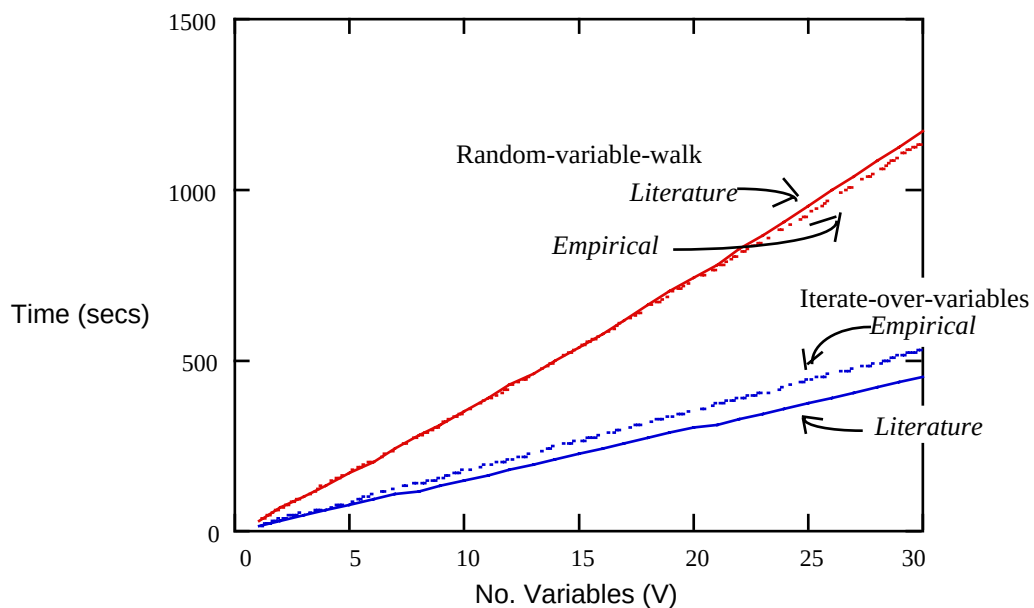


Figure 6. Comparison of time-cost functions for Table Lens and Splus methods for finding important properties of all variables V in a dataset. Time-costs are literature-based.

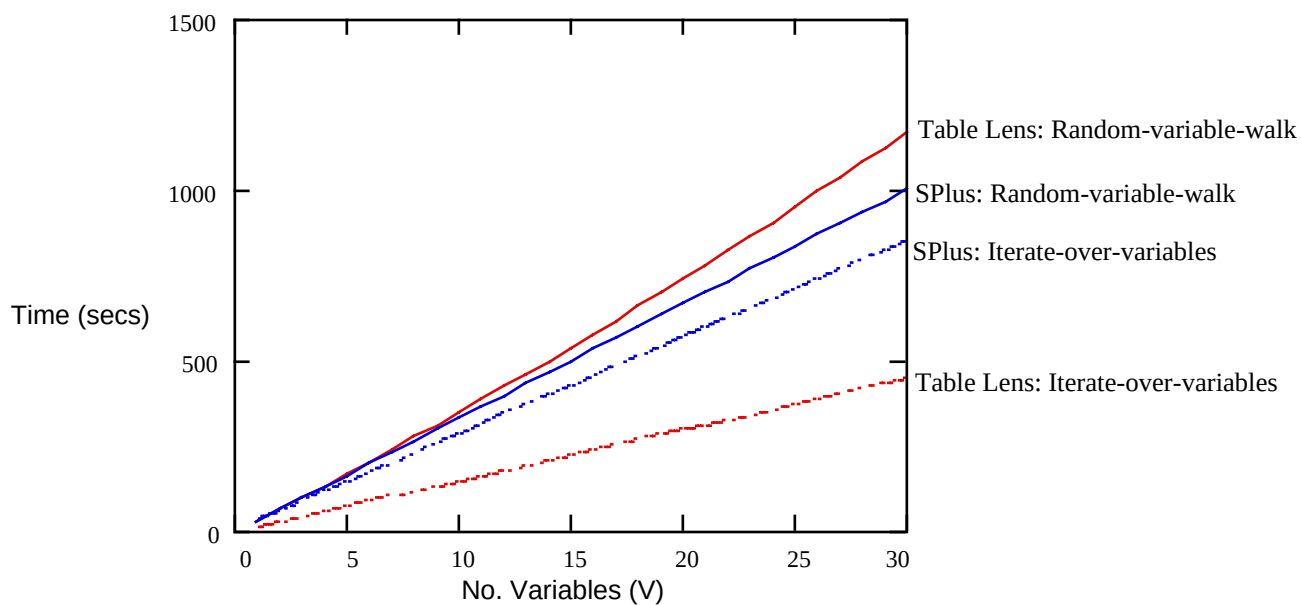


Table 4. Summary of time costs for Splus methods for assessment of properties of a batch of variable. Estimates in msec.

Method	Method Summary	Literature-based Estimate
Find-median	- Recall command and variable name - Execute mental steps and keystroke median command and arguments - Read result	5840
Find-IQR	- Recall commands and variable name - Execute mental steps and keystroke quartile command and arguments - Read upper and lower quartile results - Keystrokes to subtract results - Read result	11850
Find-shape	- Recall command and variable name - Execute mental steps and keystroke hist command and arguments - Scan to histogram display and judge shape	5344
Find-max&min	- Recall commands and variable name - Execute mental steps and keystroke min and max commands and arguments - Read results	10480
Random-variable-walk (for V variables)	Randomly: try each method above on each variable	33514 V
Iterate-over-variables (for V variables)	- Recall brush command and dataset name - Execute mental steps and keystroke brush command and arguments - Iterate through histograms making shape judgments - Do Find-median, Find-IQR, Find-min&max for each variable	8086 + 28338 V

Table 5. Time cost functions for Table Lens and Splus for basic methods of judging the related variables in a dataset of V variables. Estimates in msec.

Method Summary	Literature-based Estimate
Table Lens For V variables, skipping C variables in "clusters" • Scan through each variable left-to-right • Sort the column (mouse-point and gesture) • Scan columns to the right, judging if they are correlated	$366 + 401 V + 1156 (V - C - 1) + 175 V (V - C - 2)$
Splus For V variables, with R related variables • Scan through each pairwise scatterplot on each variable row of a brush display • For each related variable scan and read the variable names	$32 V + 175 V (V - 1) + 664 R$

Table Lens

The basic method for finding relationships among variables in Table Lens involves sorting a variable column, and then scanning other columns to see which have shapes similar to (or perhaps inverted from) the shape of the sorted variable. When a cluster of several related variables is observed and remembered, the user can eliminate those from further consideration and concentrate on the remaining variables. This method and the literature-based time-cost estimate are presented in Table 5. As an empirical check, another set of self-timed speeded tasks was evaluated. Each evaluated dataset contained 16 variables. Three datasets (called 1-8 datasets) contained one cluster of eight related variables and eight unrelated variables. Three datasets (called 4-2 datasets) contained four clusters of two related variables and eight unrelated variables. For a cluster of related variables, the inter-variable correlations were $r = .9$. All variables contained 200 normally distributed values. Judging relations in 1-8 datasets took $M = 46.87$ secs and 4-2 datasets took $M = 57.17$ secs. The literature-based estimates are, 35.63 secs for $V = 16$ variables and $C = 7$ skipped cluster variables, 47.50 secs for $V = 16$ and $C = 4$, and 63.32 secs for $V = 16$ and $C = 0$.

Splus

In Splus, we assume that the main method for judging relations among a large batch of variables involves using the all-pairwise scatter plot in the Splus batch tool. Each pairwise scatter plot must be scanned and, when a relationship is detected, the variable names must be found and read. This method and its literature-based time-cost estimate are also presented in Table 5. Figure 7 presents a comparison of the Table Lens and Splus methods in Table

5. Again, it appears that the Table Lens methods have comparable time costs to Splus.

Exploitation: Search Through the Form of Relations

Our analyses so far have not have not focused on the tasks involved in finding the specific form of relation among sets of variables, once they have been identified. Typically this relies on the use of mathematical transformations and statistical techniques such as regression. In our design section we talk about some ways of incorporating direct interaction techniques for performing techniques that achieve these tasks.

Learning Costs

Our analyses suggest that the Table Lens achieves comparable performance with a well-established interface for EDA in the form of Splus. It does so, however, by relying on the use of only a few simple direct-manipulation commands such as column sorting. This suggests, that Table Lens is likely to be easier to learn.

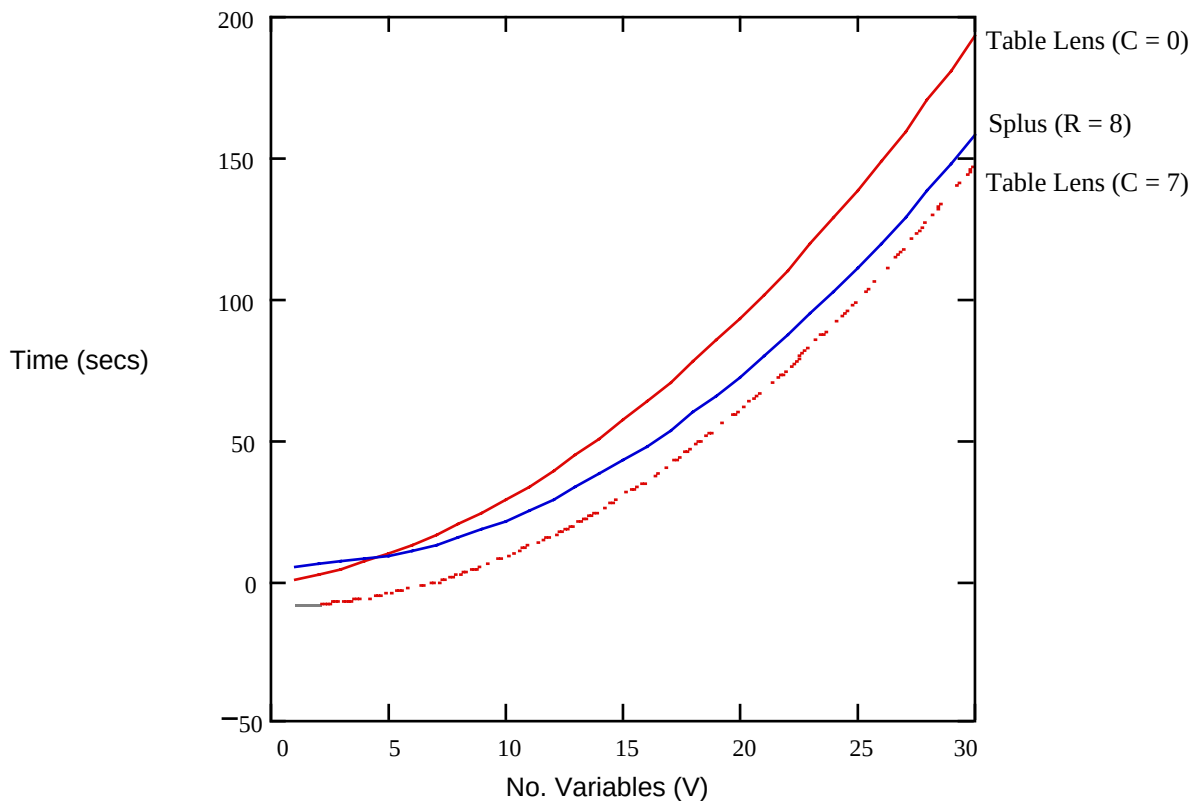
DESIGN REFINEMENT

Considering the virtues of traditional EDA displays, the structure of the studied EDA tasks, and the costs for various methods suggests a number of refinements to Table Lens which would improve performance times for the tasks studied.

Boxplot Values.

One of the advantages of the Table Lens is that it can integrate a number of separate EDA graphical tools into a single representation which can be consistently interpreted. For example, the "quartering" operation described above opens focus on the values which represent the values shown in the EDA five value summary table i.e.

Figure 7. Comparison of time-cost functions for finding related variables in a dataset using methods in Table Lens and Splus.



the median, the two quarters, and the two extremes. Done in the context of the tabular form it is quite clear how those values fall in the overall distribution (see Figure 1).

Another useful EDA graphical tool is the boxplot which graphically subsumes the letter values as well as others values. It conveys at a glance central values, skewness, tail length, and outliers. The box plot is a graphical structure that aligned with an axis representing the range of values in a batch and additional features are presented, including the median, quarters, outlier cutoffs, and outliers. Since the horizontal extent of a Table Lens column essentially represents a coordinate system for the range of values, the basic box plot can be shown superimposed on the base or the head of the column. Furthermore, rows representing extreme outliers, which lie outside the outer fence (typically defined as three-halves times the interquarter spread) can have their bars colored differently to set them apart.

Direct Manipulation Re-expression.

In both batch assessment and variable fitting task, a common need is to re-express a batch of data. A technique frequently used by expert analysts is to

transform a data set using members of a "ladder of powers," which is a particular ordering of power transforms that distort the curve with increasing power to tame non-linearities and to variably treat one end or the other of the curve. Currently, Table Lens requires using a formula to generate a new column, and would thus require an iterative search of this set of transforms with the implied loads on activity, knowledge, and memory. This iteration can be removed by providing a direct manipulation interface that allows transforming the column in place.

One obvious thought is to put a slider on the column that allowed sliding through rungs of the ladder. An alternative, which perhaps reinforces the immediate feel of data manipulation, is to place "control points" on key locations of curve of sorted values. Then by pulling those control points in the right directions, the variables could be "pulled" up or down the ladder. This second approach is similar to the approach taken in SDM (Chuah, Roth, J. Mattis, & Kolojejchick, 1995) for transforming a selected set of data values, and is largely inspired by the Brown work on direct manipulation handles on graphical objects (Herdon & Meyer, Nov 1994; Zeleznik, Herndon, Robbins, Huang, Meyer, Parker, et al., 1993).

Another aspect of this process is that the analyst is attempting to make the batch confirm to the shape of some canonical distribution e.g. the normal distribution. Thus as the ladder of power is searched, the analyst is comparing the fit of the actual curve to an ideal curve. This process can be better supported by providing a reference curve superimposed on the actual curve. In addition, the total difference could be indicated in an output area. The reference curve itself could be manipulated to select different members of, what we might call, a ladder of distributions.

Variable Permutations.

Finding correlated variables involves sorting Table Lens and scanning all columns for related variables. As the prelude to the variable modeling task, this initiates a process of comparing correlated variables to select one for fitting. In a randomly organized table this may lead to many visual traversals of the table. An operator which permuted the variables so that more correlated variables on each side of the sorted variables were brought nearby would decrease the total amount of visual traversal distance necessary for this part of the task.

The value of ordering correlated variables in a best first order can be explained with a simple Information Foraging model analogously as for ordering operations in Scatter/Gather.(Pirolli & Card, 1995). Another way of seeing the value of variable ordering is that it allows column to be skipped (as in the C=7 case of Figure 5) thus improving performance.

This operation is analogous to the 2 way manual sorting operation of the "permutation matrix" described by Bertin (Bertin, 1983). It provides the general ability to group sets of variables that move as a group.

Fit Marks and Residual Curves

Once a correlated variable has been selected for fitting, the user of Table Lens must go through a cumbersome iterative process again using computed columns to perform re-expressions of both variables, line fitting, residual observation. The mechanisms for direct manipulation re-expression can be directly used here on both variables. The other 2 subtasks of fitting and residual monitoring can be supported in similar ways.

Given 2 variables, A and B, say A has been sorted and B is visibly correlated. When interest is focused on B, this can invoke a background fitting process which annotates the bars of B's column with "fit marks," say saliently colored points placed along each bar where the fitted line would indicate that case should be. As either variable is slid on the ladder of powers, the fit marks can be updated.

With this graphical aid, the user can quickly ascertain the deficit or excess between actual and fitted values of B. Again, an output area can be provided to display a computed measure of total difference. In this case, the

user may also be interested in the shape of the residual, and in fact will ultimately want the residual as a separate column for further iteration using other variables. To support these tasks, the bases of the bars can be superimposed with bars that show the differences (perhaps in 2 colors for excess and deficits). This "residual curve" can be browsed and perhaps directly "dragged" apart into a separate column. Nominal variables, and nominal vs. quantitative variables.

SUMMARY

Table Lens supports quite well the studied EDA task, though there are a number of refinements suggested by the analysis that would increase its effectiveness. Though we haven't completed the analysis of learning costs, a quick analysis of the space of operators makes it clear that only a few operators are necessary to perform the tasks in the single Table Lens display, while a variety of displays are typically used in EDA tools and absolutely required by spreadsheets. The direct manipulation operators, the single familiar organization of the table, and the direct incorporation of graphics quite parsimoniously provide rich support for EDA tasks. We believe that with further improvements to the design, we can make accessible to a broad set of users the basic suite of EDA techniques.

REFERENCES

- Becker, R. A., Chambers, J. M., & Wilks, A. R. (1983). *The New S Language*. University of Wisconsin Press.
- Becker, R. A., & Cleveland, W. S. (1984). *Brushing a Scatterplot Matrix: High-Interaction Graphical Methods for Analyzing Multidimensional Data* (Tech. Rep). AT&T.
- Bertin, J. (1983). *Semiology of Graphics*. University of Wisconsin Press.
- Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Chuah, M. C., Roth, S. F., J. Mattis, & Kolojejchick, J. (1995). SDM: Malleable Information Graphics. In *Information Visualization 95* IEEE Computer Society Press.
- Herdon, K. P., & Meyer, T. (Nov 1994). 3D Widgest for Exploratory Scientific Visualizations. In *Proceedings of the ACM Symposium on User Interface Software and Technology* ACM Press.
- Langley, P., Simon, H. A., Bradshaw, G. L., & Zytkow, J. M. (1987). *Scientific discovery: Computational explorations of the creative processes*. Cambridge, MA: MIT Press.
- Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice Hall.

- Olson, J. R., & Olson, G. M. (1990). The growth of cognitive modeling in human-computer interaction since GOMS. , *Human-Computer Interaction*, 5, 221-265.
- Qin, Y., & Simon, H. A. (1990). Laboratory replication of scientific discovery processes. , *Cognitive Science*, 14, 281-312.
- Pirolli, P. & Card, S. (1995). Information foraging in information access environments. In Proceedings of the *Conference on Human Factors in Computing Systems, CHI-95*. Association for Computing Machinery
- Rao, R., & Card, S. K. (1995). Exploring Large Tables with Table Lens. In *Video Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*
- Rao, R., & Card, S. K. (April 1994). The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*
- Russell, D. M., Stefik, M. J., Pirolli, P., & Card, S. K. (1993). The cost structure of sensemaking. In *INTERCHI '93 Conference on Human Factors in Computing Systems*, (pp. 269-276). Amsterdam: Association for Computing Machinery'.
- Tukey, J. W. (1977). *Exploratory data analysis*. Reading, MA: Addison-Wesley.
- Zelevnik, R. C., Herndon, K. P., Robbins, D. C., Huang, N., Meyer, T., Parker, N., & Hughes, J. F. (1993). An Interactive 3D toolkit for constructing 3d widgets. In *Proceedings of SIGGRAPH'93* (pp. 81-84). ACM Press.