# Chapter 1

# Introduction

This chapter introduces ViSta, the Visual Statistics system. ViSta is designed for students and teachers in introductory and multivariate statistics, for researchers who have data to analyze but are new to statistical data analysis, and for researchers, graduate students and programmers doing research and development in visual and computational statistics. ViSta features state-of-the art visualization techniques to guide novice data analysts, to portray the overall structure of a data analysis session, to communicate the results of analyses, and to re-estimate model parameters. ViSta is available for free and can be freely redistributed.

This introductory chapter focuses on the motivation for developing ViSta, on the goals and principles of its design, and on its data analysis environments and statistical visualization techniques. The chapter also presents the hardware platforms on which ViSta runs, and details how to obtain the software.

## 1.1 Motivation, Goals and Principles

Data are the lifeblood of science. Because computerized data-analysis systems help scientists understand data, they have become of central importance to the scientific enterprise, evolving into extensive and powerful systems capable of performing many kinds of very sophisticated and complex analyses.

Unfortunately, the structure of data-analysis systems has evolved willy-nilly over the years. While much thought has been focused on the kinds of analyses that can be performed by these systems, less thought has been given to their overall structure: It seems that the more powerful a statistical system is, the more clumsy it is.

At the same time that these changes have taken place in data-analysis software, the computational hardware on which these systems run has increased in capability: Processors are faster, memory is larger, displays are bit-mapped instead of character-mapped, and mice are widely use. And all the while, cost has decreased.

Thus, compared to a few years ago, much more complex analyses can now be performed in less time and at less cost. Furthermore, a much broader cross-section of the scientific community has come to have access to data-analysis systems. However, the main difficulty with essentially all data-analysis systems is not the sophistication of the analyses that can be performed. Rather, it is the fact that these systems are not designed with the capabilities of the user in mind. This weakness makes the systems difficult to learn and use, especially for novices.

While much effort and thought have been expended to improve the capability of individual components of many statistical systems, relatively little effort and thought have been given to giving data-analysis systems a structure that would make them accessible to the full range of users, from novices to experts.

Some systems, especially the older systems such as SAS (SAS Institute, 1990) or SPSS (Norusis, 1990) seem to have no overall unifying design and little regard for the capabilities of their users. Other systems, such as S (Becker, Chambers & Wilks, 1988) are designed with the capabilities of the sophisticated user in mind, being, essentially, high-level data-analysis languages.

Recently available systems such as DataDesk (Velleman & Velleman, 1988) and JMP (SAS Institute, 1990) are designed from the ground up on the basis of a unified graphical user interface metaphor. As one might expect, these systems are indeed much more appropriate for the novice user. However, these systems are entirely graphical. They do not incorporate a data-analysis language as part of their design, and so lack the flexibility and customizability that the sophisticated user needs.

The Lisp-Stat system (Tierney, 1990) seems to have the potential to be an appropriate vehicle for implementing interfaces for the novice and the expert, since it is

designed with an object-oriented and language-based philosophy, and since it is extensible. However, Lisp-Stat lacks a graphical user interface.

Most statistical systems have not been designed to be easy to use by a wide range of users. Even with simple data analyses novice users are soon at a loss as to how to combine several data-analysis steps into a cogent statistical strategy that reveals the basic information in the data. The very power of many systems can actually hinder the data-analysis task, especially for users who are novices. We have the paradoxical situation that for many users, the increasingly powerful and sophisticated data-analysis systems are actually less suited to most users for understanding data.

ViSta is designed with the capabilities of the user in mind. The basic principal guiding the design of ViSta is that a data analysis environment should fit the sophistication of the user's data analysis knowledge. The main goal of ViSta's software design is to maximize the data analyst's productivity and satisfaction by providing the analyst with a choice of suitable data analysis environments. In this way, ViSta is designed for an audience of users having a very wide range of data analysis sophistication, ranging from novice to expert.

ViSta provides seamlessly integrated data analysis environments specifically tailored to the user's level of expertise. Visual guidance is available for students and novices, and visual tools are available for experts to create guidance for these novices. A structured graphical user interface is available for competent users, and a command line interface is available for sophisticated users. Scripts can be written which require no user interaction. Finally, the complete Lisp-Stat (Tierney, 1990) programming environment is available to researchers, graduate students and programmers who wish to extend ViSta's capabilities.

ViSta's design understands that visualization techniques are not useful for everyone all of the time, regardless of their sophistication. Thus, all visualization techniques are optional, and can be dispensed with or reinstated at any time. In addition, standard non-visual data analysis methods are available, including printed reports, a command-line interface and support for scripts for automated analysis. This combination means that ViSta provides a visual environment for data analysis without sacrificing the strengths of standard statistical system features that have proven useful over the years. ViSta's design recognizes that it may be true that a single picture is worth a thousand numbers, but that this is not true for everyone all the time. And, in any case, most of the time most of us find that pictures *and* numbers give the most complete understanding of data.

## 1.2 ViSta's Data Analysis Environments

One of ViSta's main design principles is that a data analysis environment should reflect the sophistication of the user's data analysis knowledge. Since data analysts vary in their knowledge, the data analysis system should provide a variety of environments, each designed for a different level of data analysis knowledge. In this way, we believe, the data analyst's productivity and satisfaction will be maximized.

ViSta is designed to accommodate the complete range of sophistication on the part of data analysts, from novice to expert. Since the data analysis environment which does this for a novice user is different from the one which does this for a sophisticated analyst, the design has several environments, including:

1. **Guidemaps** to guide novice data analysts through complete data analyses.
2. **Workmaps** to let data analysts create data analyses visually, and show data analysts the overall structure of their data analysis sessions.
3. **Command Line Interface** to let sophisticated data analysts dispense with the visual aids when they find them unnecessary.
4. **Guidance Tools** to let expert data analysts create guidemaps.
5. **Scripts** for non-interactive or repetitive data analyses.
6. **Lisp** (with access to **C** and **FORTRAN**) so programmers can extend ViSta.

These environments are seamlessly integrated within ViSta. Analysts can switch between them whenever desired. We discuss each of these environments in this section, along with the way in which they are seamlessly integrated.

### 1.2.1 Guidemaps for Novice Analysts

One of ViSta's main premises is that users with little or no knowledge about data analysis can benefit from an environment which *visually guides* their analysis. To this end, the sequence of steps which expert data analysts think should be taken can be presented visually as a map like the one shown in Figure 1. This "guidemap" guides those with less expertise through the series of steps in a complete statistical data analysis.

Figure 1is a guidemap for exploring data. The steps are indicated by buttons — highlighted (dark) buttons are suggested steps, whereas gray buttons are data analysis steps that are not currently suggested. The sequence in which the steps are suggested is indicated by the arrows pointing from one button to the next. The structure of the guidemap doesn't change as the analysis proceeds, although the highlighting does.

The user makes choices by pointing and clicking with a mouse on the !! side of a highlighted button. Help about the step may be obtained by clicking on the ?? side.

After a suggested step is taken the selection of active buttons changes to show the user which actions can be taken next.

In the guidemap shown in Figure 1, the button highlighting indicates that the analyst has the choice of two actions: browse the data or return (to the previous guidemap). When the user chooses one of these actions, the action takes place and the chosen button turns gray, since it is no longer a recommended action. If the user chooses to browse the data, the datasheet appears. After it is closed the highlighting of the buttons in the guidemap changes: The "Browse Data" button becomes gray and the "Visualize Data" and "Summarize Data" buttons become highlighted. In this way ViSta guides the user, saying that the next steps should be to either visualize or summarize the data. These two buttons have to be used before the next two buttons ("List Observations" and "List Variables") are activated. In this way the user is guided through the steps for exploring data. (For information on the theoretical underpinnings of guidemaps see Young & Lubinsky, 1995)

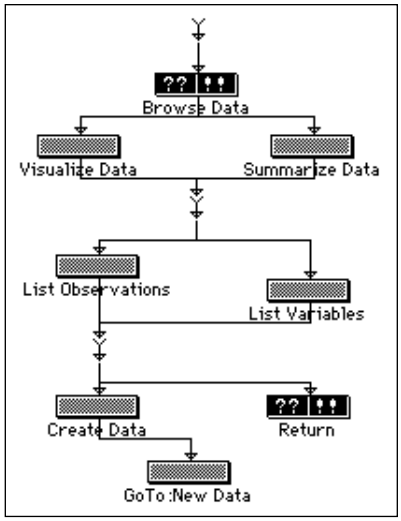



**Figure 1:**
**The GuideMap for Exploring Data**

### 1.2.2 Workmaps for Novice and Competent Analysts

Another of ViSta's main premises is that users at nearly any level of data analysis sophistication can benefit from an environment that *visually structures* their analysis. ViSta's *WorkMaps* are visual diagrams of the steps taken in a data analysis session. An example is shown in Figure 2.Unlike a guidemap, whose structure doesn't change, a workmap is created and expands as the steps of the analysis take place. The analyst uses the workmap's menus to carry out a data analysis. As the analysis proceeds the workmap is automatically created. It serves as a history of the analysis, and can be used to return to previous steps. The workmap concept was introduced by Young & Smith (1991).

The workmap shown in Figure 2 summarizes the steps of an on-going analysis. We see that the analyst began with the "CarRatings" data. These data were normalized, creating new data named "Norm-CarRatings". The analyst then loaded data named "Car-Prefs", creating a third data icon. These data were analyzed by the procedure for principal components analysis, producing an analysis procedure icon named "PrnCmp", and a model icon named "PCA-Car-Prefs". The analyst then requested that the model create data objects of scores and coefficients, then merged the normalized ratings with the scores to form the "rats-scores" data. This data was used for a multivariate regression analysis, which in turn was used to create output data from the regression. Any of the data or model icons in this diagram can be opened to show a visualization or a report.

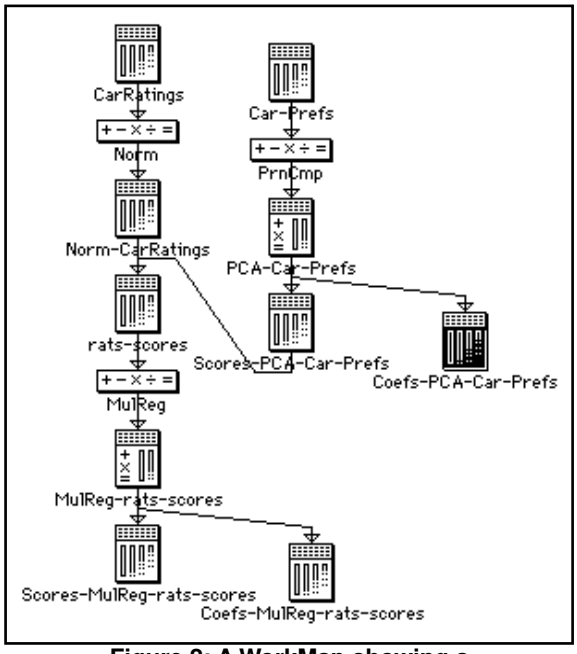



**Figure 2: A WorkMap showing a Series of Data Analysis Steps**

### 1.2.3 Command Lines for Sophisticated Users

ViSta includes a command line interface for sophisticated data analysts. The commands are typed at the keyboard. The analysis takes place as each command is typed, and the workmap is created as the analysis proceeds. The example in Figure 3 is the command-line equivalent of the workmap-created analysis shown in Figure 2. Here, data named "carratings" are loaded and then normalized, with a report (listing) being obtained. A new set of data, named "car-prefs" is then loaded into ViSta. It is submitted to a principal components analysis, after which a report and visualization of the

```
(load-data "carratings")
(normalize-data)
(report-data)
(load-data "car-prefs")
(principal-components)
(report-model)
(visualize-model)
(create-data)
(setcd scores-pca-car-prefs)
(setcd norm-carratings)
(merge-variables)
(multivariate-regression)
(create-data)
```

**Figure 3: Command Lines**

component model are obtained. Then, output data objects are created from the principal components analysis, and the scores and ratings are merged. Finally, a multivariate regression is performed using the merged data, with the results being output into data objects. This reproduces the data analysis whose workmap is shown in Figure 2, and includes report and visualization steps not recorded on the workmap.

### 1.2.4 Guidance Tools for Expert Users

ViSta provides graphical tools so that experts can create the guidemaps that are used by novices. The theory underlying these tools is described by Young & Lubinsky (1995). The way the tools are used is explained in a later chapter.

### 1.2.5 Scripts for Automated Analysis in Repetitive Situations

The four environments discussed above are all *highly interactive.* This means that as soon as an icon is clicked, or a command is typed, ViSta responds. This is desirable in many situations, especially when analyses are being performed on a one-shot or exploratory basis. However, in other situations, such as when an analysis will be repeated again in the future on a new wave of data, it is better to be able to collect all commands together into a file and run them all at once without user interaction. When the commands shown in Figure 3 are saved in a file, they become a script which can be loaded into ViSta to create an analysis that requires no user interaction. Note that the script creates the workmap shown in Figure 2, if desired.

### 1.2.6 The Lisp, C and FORTRAN languages

In addition to the five data analysis environments just outlined, ViSta includes a complete object-oriented programming environment for programmers who wish to extend or customize ViSta's capabilities. The programming environment is Lisp, as implemented in the XLisp-Plus (Almy, 1993) and XLisp-Stat (Tierney, 1990) systems. This environment also gives access to programs written in C and FORTRAN. Lisp-Stat has been extensively reviewed by a number of authors (Baxter & Cameron, 1991; Lubinsky, 1991; Weihs, 1991; Young, 1991), and is the basis of at least one other extensive development project (Cook & Weisberg, 1994). Lisp-Stat includes object, windowing and graphical systems and is an open system, providing access to all of its code.

Note that data analysts never have to deal with Lisp while using ViSta. It is only needed by programmers who wish to extend or enhance ViSta's capabilities.

### 1.2.7 Seamless Integration of All Environments

The five data analysis environments and the programming environment are all seamlessly integrated. Guidemap buttons correspond to menu items, and generate commands that are identical to those typed at the command line. Workmap menu

items also generate the same commands. In fact, the titles of the guidemap buttons and the names of the workmap menu items are both identical to the commands that can be typed. These commands, in turn, generate the structured analysis diagram and perform the data analysis. Scripts, as we have shown, contain the same commands. Finally, the tools for creating guidemaps are based on the same underlying commands. Thus, all data environments are seamlessly integrated via the underlying data analysis commands.

Because of this seamless integration, it is possible to switch between the several kinds of environments at any time. When the analyst moves into an unfamiliar type of data analysis or looses track of the overall structure of the analysis, s/he can switch from the command line interface to the workmap's menu-based interface, with the entire structured history of the analysis session being presented. Similarly, guidance diagrams can be switched on or off as desired, without loss of continuity. Also, once a script-based analysis has been completed, the analysis can continue interactively in any of the above ways. In addition, the programming environment permits the programmer to write new features and then test them by switching to any of the data analysis environments.

## 1.3 ViSta's Statistical Visualization Methods

One of the main design principals of ViSta's statistical visualization methods is that a single picture is worth a thousand numbers. Statistical visualization uses geometrically based statistical models to provide visual insight into the structure of data or data models. Consider the principal components model (Jackson, 1991). It can be viewed as a geometric model that represents observations as points in a high-dimensional space whose dimensions correspond to the variables. The statistical visualization of the principal components model presents the results of the analysis as a group of interacting plots, the purpose being to intuitively communicate the results of the analysis through pictures. When this visualization is combined with traditional reporting techniques (i.e., tables), the user gains a greater understanding of the results than when either technique is used alone. That is, most of the time most of us find a picture *and* a table to be worth more than either by itself.

The following three kinds of statistical visualization tools are available in ViSta:

1. **Linked Plots:** This set of statistical visualization tools is used to present data structure and to present the results of statistical analyses.
2. **Spreadplots:** These statistical visualization tools are used to explore the structure of high-dimensional data and of models of such data.
3. **Statistical Re-Vision:** This set of statistical visualization tools is used to help search for meaningful and parsimonious model parameterizations.

### 1.3.1 Empirically Linked plots - Groups of Interacting Plots

One of the primary statistical visualization tools in ViSta is the linking of several plots through their data's observations and variables, as discussed by Stuetzle (1987). We call these "empirically" linked plots since they are linked via the data.

Figure 4 presents an example of a layout of the five empirically linked plots that form the visualization of table data, a kind of data which has several categorical variables and a single continuous variable. These data concern how far Foxes and Coyotes were observed to wander during the seasons of the year. The upper-left plot is a box, diamond and dot plot that is showing connected box-plots of the distribution of wandering distance for each season. We see the most wandering during Spring, and least during Winter. Below this plot are a quantile plot for Fall and a quantile-quantile plot for Fall versus Summer. The curved line in these plots suggests that the data may not be normally distributed.

The two windows on the right show the data's categorizing sources (upper window) and the levels of the selected source (lower). These two windows control what is shown in the three plots on the left. Clicking on an entry in the upper-right window changes the categorization of the data shown in the box and diamond plot, as well as the information shown in all three lower windows. Clicking on an entry in the lower-right window changes the information in the quantile plot, and clicking on two entries changes the quantile-quantile plot. Thus, these five windows are all empirically linked via the category structure of the observed data.

Figure 5 presents an example of a layout of the four empirically linked plots that form the visualization of multivariate data, a kind of data which has several continuous variables. The data shown in the figure concern certain mechanical aspects of automobiles. At the upper-left is a scatterplot-matrix. To its right is a spin-plot. At the bottom-left is a scatterplot, and to its right is a histogram. These plots are empirically linked via their observations: The labeled points in the scatterplot are highlighted in the spin-plot and histogram (and could be in the plot-matrix). As one selects points in any plot (by clicking or dragging the mouse), points for the same observations can be highlighted in any other plot. Being able to see where observations appear in several plots lets the analyst get a better idea of the data's structure.

These four plots are also empirically linked via their variables: By clicking on a cell of the scatterplot-matrix (which has scatterplots of all pairs of variables) the user can choose which variables are plotted in the other plots. The scatterplot and histogram are showing variables which correspond to the cell in the scatterplot-matrix which has the "finger" cursor located on it. These are also two of the spin-plot's three variables. Clicks and shift-clicks on cells in the scatterplot-matrix determine which variables appear in the other plots. Being able to display various combinations of variables in the other plots lets the analyst look at many views of the data's structure.

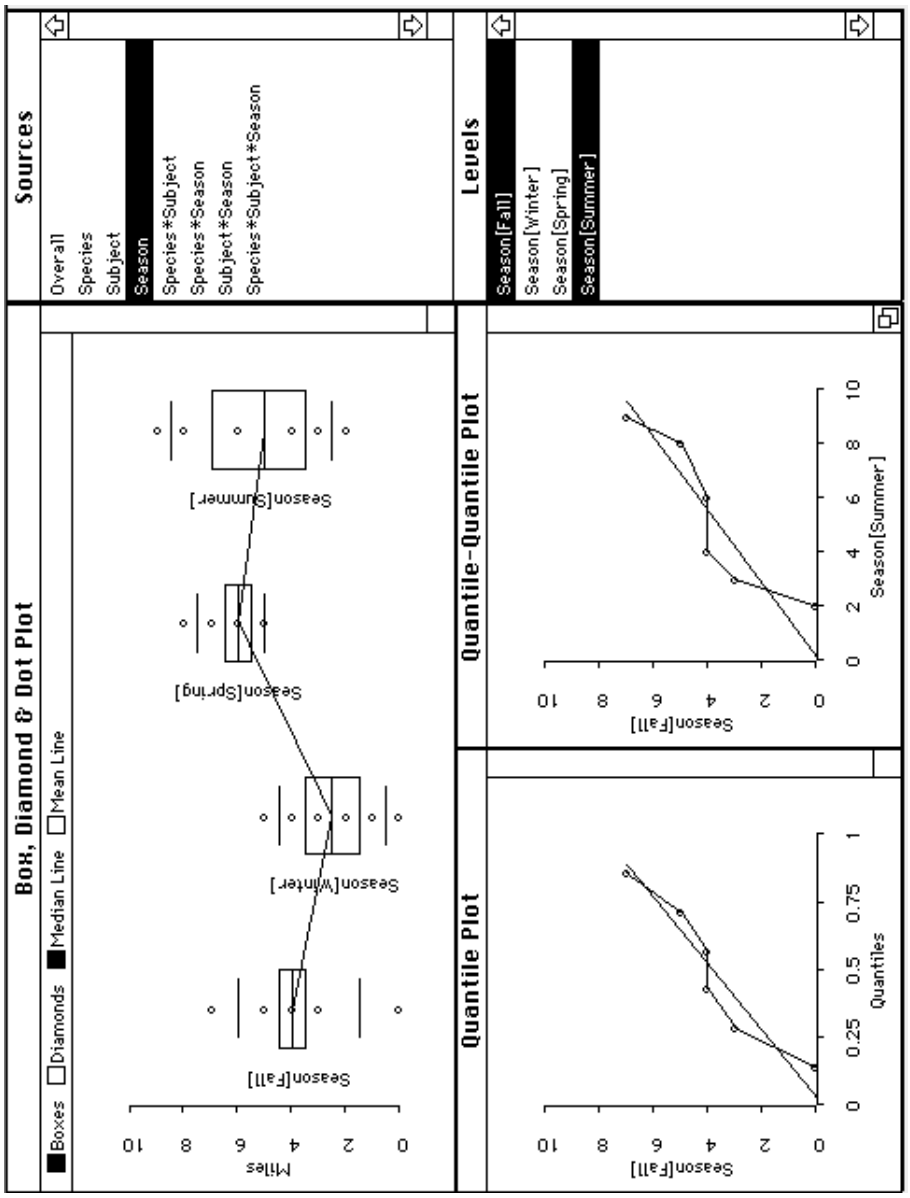



**Figure 4: Empirically Linked Plots for the Table Data Visualization**

### 1.3.2 Spreadplots - Algebraically Linked Plots

ViSta includes spreadplots (Young, Faldowski & Harris, 1991) among its statistical visualization techniques. An example is shown in Figure 6. A spreadplot is the graphical equivalent of a spreadsheet: It is a group of several interacting dynamic plots, with the several plots being *algebraically linked by equations*. Note that algebraic linkage is fundamentally different from empirical linkage. Empirical linkage involves the data's observations and variables. Algebraic linkage involves a model's equations. ViSta's spreadplots can have both kinds of linkages between plots in the same spreadplot.

Figure 6 is titled "A Guided Tour Spreadplot" because there are algebraic links between the plots. There are two kinds of algebraic links. First, there are equations which link the two target plots with the tour-plot. These equations create the specifics of the high-dimensional spinning that occurs in the tour-plot. The user actually

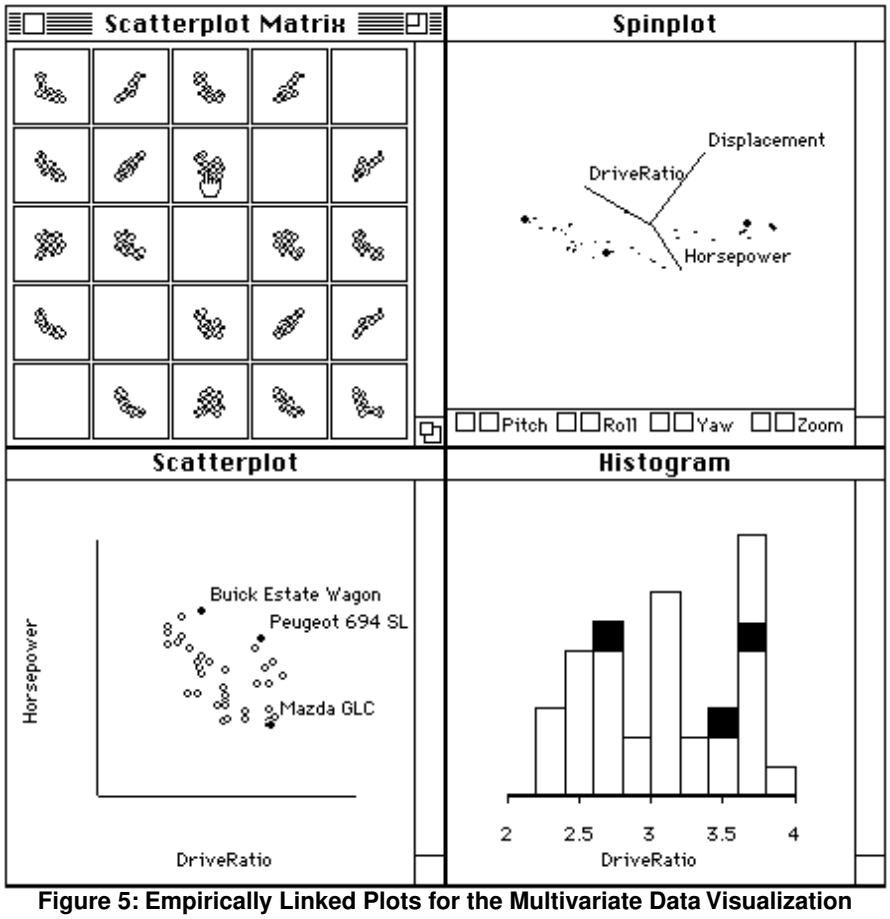
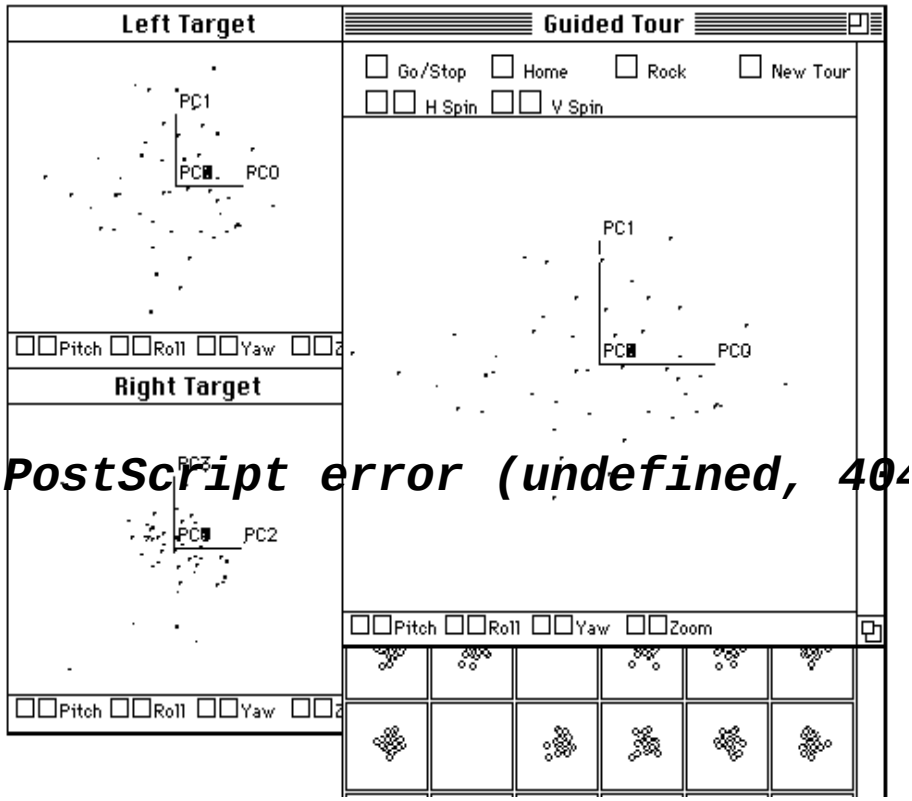



**Figure 5: Empirically Linked Plots for the Multivariate Data Visualization**

has the choice of two sets of equations for two different types of tours. One set of equations (Buja & Asimov, 1986) implements the high-dimensional rotation model mentioned above. The other set (Young, Kent & Kuhfeld, 1988) implements a high-dimensional linear interpolation model. Both are discussed by Young & Rheingans (1991).

The second type of algebraic link between the group of plots implements the residualization model proposed by Young, Kent & Kuhfeld (1988). When the "New Tour" button is clicked the specific position of the tour-plot in its spin between the two targets is used, along with the residualization equations, to update the two target windows. These new targets are then used, via the high-dimensional spinning equations that link the windows, to modify the path taken by the tour-plot during its high-dimensional spin.