Multivariate Statistical Visualization

Forrest W. Young,
Richard A. Faldowski &
Mary M. McFarlane

Psychometrics Laboratory
University of North Carolina at Chapel Hill

 Published in Rao, C.R. (Ed.) Computational Statistics. Handbook of Statistics, Vol **9**. pp 959-998. Elsevier Science, Amsterdam. 1993.

For more information contact the first author at: UNC Psychometrics, CB 3270 Davie Hall, Chapel Hill NC 27599-3270. 919-962-5038. forrest@unc.edu. http://forrest.psych.unc.edu/

Multivariate Statistical Visualization

Abstract: In this paper we describe multivariate statistical visualization techniques designed to improve the quality, accuracy and satisfaction of the statistical data analysis process. We describe techniques for visualizing multivariate data structure, for visualizing multivariate data models, and for visualizing multivariate data analysis sessions. We illustrate these techniques with ViSta, our statistical visualization research and development testbed.

1 Introduction

Statistical data analysis systems have long included graphics to help users see the results of analyses. Such statistical graphics have also been used to help users explore data for structure. Dynamic statistical graphics -- graphics which incorporate motion -- can be powerful tools for exploring data structure. They can be powerful because they help the scientific explorer visually analyze -- to visualize -- structure.

Dynamic statistical graphics are especially powerful for visualizing structure in multivariate data. This is because multivariate observations can be abstractly represented as points in a space which has a dimension for every variable, and because dynamic statistical graphics have been designed to visualize structure of high-dimensional space. Since the early stages of scientific inquiry involve exploration, and since scientific exploration leads to scientific hypotheses, dynamic statistical graphics can be central to the process of gaining scientific insight about multivariate data.

For many years, statisticians focused on developing and improving inferential methods designed to test hypotheses, to the neglect of exploratory methods designed to form hypotheses. In recent years, there have been many new developments in exploratory data analysis methods. Tukey, in his landmark book, Exploratory Data Analysis (1977, p. V), states that exploratory data analysis

> "is about looking at data to see what it seems to say. It concentrates on simple arithmetic and easy-to-draw pictures. It regards whatever appearances we have recognized as partial descriptions, and tries to look beneath them for new insights. Its concern is with appearance, not with confirmation."

The work reported here falls under the general rubric of "exploratory data analysis," and is guided by the philosophy of "looking at data to see what it seems to say." We focus on dynamic statistical graphics methods for exploring multivariate data, methods which we call *multivariate statistical visualization* methods.

Multivariate statistical visualization methods capitalize on the pattern recognition power of human vision and on the computational power of graphics workstations to help data analysts look for structure (form hypotheses) that may be in their multivariate data. The goal of multivariate statistical visualization is to aid in forming hypotheses about the data's high-dimensional (hD) geometric structure, even though we can only see in 3D. To do this, the visualization must: 1) respect the data's high-dimensional geometry; 2) respect the user's three-dimensional perception; and; 3) respect the workstation's two-dimensional screen and its other computational limits. Thus, the problem that all multivariate statistical visualization methods must tackle is how to present hD information in a 2D plane, such that our 3D perception can understand the hD geometry.

1.1 Historical Background

Presenting 3D in a 2D plane is not new, of course. Artists have done this for centuries, statisticians for over a century, and computers for nearly three decades. Indeed, even techniques for presenting hD in a 2D plane are not new. Tufte (1983, p. 40) presents a marvelous example, dating from 1861, of a 2D statistical graphic that incorporates six dimensions of information concerning the fate of Napoleon's army in Russia.

There are sophisticated computer techniques that create images that appear to genuinely occupy 3D volume. These techniques do a very convincing job of tricking us into "seeing 3D." Even ordinary computer-generated 2D printer plots can have additional "dimensions" added by labeling the points in the 3D space. Stuetzle (1987) has developed a system for manipulating several 2D "plot windows" that provide multiple views of hD data, helping to provide understanding of multivariate structure. Also, with common computer-generated color graphics, more dimensions may be added by using various colors and shapes to distinguish the points on discrete dimensions. In addition, there are a variety of techniques for communicating 3D and hD on a 2D plane, including perspective and stereo projections, movement, dynamically-changing object shapes, etc. Many of these techniques are discussed in this paper.

Multivariate statistical visualization methods can be classified as passive or active. Active methods require the user to interact with the computer to create movement, whereas passive method only require the user to passively watch movement that the computer creates. We briefly mention some examples of each kind of multivariate graphics in the next few paragraphs.

Fisherkeller, Friedman and Tukey (1974) developed an active multivariate statistical visualization system that constructed a spinplot -- a 3D scatterplot that could be rotated by the user. The sophistication of this early approach has been greatly increased in later descendants (Donoho, et al., 1982; Friedman, McDonald and Stuetzle, 1982; and Donoho, Donoho and Gasko, 1986; Gabriel and Odoroff 1986), and has been incorporated recently in many commercial statistical systems. Huber, an early developer of this approach, discusses his experiences in a later paper (Huber, 1987). With these systems, the user rotates the spinplot with a mouse or cursor keys. The user-controlled rotations create a reasonable sensation of depth. Spinplots are discussed later in this paper.

Additional active multivariate statistical visualization methods have been developed to assist the data analyst in understanding the structure of data. Many of these methods

involve forming subsets according to locations or properties of the displayed objects or according to values of variables in the data set. Donoho, Donoho and Gasko (86) implement "slicing" and "masking," methods for dynamically subsetting observations on a fourth dimension to determine where these observations are in the 3D scatterplot. Becker and Cleveland (1986) and Young Kent & Kuhfeld (1988) discuss "brushing," a way of dynamically subsetting observations simultaneously on two variables, which helps the user determine where the observations fall on yet other variables. Many developers implement other subsetting methods that group observations according to variable values.

Subsetting is particularly powerful when combined with a tool called "metamorphing" by Young Kent & Kuhfeld (1988). This tool allows the user to change the appearance of objects on the screen that represent subsets of observations. Various researchers have developed systems which allow objects to have attributes such as color, shape, labels, size, etc., and then allow these attributes to be metamorphed for subsets of observations, thereby helping to easily identify the different subsets.

Tukey & Tukey (1980) proposed what is now known as the scatterplot matrix, a graphical matrix where each “element” is a 2D scatterplot formed by plotting the row variable against the column variable. Recent developments of this approach are presented by Carr, Littlefield, Nicholson and Littlefield (1987) and are available in many commercial systems. When brushing and metamorphing are performed on scatterplot matrices the result is a very useful active multivariate statistical visualization method.

Passive multivariate statistical visualization methods have been developed to search through the hD data space and to display smoothly changing, dynamic projections of this search. Friedman and Tukey (1974) developed a passive method they call Projection Pursuit for looking through the hD data space to find "interesting" 2D views. This approach is discussed extensively by Huber (1985). Nicholson and Carr (1985) have developed another passive method that presents rocking views of objects located in 3D space whose dynamically changing shape represents the changing values of two additional variables. With this system, the rocking takes place on all five dimensions simultaneously.

Asimov (1985) and Buja & Asimov (1986) have proposed the Grand Tour, a passive multivariate statistical visualization method designed to reveal structure in hD space. This method displays smoothly-changing 2D projections of hD data space, with the change in projection being controlled by computer algorithms. Closely related to the Grand Tour is the Guided Tour, an active multivariate statistical visualization method developed by Young Kent & Kuhfeld (1988), Hurley & Buja (1990), and Young & Rheingans (1991a). A Guided Tour is designed for visualizing structure in high-dimensional data that uses smoothly-changing 2D or 3D projections of hD data space, with the changes in projection being guided by the user through high-interaction, immediate feedback point-and-click actions. These methods will be discussed in Section 2.3.

Recently, a new active multivariate statistical visualization method has been introduced by Young and his co-workers (Young, Faldowski & Harris, 1992; Faldowski, 1992; McFarlane, 1992). The method, which is called a “spreadplot”, is a graphical analogue of a spreadsheet: it is a set of dynamic plots that are algebraically linked together via a set of equations. When the user makes changes in one plot, the other plots change

according to the equations linking the plots. Spreadplots are particularly appropriate for viewing a multivariate model's views of structure in multivariate data. The plots present several model views of the data, while the equations that link these plots are based on the model's itself. These developments will be discussed in Section 2.4.

Spreadplots support a new class of graphical tools which perform interactive graphical modeling. With these tools users modify the model's parameter estimates by manipulating elements of the graph -- i.e., they move points and vectors. When estimates are changed, these new estimates are used by the equations linking the plots to update all plots so that they show the new view of the data provided by the revised parameter estimates. This process allows users to not only visually explore their data, but also to visually explore various models of their data, a powerful new way to "see what the data seem to say".

1.2 ViSta: A Visual Statistics Testbed

In this paper we discuss nearly all of the techniques discussed in the previous section. The discussion uses ViSta (Young, 1992), a visual statistics research and development testbed, to illustrate many of the techniques. ViSta is written in Lisp, using the Lisp-Stat environment (Tierney, 1991). It has been developed on Apple Macintosh microcomputers. It should also run (with minor modifications) on Unix workstations under X-Windows, and on IBM-compatible microcomputers under Microsoft Windows. You can contact the authors for further information about the availability of ViSta.

ViSta is not a complete statistical system. Rather, it is a testbed for research and development in statistical visualization techniques. As such, it supports the visualization techniques reported in this paper, but does not include many more common features that appear in typical statistical systems. ViSta is designed for an audience of users having a very wide range of data analysis sophistication, ranging from novices to experts. ViSta provides data analysis environments specifically tailored to the user's level of expertise. Guidance is available for novices, and tools are available for experts to create guidance for novices. A structured graphical user interface is available for competent users, and a command line interface is available for sophisticated users.

ViSta's design takes into account 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. ViSta combines its novel visualization techniques with standard statistical system features that have proven useful over the years. This combination means that ViSta provides a visual environment for doing data analysis without sacrificing the strengths of command lines, batch processing, and textual reports. ViSta's design rests on the assumption that combining traditional approaches with cutting edge visualization techniques gives the user the most complete understanding of the data.

1.3 Organization of this Paper

In this paper we organize multivariate statistical visualization techniques into three areas, namely those for visualizing multivariate data; those for visualizing multivariate models of data; and those for visualizing entire sessions of multivariate data analyses.

We briefly define each of these in this section, and then go into each in depth in the remainder of the paper.

Visualizing Multivariate Data Structure: Multivariate statistical visualization techniques that are designed to help visualize data structure include scatterplot matrices, spinplots and tourplots, and dynamic versions of such familiar graphics as scatterplots, histograms, box-plots, etc. Visualizations involve the display of these graphics in multiple windows that are linked by their observations and/or their variables. When an observation is highlighted, metamorphed or labeled in one window, it can be highlighted, metamorphed or labeled in other windows. These graphics are designed to help the analyst visually explore spaces representing the data's structure. This topic most closely parallels the work on plot-windows presented by Stuetzle (1987).

Visualizing Multivariate Models: Multivariate statistical visualization techniques that are designed to help visualize models of multivariate data include all of the techniques discussed above, as well as spreadplots involving those techniques. Spreadplots support interactive graphical modeling techniques that are designed to help the analyst visually explore the effects of revising a model's parameter estimates. The parameter estimates are represented by graphical elements such as points or vectors in the multi-window spreadplot.These estimates can be revised with graphical tools for moving the points or vectors. Once new estimates are obtained, the implications of the new estimates are rapidly displayed as changes in the model, its fit, and its residuals. While the new estimates may not be mathematically optimal, they may be more meaningful, leading to greater insight about the data than the optimal estimates. This topic most closely parallels the work on spreadplots presented by Young, Faldowski. & Harris (1992), Faldowski (1992) and McFarlane (1992).

Visualizing Multivariate Analyses: Techniques for visualizing multivariate analyses fall into two groups. First, there are techniques designed to help visualize the overall structure of an on-going multivariate data analysis session. The visualization is a "workmap"**.** As the session progresses, the workmap grows, showing each step taken. Thus, when data are input, an icon appears that represents the data. When an analysis method is applied to data, a "method" icon appears that is connected to the data icon, representing the fact that the data are being processed by the method. When the analysis takes place, a "model" icon appears connected to the method icon. This shows that the data have flowed through the method to yield a model. This workmap can be used to remind the analyst about the analysis that has taken place, and can be used to return to previous analysis steps. This topic most closely parallels the work on a structured interface for data analysis presented by Young & Smith (1991).

The second multivariate analysis visualization technique is designed to guide an on-going multivariate data analysis session. The visualization is a "guidemap", and is based on the assumption that users with 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 are presented visually as a cyclic graph. The graph guides those with less expertise through the series of steps in a complete statistical data analysis. This topic most closely parallels the work on strategies for guiding statistical novices presented by Lubinsky, Young & Frigge (1990).

We discuss each of these topics in the remainder of this paper.

2 Visualizing Multivariate Data

In this section we discuss the theory underlying the visualization of multivariate data. We then present two major techniques based on this theory, each accompanied by an example.

2.1 Geometrical Representation of Multivariate Data

The multivariate data visualization techniques discussed in this paper are based on a geometric model of data that is commonly used in statistics. This model views the cases (observations) as points in a high-dimensional space which has a dimension for each variable. This geometric model is used to construct plots which are low-dimensional views of the data. Since the model is at the very heart of our approach to multivariate statistical visualization, we need to precisely define the model before proceeding further.

We begin by supposing that multivariate data consist of h numerical variables observed on each of n cases or observations. We further suppose that these data are collected together into a matrix X, an $(n \times h)$ matrix of data with elements x_{ia}. This matrix has n rows, one for each of the n cases, and h columns, one for each of the h variables. Without loss of generality, we assume that X is "column centered": i.e., that the mean of each column is zero.

Now we can introduce the geometric model of data, which is called a *data space*. A data space is an abstract view of data. In the data space each case of the data is represented by an h-dimensional observation vector x_i whose a'th element is the observation on variable a. Thus, abstractly, the entire set of data is represented by n points in an h-dimensional data space. We denote the data space as $\mathbb{R}^h$, an h-dimensional space of real numbers. The rows of the data matrix contain coordinates of the points in this space, the columns are the dimensions of the space. Since we have assumed that X is column centered, the centroid of the space is at the origin.

The data space is the abstract foundation on which all of our visualizations are built: Every graphic provides a low-dimensional view of the data space, the data space being orthogonally projected onto the low-dimensional graphic. For example, we can think of a scatterplot as a plane which is located somewhere in the data space, with the observation-points (and sometimes the variable-dimensions) being orthogonally projected onto the plane for our inspection. A dot-plot is similar, except it represents the result of orthogonal projection onto a line in the space. Moving plots, such as tourplots, can be understood to be a three-space that is touring through the data space, with the points (and dimensions) in the data space being orthogonally projected onto the three-space, the projections being constantly updated as the 3D-space tours throughout the high-dimensional data space.

2.2 Plot-Windows: Empirically Linked Plots

Since all of the graphics we deal with are views of the data space, they all represent different views of the same thing. Thus, they all have the same set of points, and these points all represent the same observations. Therefore, actions that take place in one plot-window (such as labeling a point, or metamorphing the point's symbol) can be mapped

straight-forwardly onto each of the other plot-windows, since there is a one-to-one correspondence between points in one plot-window and points in another. We say that the plot-windows can be "linked" via their observations. This is one of the central ideas discussed by Stuetzle (1987).

Optionally, the graphics can also present a set of vectors which represent projections of the data space's dimensions, the dimensions corresponding to the data's variables. Just as there is a one-to-one correspondence of observation-points between various plot-windows, so is there a one-to-one correspondence of variable-vectors between various plot-windows. Thus, plot-windows can also be linked via their variables.

When plot-windows are empirically linked together via their observations or variables they jointly act as a single visualization of the data space. We call these "empirically" linked plots because they are linked through the data (and because we need to distinguish this type of linkage from the algebraic linkage used in the spreadplots described in Section 2.4). There are many types of basic plot-windows which can be linked together, including histograms, dot-plots, box-plots, scatterplots, spinplots, scatterplot-matrices, and tourplots. Dot-plots, box-plots and histograms are one-dimensional plots which display a single variable x_a of the multivariate data; scatterplots are 2D plots which display pairs of columns of X; spinplots are 3D plots which display triples of columns of X, and tourplots and scatterplot-matrices are hD plots that display many columns of X. (We explain the hD plots below. We assume you are familiar with the others. If not, see Cleveland & McGill, 1988)

Once the plot-windows are linked together, tools for brushing and selecting points can be used to select observations in one of the plots, while tools for metamorphing and labeling points can be used to identify the selected or brushed observations in all plots. Furthermore, the scatterplot matrix can be used as a control panel to determine which variables are shown in other linked plots. (We assume you are familiar with these techniques. If not, see Cleveland & McGill, 1988).

Figure 1 presents an example of a visualization of data concerning automobiles. The automobiles are rated on six variables by a consumers magazine. Five of these variables have been selected for visualization. The visualization includes four plot-windows and two windows presenting car names and variable names. At the upper-left is a scatterplot-matrix, a matrix whose cells are scatterplots formed by the row and column variables. To its right is a spinplot. At the bottom-left is a scatterplot, and to its right is a histogram. The additional two windows present lists of variable and observation names.

The four plot-windows and the OBS window are empirically linked via the data's observations: The observations highlighted in the OBS window are also highlighted in the scatterplot, spinplot and histogram (and could be in the plot-matrix). Note that these are large American cars, and that the spinplot has been rotated to show these are separated from the other, smaller automobiles. From the scatterplot we see that these automobiles have high weight and displacement. The spinplot also shows that these tend to have high horsepower. When one selects or brushes points in any plot (by clicking or dragging the mouse), points for the same observations are highlighted in the other plots. Points that are selected can be metamorphed: Note that this has been done, as some points are represented by crosses and some by disks. Being able to see where observations appear in several plots lets the analyst get a better idea of the data's structure.

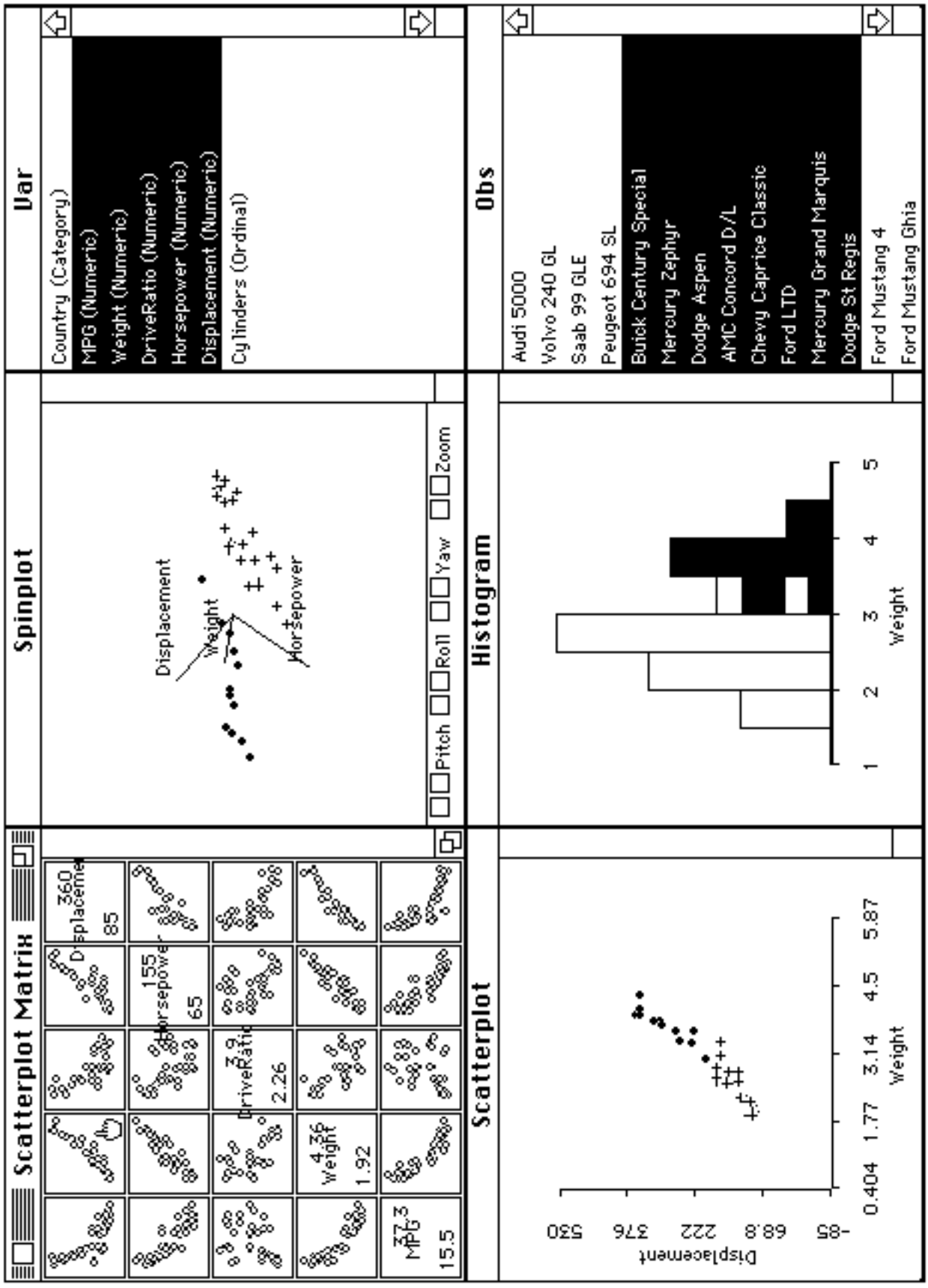

Figure 1: Empirically Linked Plot-Windows

The four plots are also empirically linked via their variables: By clicking on a cell of the scatterplot-matrix the user can choose which variables are plotted in the other plots. In the figure the scatterplot-matrix has been clicked on where the “finger” cursor is located. As a result, the scatterplot and histogram are showing variables which correspond to that cell, and the spinplot also uses these two 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. In this example, using the scatterplot-matrix to select the three variables shown in the spinplot reveals that these three variables (Weight, Displacement and Horsepower) are all fairly strongly related in a linear fashion, with the cars dividing into two groups.

2.3 Tourplots: High-Dimensional Spinplots

A tourplot is a spinplot that spins in more than 3 dimensions (Asimov, 1985; Buja & Asimov, 1986, Young, Kent & Kuhfeld, 1988; Young & Rheingans, 1991a). A guided tourplot spins as directed by the user, the user creating a guided tour of the data. An unguided tourplot spins as it wishes, taking the viewer on a “grand” tour of the data. Just as a spinplot is designed to help the user visualize structure in three-dimensional data, a tourplot is designed to help the user visualize structure in high-dimensional data.

The most important aspect of the guided tour is what we call *the visible space*: a 3D picture of the data formed by orthogonally projecting the data space $\mathbb{R}^h$ onto $\mathbb{R}^3$ and then displaying the projected data in a spinplot. Let us denote the canonical basis vectors of the data space $\mathbb{R}^h$ by e_a, a=1, ... h. They are in one-to-one correspondence with the observed variables. The projection is orthogonal with respect to the canonical inner product in $\mathbb{R}^h$. Such orthogonal projections enable us to form 3D pictures which have mutually perpendicular x, y and z axes. The plot-window containing the visible space displays a sequence of projections in rapid succession. We denote any one of these as V_p, an (n x 3) matrix of data with elements v_{iap}. This matrix has n rows, one for each of the n cases, and 3 columns, one for each of the 3 variables. The projection is one of the series $V_0, V_1, V_2, \ ... \ V_{p-1}, V_p, V_{p+1}, \ ...$, where each V_p is in $\mathbb{R}^3$. The visible space, and its matrix representation, involve dynamically changing projections, thus the subscript p. The visible space contains points, one point for each case as it is projected from the high-dimensional data space into the visible space. The visual space may also contain vectors, one vector for each variable as it is projected from the data space.

The central problem in designing a guided tour of data space is how to enable the user to construct the sequence of projections V_p, and their corresponding visible spaces. As has been discussed by Young, Kent & Kuhfeld (1988) and by Hurley & Buja (1990), this is done by giving the data analyst tools for constructing a series of "target spaces", and tools for smoothly interpolating between the target spaces.

We begin by defining the initial visible space V_0 and two initial target spaces T_0. and T_1. The definition of the initial visible space is, simply, that V_0 is an (n x 3) matrix whose three columns equal three of the columns of X. The definition of the initial target spaces is equally simple: $T_0 = V_0$., and T_1 is define as three columns of X that are not the same as those used for T_0. The subscripts on the visible and target space matrices indicate that they vary, with the initial matrices indicated by zero. The subscripts are different for the two matrices: For the visible space V_p we use p to indicate that the visible

space presents varying projections from the data space. For the target space T_t we use t to indicate that the target changes over time.

The guided tour is a trigonometric interpolation that rotates a 3D projection of the data space from the first target by 90^o to the second target, and then by 270^o back to the first target, following the shortest geodesic path in 6-space. This rotation is shown in the tourplot window as a dynamically changing projection of the data space. The rotation continues until the user takes some action to guide it in a different direction.

The trigonometric interpolation that performs the rotation is:

$$V_p = T_t [\cos U_p] + T_{t+1} [\sin U_p] \quad \textbf{(EQ 1)}$$

where V_p is the matrix of coordinates of the objects seen in visible space, where the cosine and sine functions are applied to the *diagonal* of U_p, and where U_p is a diagonal 3x3 matrix with diagonal values $0^o \leq u_{paa} \leq 90^o$. The values u_{paa} increment from 0^o to 90^o dynamically over p, the increment being a multiple of 5^o.

An example of a tourplot, shown in four positions as it rotates in high-dimensional data space, is shown in Figures 2 through 5. The data used for this example are the crime rate, per 100,000 population, for seven major types of crime in each of the 50 United States for 1977. (The data were gathered by the FBI and were published in the 1979 Statistical Abstract of the United States by the US Department of Commerce). These data have been submitted to a principal components analysis. The figures show the scores of the 50 states on the principal components. This example is discussed more extensively by Young & Rheingans (1991a) who also present a video example (Young & Rheingans (1991b).

Figure 2 shows the tourplot, which is in the large window, in its initial V_0 position. What we see is the plane of the 3D space formed by the first three principal components. The first component is displayed horizontally and the second vertically (we can't see the third, which is pointing towards us). The 50 points represent the scores of the 50 states on the principal components. The small window named "First Target" shows T_0, and the "Second Target" window shows T_1.

This implementation uses real-time dynamic graphics which are guided by the user with high-interaction, immediate feedback, point-and-click mouse actions. The guided tour lets the user create and control rotation in a portion of a high-dimensional space which can have up to six dimensions. By clicking on the "Go/Stop" button the user starts the tourplot spinning towards the second target, which in this example are formed from principal components 4 5 and 6 (denoted PC4, PC5 and PC6). Figure 3 shows it spun about 1/3 of the way (about 30^o) towards the second target, as can be told by the position of axis label PC1 relative to PC4, and PC2 relative to PC5. Figure 4 shows it spun about 2/3'rds of the way, and Figure 5 shows it at the second target position (as can be seen by comparing with the "Second Target" window), where we see the plane formed by PC4 and PC5.

Previous exploration of these data (Young & Rheingans, 1991a) tells us that the first principal component is overall crime rate, whereas the second one is relative rate of property vs. personal crime. We have selected and labeled some of the states in the

space to emphasize this interpretation. On the left of Figure 2 we see low crime-rate states, and on the right high crime rate states. At the top are states with average rate, but excessively high rates of property crimes, whereas at the bottom are average rate states with excessively high personal crime rates.

The tourplot is useful to see whether these clusters of states are clustered in hD space. If all 2D and 3D views of the tourplot show the cluster of states remaining compact, then it is probably the case that they are clustered in hD space. However, if some views show the states separated, then the states are not clustered in hD space. For example, while Massachusetts, Rhode Island and Hawaii are close together in Figure 2, they separate in the other figures. California and Nevada remain close in these four views, but New York, which starts out close to them in Figure 2, separates from them in other views (closer study reveals New York is an outlier). On the other hand, North and South Dakota and West Virginia seem to remain close together in all four views, as do North and South Carolina, and Mississippi, Louisiana and Alabama.

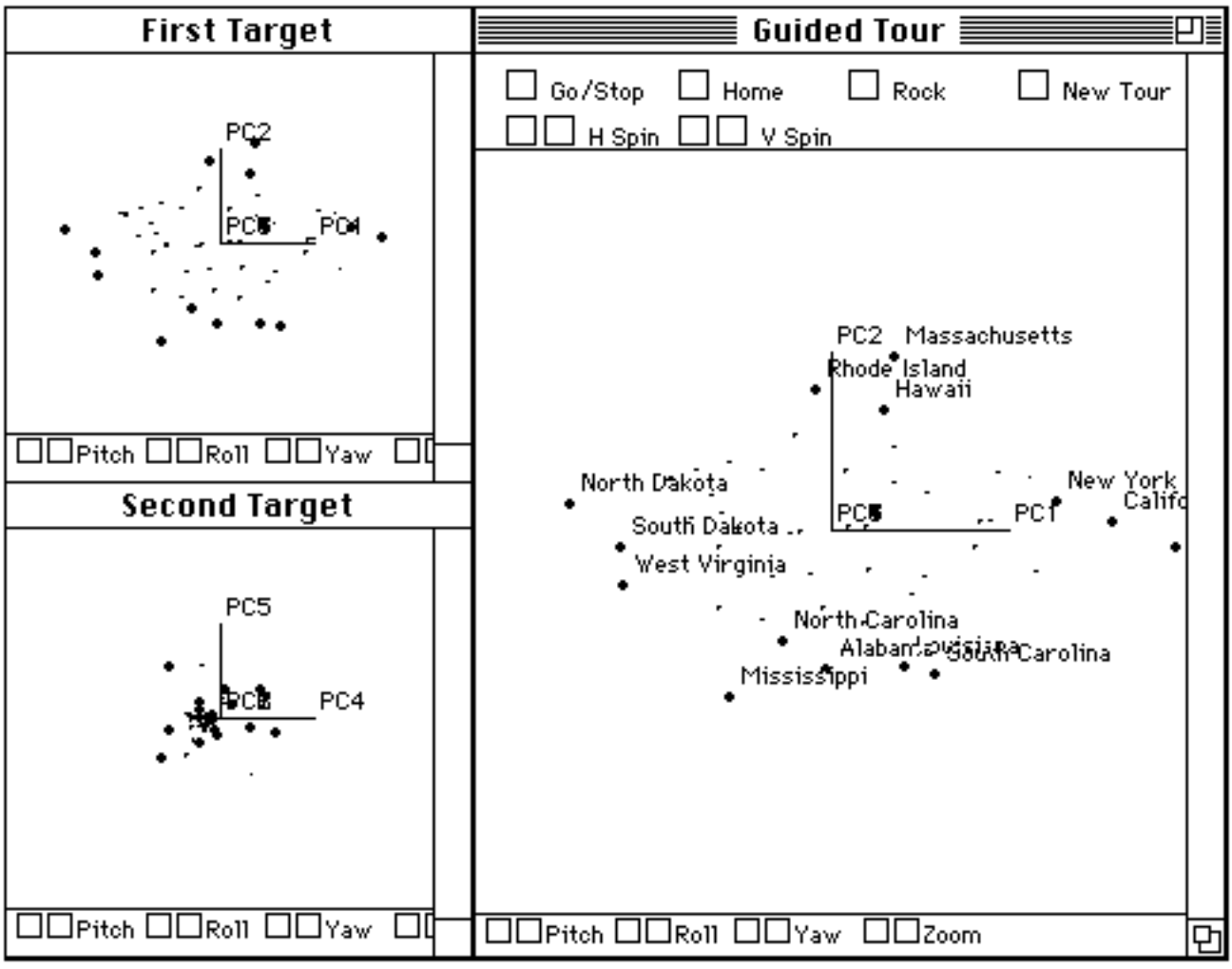

Figure 2: Tourplot in First Target Position

Note that the other buttons at the top of the tourplot. The user can use these to guide the tour taken by the tourplot. In particular, the “H Spin” and “V Spin” buttons control the rate of spinning on the horizontal and vertical axes. These buttons allow the user to change the path that the tour spins along through hD data space, thus enabling the user to control which part of the space is toured. Vertical spinning can be stopped so that spinning occurs only horizontally (involving the two targets’ horizontal axes). The opposite can be done so that the space spins only vertically. Spinning can be sped up on one or the other axes to change the tour path. In addition, the “Rock” button causes the tour to rock back-and-forth over a small angular displacement, helping the user to get a better understanding of the structure at some point in the tour. Finally, the “Home” button returns the plot to its initial position, so that the user can always start the tour over again when desired.

The remaining aspect of the Guided Tour is represented by the “New Tour” button. This button changes target spaces (as described momentarily) enabling the user to tour new parts of data space. Figure 6 shows the data space in a position that the user found inter-

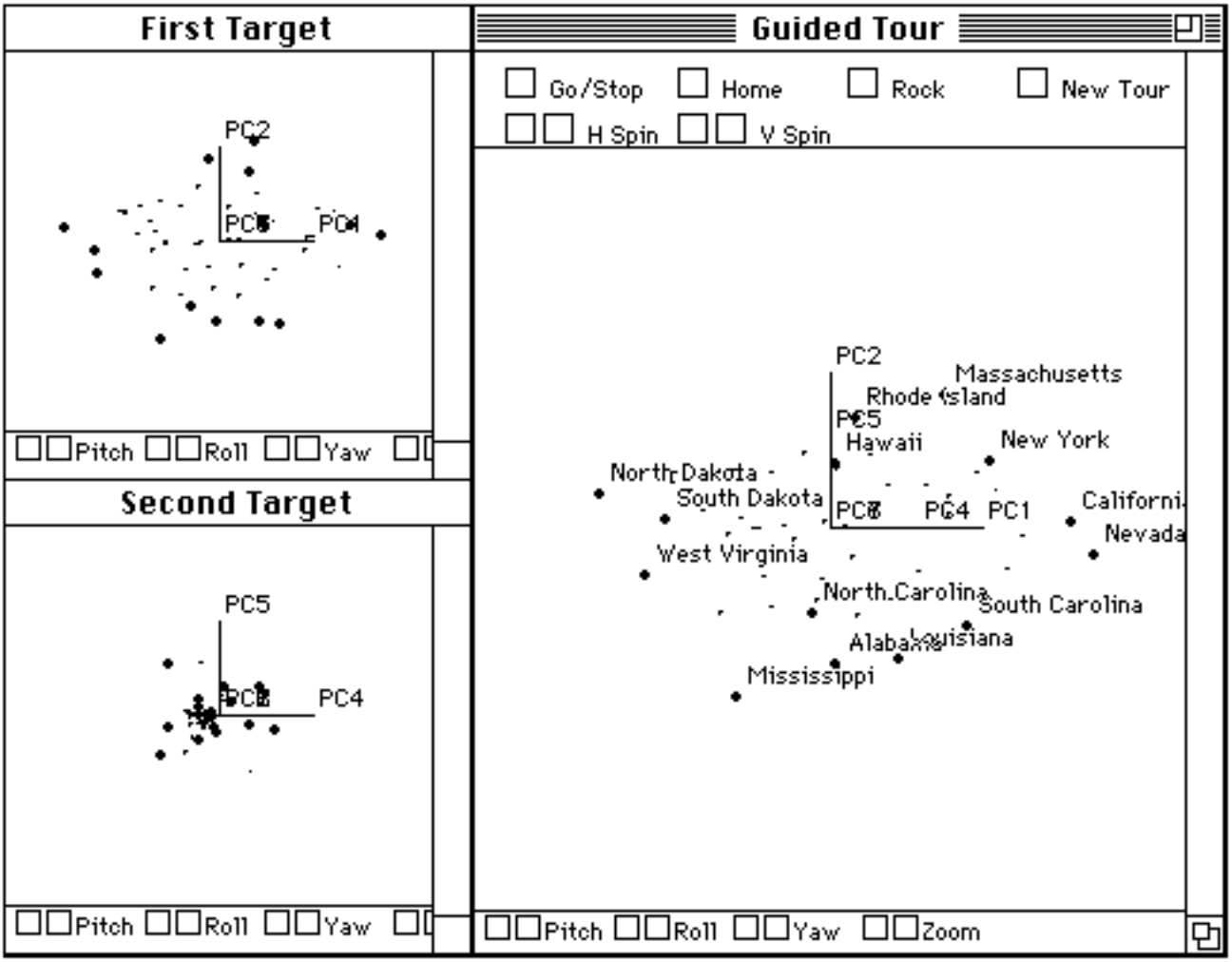

Figure 3: Tourplot Partially Rotated Towards Second Target

esting (the labeled states are somewhat more clustered than in other views). With the space stopped in this position, the “New Tour” button was clicked. This changes the “First Target” window to become identical to the tourplot, and changes the “Second Target” window according to equations given in the rest of this section. This button calculates the next two target spaces in the sequence of targets so that the user can create alternative 3D views of the data space, these views being used as targets for rotation.

The “New Tour” button does the following: It calculates the largest 3D space that is orthogonal to the visible space V_p (the largest invisible space). This space is “largest” in the sense that it contains the three longest mutually orthogonal dimensions which are also orthogonal to the visible space. It is also largest in the sense that it is the maximum variance 3D space orthogonal to the visible space. This tool is called hD-residualization by Young Kent & Kuhfeld (1988) and Young & Rheingans (1991a) because it computes the largest “residual” space in the invisible portion of the high-dimensional data space. The hD-residualization equations are based only the fact that the data space X is related

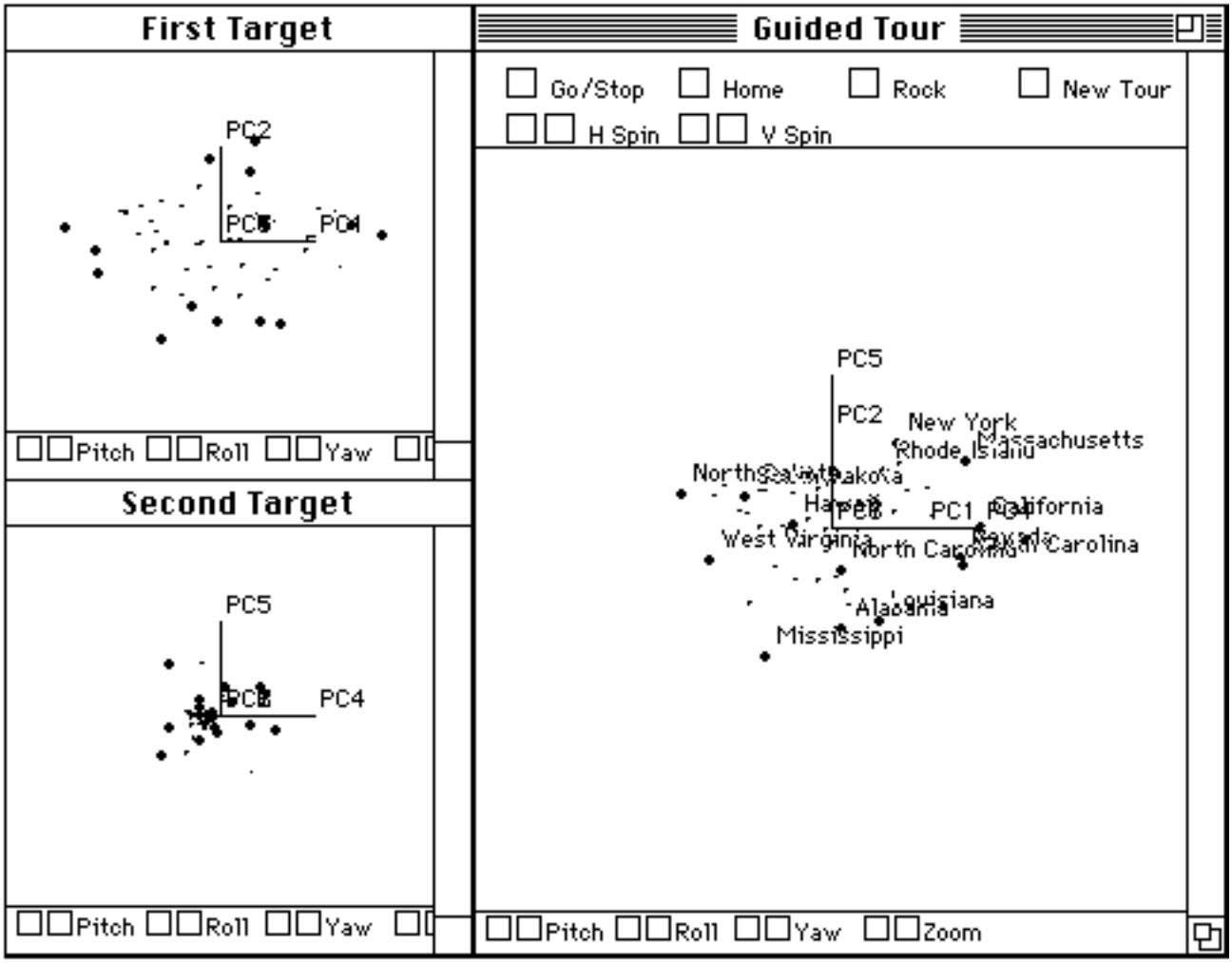

Figure 4: Tourplot Further Rotated Towards Second Target

to the visible space V_p by the equation (we omit the subscript on V_p for simplicity and because these equations hold for all values of p):

$$X = VB + R \quad \textbf{(EQ 2)}$$

where R is an (n x h) matrix of residual information between the two spaces, and B is a (3 x h) matrix of coefficients of three orthogonal linear combinations of the h variables, determined by the equation

$$B = \bar{V}X \quad \textbf{(EQ 3)}$$

where

$$\bar{V} = (V'V)^{-1}V'. \quad \textbf{(EQ 4)}$$

Then

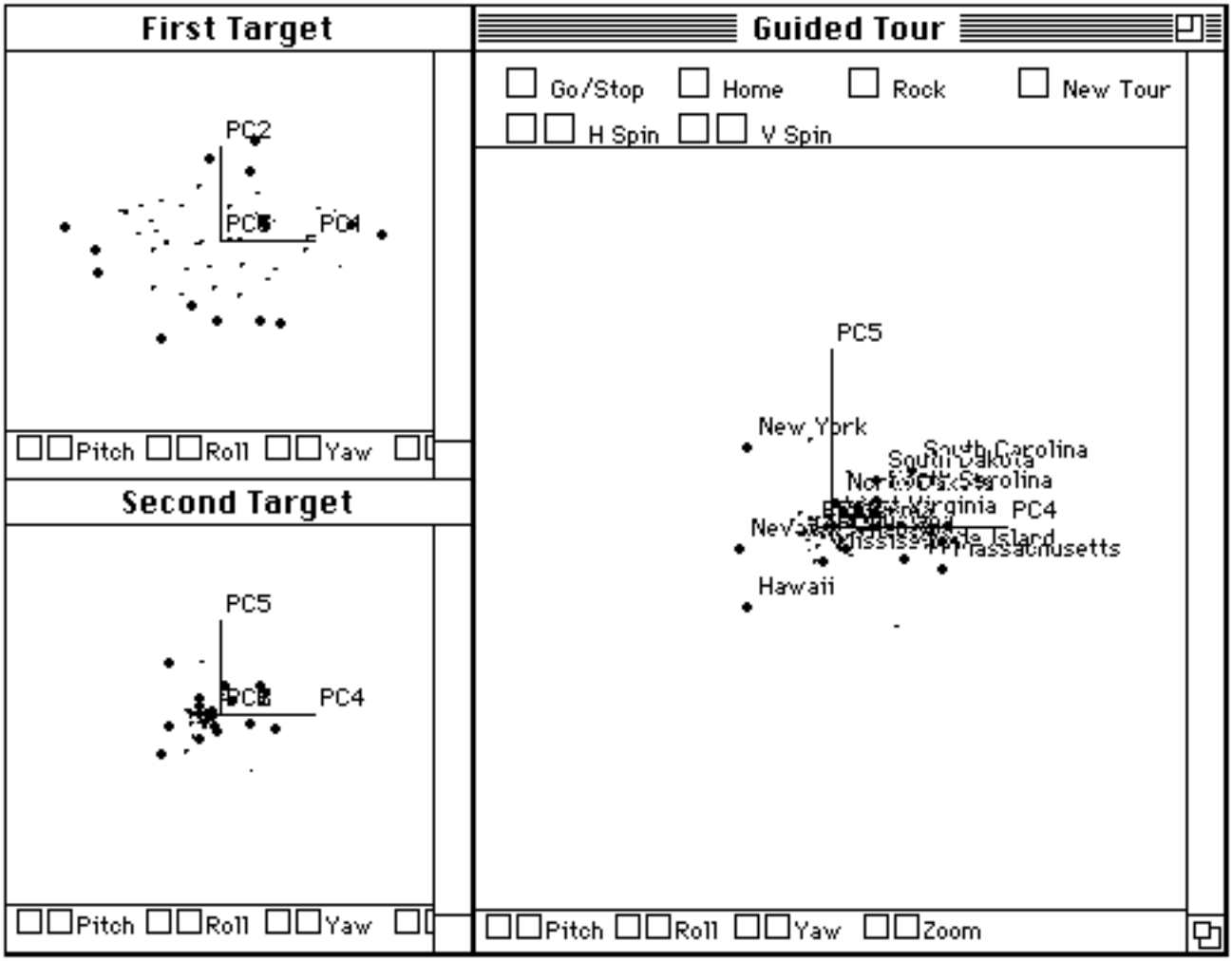

Figure 5: Tourplot in Second Target Position

$$R = X - V\nabla X \quad \text{(EQ 5)}$$

can be decomposed into

$$R = PQS' \quad \text{(EQ 6)}$$

using a singular value decomposition. We then define

$$T_{2t} = V \quad \text{(EQ 7)}$$

and

$$T_{2t+1} = PQ. \quad \text{(EQ 8)}$$

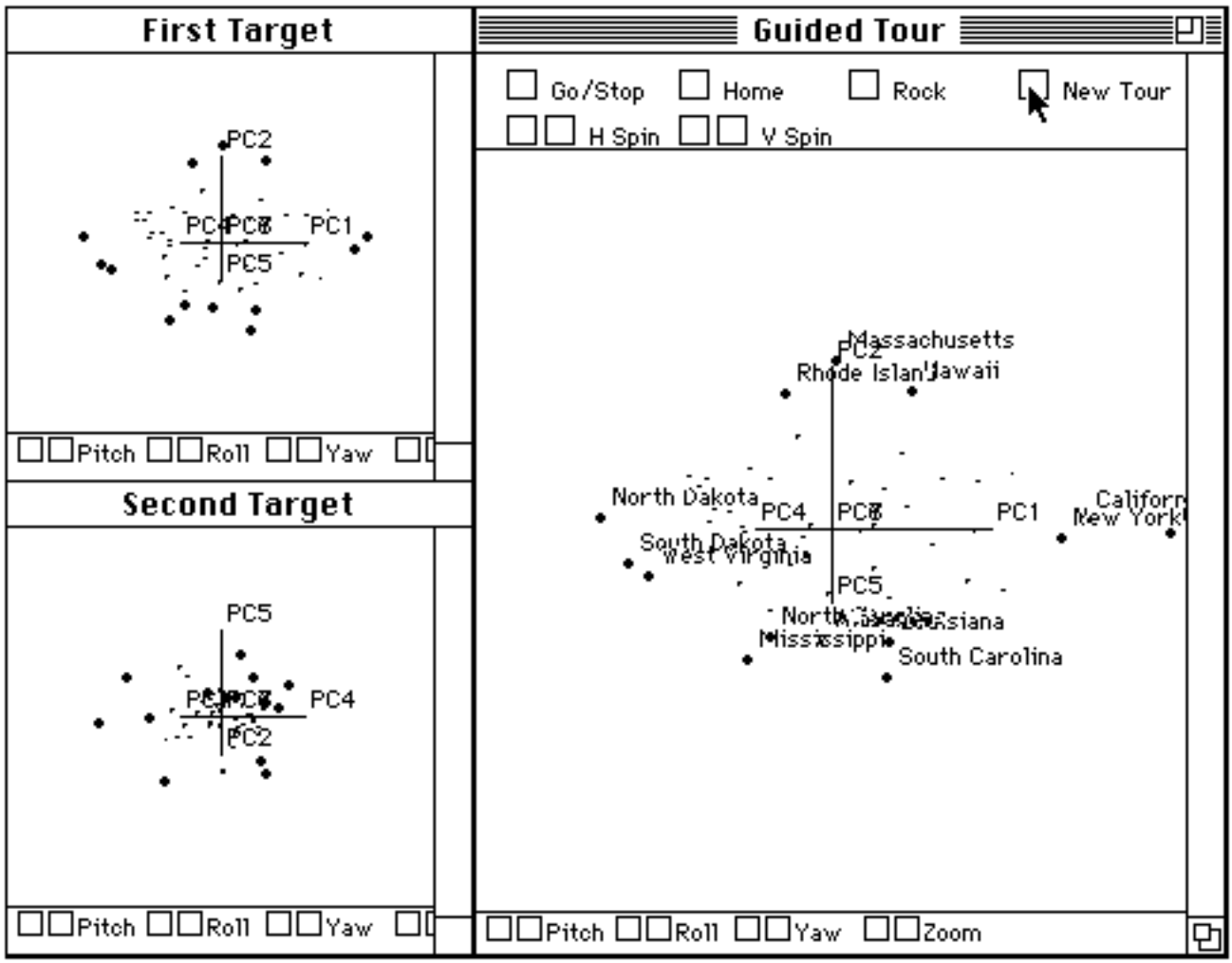

Figure 6: Tourplot with New Targets

2.4 Spreadplots: Algebraically Linked Plots

A spreadplot is a group of dynamic plots that are algebraically linked together by equations. When the user makes changes in one plot, the other plots change according to the equations linking the plots. Note that algebraic linkage is fundamentally different from empirical linkage. Empirical linkage involves observations and variables. Algebraic linkage involves equations. In fact, both kinds of linkages can simultaneously exist between plots in a spreadplot.

A spreadplot is the graphical equivalent of a spreadsheet: The main parallel between the two concepts is that each consists of a group of interacting cells, with the several cells being algebraically linked by equations. The obvious difference is that a spreadsheet's cells are numeric, whereas a spreadplot's are graphical. Just as in a spreadsheet, when changes are made in one cell, the algebraic links cause other change to occur. The difference is that a spreadplot shows the graphics that result from the underlying equations, not the numbers.

Perhaps the most important difference between a spreadplot and a spreadsheet is that spreadplot cells do not have to be arranged in a rectangular grid. The analogy to a spreadsheet breaks down when we think about "rows" or "columns" of plot-cells: A row (or column) of the spreadplot does not have a natural meaning. Rather, plot-cells are algebraically related to other plot-cells, but the actual arrangement of the cells into the spreadplot's "sheet" is arbitrary.

There is, however, an important parallel between spreadsheets and spreadplots: In both, some cells allow the user to change the information presented in the cell, whereas other cells do not. In particular, a graphical cell in a spreadplot can be one which lets the user make graphical changes whose implications flow to other cells in the spreadplot via the equations connecting the cells. However, not all graphical cells in the spreadplot need to support user-interaction. This is the same notion discussed in the introduction: cells in which a user can create changes contain active graphics, whereas those which do not support user interaction contain passive graphics. Thus, a passive cell contains a graphic which changes as a result of user interactions flowing from active graphical cells. Of course, the graphics in active cells can also be changed as a result of user interaction in other active cells.

The tourplot shown in figures 2-6 is an example of a spreadplot. The three plots are algebraically linked by the residualization equations given above. When the "New Tour" button is clicked the specific position of the tourplot in its spin between the two targets is used, along with the residualization equations, to update the two target windows via equations 7 and 8. Specifically, the "Guided Tour" window is linked to the "First Target" window by equation 7 and to the "Second Target" window by equation 8.

3 Visualizing Multivariate Models

In the previous section we presented graphical tools for visualizing multivariate *data*, focusing on plot-windows (groups of empirically linked plots), tourplots (high-dimensional spinplots), and spreadplots (groups of algebraically linked plots). In this section our attention is directed towards graphical tools for visualizing multivariate *models* -- tools that a specifically designed for specific models of multivariate data.

Before proceeding to the new tools introduced in this section, we wish to emphasize that all of the tools discussed in Section 2 on Visualizing Multivariate Data can be -- and will be -- used in various ways with models discussed in this section. They will be used to provide views of the data as seen through a model of the data, where these views are somehow related to the algebraic properties of the model.

However, the main focus of this section is on a new tool for statistical visualization that is only applicable when we have a model of some data. We call the tool Interactive Graphical Modeling. We introduce it in the next section. Then, in Section 3.2 we apply this tool to Principal Components Analysis, andin Section 3.3 we apply it to Multidimensional Scaling.

3.1 Interactive Graphical Modeling

Interactive Graphical Modeling is a statistical visualization technique for visually exploring the nature of alternative parameterizations of statistical models. The technique uses graphical tools to modify a model's parameterization, with the implications of the modifications being displayed as changes in the dynamic graphs that portray the model, its residuals, and its fit. A data analyst would use these tools to explore for a model of the data which provides better understanding of the data than the one provided by a traditional algebraic analysis.

Interactive graphical modeling assumes that an explicit data analysis model has been fit to the data by some means that generates initial estimates of the model’s parameters. This might be done, for example, for multiple regression, principal components or multidimensional scaling, using standard OLS techniques[1]. Once a model has been fit, it is visualized by a spreadplot whose cells reflect the geometry of the model, and whose algebraic links reflect the algebraic nature of the model. At least one of the cells, perhaps more, displays the model’s parameter estimates as a plot of points or vectors (or other appropriate graphical elements). This is the plot that presents the “structure” of the model. This plot has tools for changing the model’s parameter estimates, such as a tool for moving points or vectors.

In addition to the structure plot there would usually be plots displaying fit indices and residuals[2]. The structure plot is linked via appropriate equations to the fit and residual plots, so that when the model’s parameter estimates are changed (by moving points or vectors) the revised parameter estimates are used in the equations to update the plots of

1. These are the selection of approaches currently available in our ViSta testbed.
2. Residual plots are not yet available for MDS.

the residuals and fit values. In this fashion, interactive graphical modeling enables the data analyst to re-visualize the model many times, to support a search for the visualization that provides the most insight into the data.

We now turn to two specific examples of interactive graphical modeling, one for principal components analysis and the other for multidimensional scaling. Both of these examples should be thought of as work-in-progress. More complete descriptions of interactive graphical modeling is available for principal components analysis in Faldowski (1992), and for multidimensional scaling in McFarlane (1992).

3.2 Principal Components Analysis

3.3 Multidimensional Scaling

The data for Multidimensional Scaling (MDS) are different than the multivariate data we have been working with in previous parts of this paper, although the essence of the geometrical model (the data space) remains unchanged. Instead of an ($n \times h$) multivariate matrix of observations about n objects on h variables, there is an ($n \times n$) matrix of distance-like data that specifies the approximate distances between the n objects in some low-dimensional space. This matrix is symmetric with zeros on the diagonal. These data, which are called *dissimilarity* data rather than multivariate data, are denoted by the matrix Δ, which has elements δ_{ij}.

The geometrical model that we adopt for dissimilarity data is nearly the same model that we adopted for multivariate data. We assume that there is a data space in which each object is represented by an n-dimensional observation vector x_i, such that the Euclidean distances d_{ij} between the points in the data space equal the dissimilarity δ_{ij} observed between the two objects. Thus, abstractly, the entire set of data is represented by n points in an n-dimensional data space. We denote the data space as $\mathbb{R}^n$, an n-dimensional space of real numbers.

The purpose of an MDS analysis is to represent objects as points in a relatively low-dimensional space $\mathbb{R}^r$ such that the dissimilarity data are accurately represented by the interpoint distances in this space. We denote the r-dimensional space by the ($n \times r$) matrix X. The rows of X contain coordinates of the points in the r-dimensional space, the columns are the dimensions of the space. We assume that X is column centered, the centroid of the space is at the origin. To determine the nature of the dimensions in such an analysis, the analyst must be able to somehow visualize the points in the space. When clusters or patterns of points become evident, a reasonable interpretation of the dimensions may be facilitated.

In the ViSta-MDS algorithm[1] the dissimilarities are converted to a matrix of scalar products through a conversion discussed in Schiffman, Reynolds & Young (1981). The most important property of the scalar products matrix is that it is a matrix of products of

1. Note that ViSta-MDS permits more than one dissimilarity matrix, and that these matrices may be asymmetric. In this case the matrices are averaged, and the averaged matrix is made symmetric by averaging corresponding elements on either side of the diagonal.

vectors about an origin that corresponds to the centroid of all the points. Because this is a matrix of scalar products, we know that we can obtain the coordinates of each stimulus on each dimension by performing a singular value decomposition.

If the scalar products matrix is denoted *A*, we have

$$A = UVU', \quad \textbf{(EQ 9)}$$

where *U* is a matrix of singular vectors (eigenvectors) and *V* is a diagonal matrix of singular values (eigenvalues). We can then obtain our coordinates matrix, *X*, by performing the matrix multiplication

$$X = UV^{1/2}. \quad \textbf{(EQ 10)}$$

The jth column of the matrix *X* contains the coordinates of the n stimuli on the jth dimension. Only the first *r* columns of the *X* matrix are used.

The results of the previous equations form the initial spreadplot produced by ViSta-MDS. An example of the ViSta-MDS spreadplot is shown in Figure . This example involves data collected by Jacobowitz (1975). These data are the averages of the dissimilarity judgments of 8 judges about 15 colors. The entire set of colors includes those listed in the “Stimuli” window of the spreadplot, plus Black, which is scrolled off the top of the window. Several of the colors are those found on the spectrum (e.g. blue, green, yellow), but others (such as pink, gold, black and silver) do not appear on the spectrum. It was hypothesized that the MDS space *X* would consist of two regions, one containing the spectral and the other the non-spectral colors.

This MDS *X* space is plotted in the scatterplot-matrix, the scatterplot and the spinplot. As can be seen from these plots, the space does consist of the two hypothesized regions: After ten iterations[1] the spectral colors appear as a circle in the plane formed by the first and third dimensions, while the non-spectral colors are positioned away from the representation of the spectrum.

The “scree plot” shows the proportion of the scalar products’ variance which occurs along each successive dimension in the initial solution space (this plot does not update during the iterations). This plot provides a measure of the “usefulness” of each dimension. In order to decide how many dimensions of the stimulus space are useful, analysts look for an “elbow” in the scree plot. The hinge of the elbow is considered to be the last useful dimension.

Note that the scree plot shows fit for the scalar-products derived from the dissimilarity data, not the dissimilarity data itself. One the other hand, the “Stress plot” shows fit to the dissimilarity data itself. Specifically, it shows the degree to which the interpoint distances match the dissimilarity judgments with a measure called “Stress”. This measure is the square-root of the proportion of sum-of-squares of the data that is not fit by the model:

1. The initial spreadplot is not shown because it is essentially the same as the one shown, and because we wish to save space. The iterative algorithm is given below.

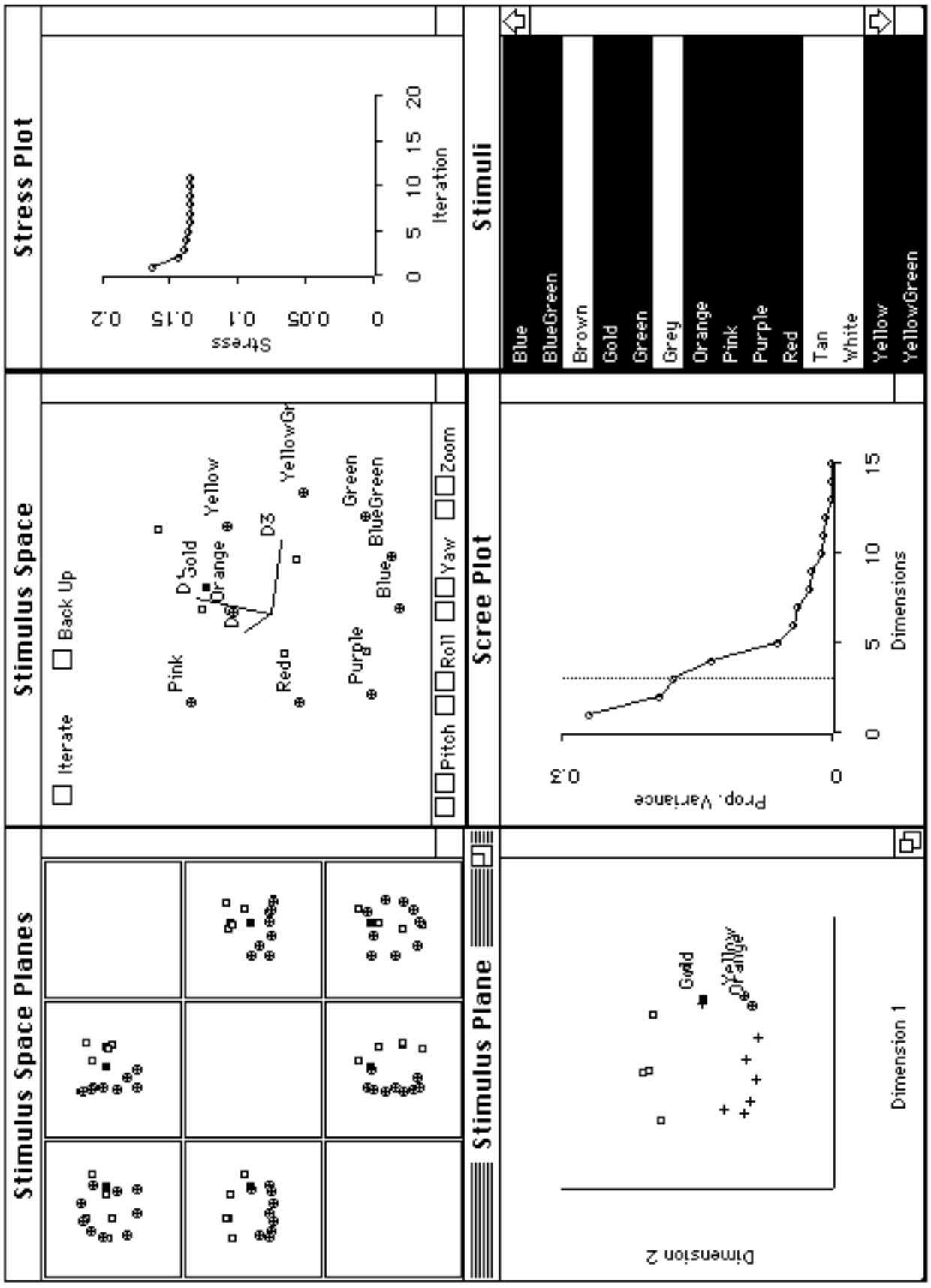

Figure 7: MDS Spreadplot before Interactive Graphical Modeling

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(\delta_{ij} - d_{ij})^2}{\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_{ij}^2}}, \qquad \text{(EQ 11)}$$

where the Euclidean distance d_{ij} between stimulus i and stimulus j is defined as

$$d_{ij} = \sqrt{\sum_{a=1}^{r}(x_{ia} - x_{ja})^2}. \qquad \text{(EQ 12)}$$

Note that the initialization procedure does not optimize fit of the model to the data, but to the scalar-products derived from the data. For this reason, ViSta-MDS contains an iterative procedure designed to optimize the fit of the model to the dissimilarities. Pushing the “Iterate” button (on the spinning plot at the top center of the screen) brings up a dialog box that asks the user for the desired number of iterations. Once the user specifies the number, the optimizing iterations begin. The stress for the initial configuration of points is σ=.163, while σ=.135 after 10 iterations.

The iterations are based on the Guttman-transform (Guttman, 1968) of the dissimilarities matrix and are designed to move the points so that the value of stress is minimized (so that the interpoint distances are as similar as possible to the dissimilarities). The transformation is based on the equation

$$\bar{X} = \frac{1}{2n}BX \quad , \qquad \text{(EQ 13)}$$

where the matrix B contains the elements

$$b_{ij} = \frac{-2\delta_{ij}}{d_{ij}} \qquad \text{if } i \neq j$$

$$b_{ii} = \sum_{i=1}^{n}\sum_{k\neq i}^{n}\frac{2\delta_{ik}}{d_{ik}} \qquad \text{if } i = j$$

$$b_{ij} = 0 \qquad \text{if } d_{ij} = 0 \qquad \text{(EQ 14)}$$

The matrix $\bar{X}$ is the configuration for the next iteration. The ratio of dissimilarities to distances is the basis of the Guttman-transform. A ratio of one implies that the distances perfectly match the dissimilarities. If the ratio is larger than one, the points are moved farther apart from each other. If the ratio is smaller than one, the points are moved closer together, and if the ratio is one, then there is no need to move the points at all.

The Guttman-transform produces non-increasing values of stress; that is, each successive solution is at least as good as the previous one in terms of its fit to the dissimilarities. When we have iterated to a minimum value of stress we would like to be able to

say that we have arrived at the overall minimum -- the “global” minimum. However, there may be other solutions that produce equally low, or even lower, values of stress. That is, the solution that we have may be a “local minimum”, not an overall global minimum, of the stress function. However, the solution produced by the iterations is often accepted by naive users as the only solution, even though there may well be other, more intuitively correct solutions, that fit as well (or may even fit better).

The problem of local minima is combatted with interactive graphical modeling. Interactive graphical modeling allows the analyst to graphically move a point in the MDS configuration and view the resultant change in the overall fit and structure of the model. When the point is moved, the algebraic links in the spreadplot are such that the corresponding elements in the matrix of coordinates *(X)* automatically update, a new value of stress is calculated and all plots are changed to reflect the newly revised model. If the value of stress does not worsen when the point is moved, then the previous configuration was a local minimum. Even if the fit worsens, we can iterate from the new configuration of points to see if the moved point remains in its new position. If so, then the new solution is a new (possibly local) minimum, and we say that the point “belongs” in the new location. If the point returns to its previous location, then we have returned to the previous (still possibly local) minimum, and we say that the point “belongs” in that previous position. If the new fit and structure are not satisfactory, the point(s) may be returned to the original position(s) using the “Back Up” button on the spinning plot.

As was noted above, the iterative procedure has arrived at a configuration of points that displays the two hypothesized regions, one for spectral colors and the other for non-spectral colors. This is one intuitively acceptable solution. However, it is also conceivable that judges would place the non-spectral color gold in between the spectral colors yellow and orange, as gold can be considered a combination of those two colors. It would be interesting to determine whether such an adjustment to the solution space would result in another (possibly local) minimum.

To test this theory, the “gold” point was moved from its position in Figure to a position between the orange and yellow points. The stress value increases some. The new configuration was then iterated five times. Throughout the iterations, the moved point remained very close to its new position (see Figure), indicating that the new configuration can be considered a local minimum. The stress for this new configuration is σ=.137, compared to the previous stress of σ=.135. This is an excellent example of a case in which a point fits nearly equally well in two different locations.

The problem of local minima has plagued analysts since the first developments in multidimensional scaling. Interactive graphical modeling provides MDS users an easy-to-use, powerful environment for exploring alternate MDS solutions. Through the use of interactive graphical modeling, analysts can search for other model representations that may fit the dissimilarity data as well as the optimized solution. The immediate updating of all information regarding the model allows the analyst to quickly determine whether the new model is an acceptable local minimum.

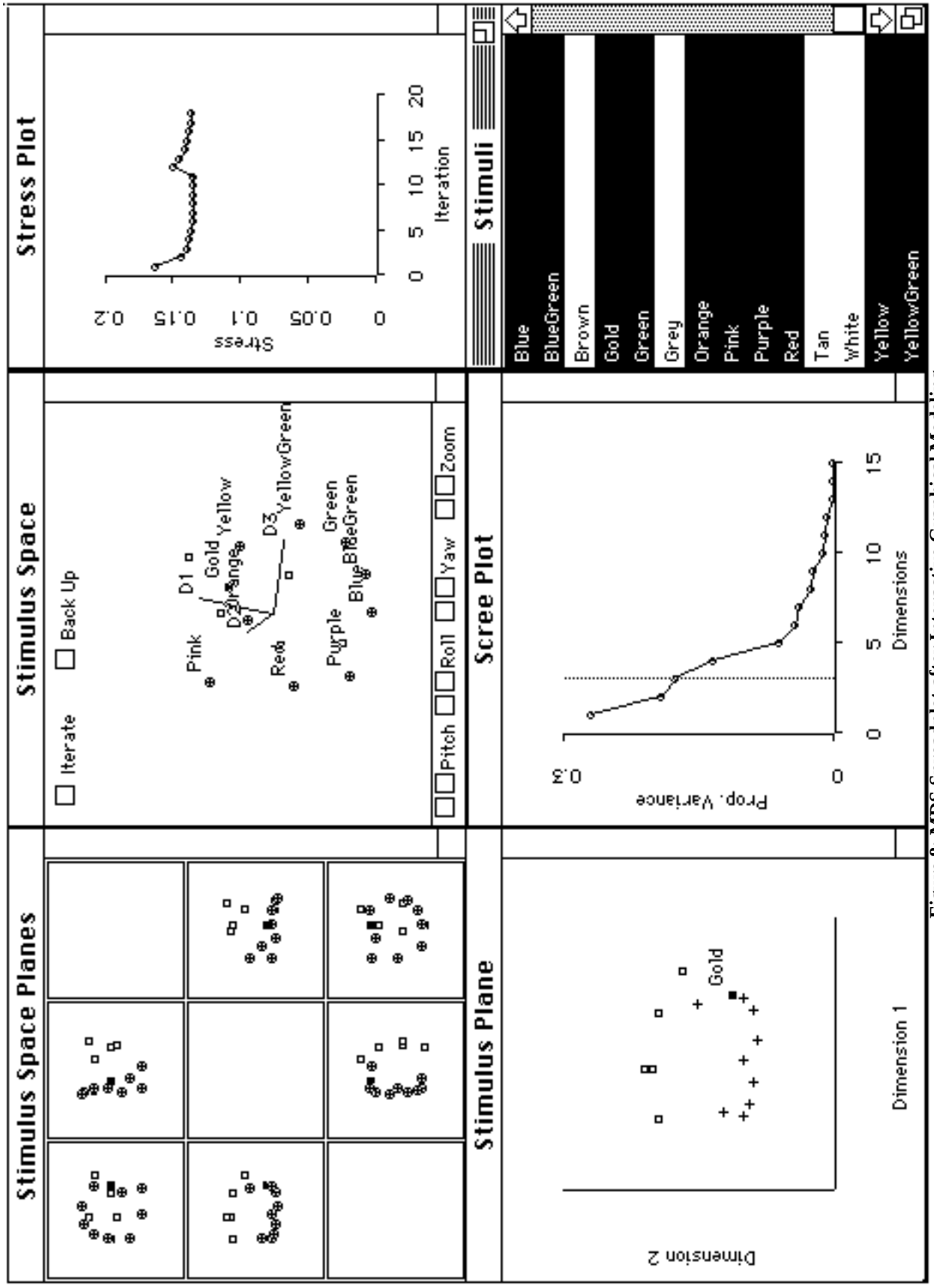

Figure 8: MDS Spreadplot after Interactive Graphical Modeling

4 Visualizing Multivariate Analyses

In Section 2 of this paper we discussed visualizing multivariate data, focusing on dynamic statistical graphics that can be used to explore and visualize the structure of multivariate data. In Section 3 we discussed visualizing multivariate models, focusing on interactive graphical modeling tools that can be used to explore and visualize models of multivariate data. In this section we discuss visualizing entire multivariate analysis sessions, focusing on the computational environment in which the multivariate analyses take place.

Our basic assumption is that the data analyst should be provided with a data analysis environment designed to maximize data analysis productivity and satisfaction. To accomplish this goal, the environment should reflect the sophistication of the user's data analysis knowledge. Furthermore, the environment should be designed to accommodate the complete range of data analysis sophistication, from novice to expert. Since the data analysis environment which does this for a novice is different from the one which does this for a sophisticated analyst, there should be specific aspects of the environment which are designed for specific levels of sophistication.

In particular, we believe that a data analysis environment that is appropriate for the entire range of sophistication should have the following five features: First, there should be optional *guidemaps* -- graphical diagrams that provide guidance -- to guide novice data analysts through complete data analyses. Second, there should be optional *workmaps* -- graphical diagrams that show the evolving structure of an ongoing analysis session -- to inform competent data analysts of the overall structure of their data analysis sessions. Third, there should be an optional *command line interface* to let sophisticated data analysts dispense with the visual aids when they find them unnecessary. Fourth, there should be an optional *batch mode interface* so that repetitive or "canned" analyses which do not require the presence of a data analyst can be performed. Fifth, and finally, there should be optional *guidance tools* to let expert data analysts create the guidance diagrams that are used by less expert analysts.

These five features should be very tightly coupled -- seamlessly integrated -- within a single data analysis environment so that the data analyst can switch effortlessly between them whenever desired. We discuss each of these notions in this section, along with the notion of tight coupling.[1]

4.1 Guidemaps for Novice Users

A statistical data analysis system should guide novice data analysts through the steps of the data analysis, particularly for multivariate data analysis. While this concept has been discussed (Chambers, 1981; Gale & Pregibon, 1982; Gale, 1988; Oldford and Peters, 1988; Pregibon & Gale, 1984, Hand, 1984; 1985; Lubinsky & Pregibon, 1988; Lubinsky, 1989; Lubinsky, Young & Frigge, 1990) guidance has been incorporated in only one commercial statistical system that we are familiar with (BBN Software, 1989), and this guidance is not presented as a visualization, but rather as unstructured text panels.

1. At the time this is being written, ViSta includes all of these notions except guidemaps and guidance tools.

Our concept (Lubinsky, Young & Frigge, 1992) is to provide guidance to the novice user via a visual diagram that indicates which steps should be chosen next - a *guidemap*. The structure of the guidemap doesn't change as the analysis proceeds, although its highlighting changes. Furthermore, new guidemaps appear as the analysis proceeds to guide the user with details of the analysis. In a guidemap the steps are indicated by buttons, and the sequence of steps by arrows pointing from one button to the next. Figure shows an example of a high-level and very general guidemap for multivariate analysis.

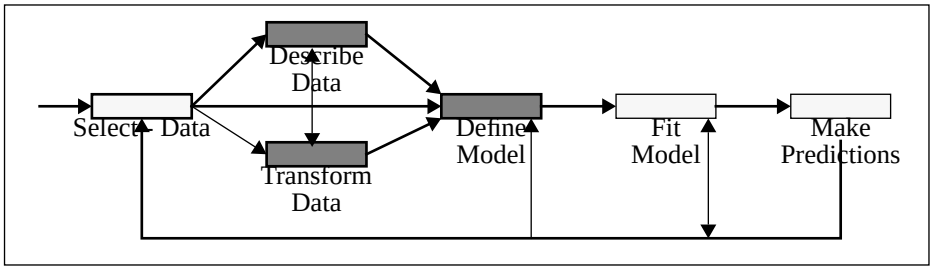

Figure 9 - A Guidemap

The user makes choices by pointing and clicking on the buttons with a mouse. Active buttons (which are dark) are suggested actions, whereas inactive buttons (the light ones) are actions that are not suggested. After a suggested action is taken the selection of active buttons changes to show the user which actions can be taken next. In this diagram the user has already selected data - the "Select Data" button is inactive and the following buttons are active. When the user clicks on one of the three active buttons, the corresponding action takes place, the button lightens in color, and the following buttons become active. For example, once the model is defined by clicking the "Define Model" button the "fit-model" button becomes inactive. Note that the guidemap is a cyclic graph whose nodes are the possible actions, and whose edges are the possible sequence of actions. For more detail, and an example of a partially working guidemap prototype, see Lubinsky, Young & Frigge (1990).

4.2 Workmaps for Competent Users

A data analysis environment should provide competent data analysts with a graphical interface that is a visual diagram of the steps taken in the analysis. Unlike a guidemap, which doesn't change, this *workmap* is created and expands as the analysis takes place. The user points and clicks to perform analyses and to create the structured analysis diagram. Note that the workmap is an acyclic graph whose nodes are the possible actions and whose edges are the sequence of possible actions. For more details see Young & Smith (1991).

An example of a workmap is shown in Figure . In this analysis the analyst first loaded in datafile named "car-ratings", creating a data icon with the same name. These data were then standardized, creating a new data object with an icon named "STD:car-ratings". The analyst then loaded in a second datafile named "car-pref14", creating a third data object and another data icon with the same name. These data were analyzed by the "PrinComp" method for principal components analysis. This produces a method icon named "PrinComp", and a model icon named "PCA:car-pref14". The analyst then requested that the model create three new data objects of scores, coefficients and input

data. Finally, the analyst merged the standardized ratings with the principal component scores. Any of the icons in this diagram can be opened in various ways to visualize or report data or results. This example corresponds to the example in Section 4.3 on command lines, and the example in Section 4.5 on batch mode.

4.3 Command Lines for Sophisticated Users

For sophisticated data analysts the data analysis environment should provide a command line interface. An example of commands used in ViSta is shown in Figure 11.

```
> (def car-ratings (load-data ":ViSta:Data:car-ratings.lsp"))
CAR-RATINGS
> (def std-car-ratings (standardize-data :dialog nil :mean 0 :stdv 1))
STD-CAR-RATINGS
> (send std-car-ratings :report-data)
NIL
> (def car-prefs (load-data ":ViSta:Data:car-pref14.lsp"))
CAR-PREFS
> (def pca-car-prefs (principal-components :dialog nil :corr t))
PCA-CAR-PREFS
> (send pca-car-prefs :create-data-objects :dialog nil
        :scores t :coefs t :input t)
#<Object: 1870806, prototype = MV-DATA-OBJECT-PROTO>
>
```

Figure 11 - Command Line Interface

These commands are entered through the keyboard, causing the analysis to take place.They also create the structured analysis diagram. (The diagram may be hidden, if desired). In this example, data named “car-ratrings.lsp” are loaded from the ViSta:Data folder. These data are standardized, with a report (listing) being obtained. Then data named “car-pref14.lsp” are loaded from the same folder. These data are submitted to a principal components analysis. Finally, three data objects are created of the results of the analysis.

4.4 Guidance Tools for Expert Users

A data analysis environment should provide expert analysts with tools to create guidance diagrams that can be used by other users. These diagrams should be constructed by using the mouse to point and click, or by using the command line to type commands. A guidance diagram has already been shown, but we have not yet developed tools for creating guidance diagrams, this being a research topic for the future.

4.5 Batch Mode - Automated Analysis in Repetitive Situations

The four kinds of environments discussed above are all *highly interactive.* This means that as soon as an icon is clicked, or a command is typed, the data analysis environment 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 preferable to be able to collect all commands together into a file and run them all at once without user interaction. This is called “batch” mode because all commands are run as a batch.

An example of a ViSta batch mode file is shown in Figure 12. In this example the system will load data concerning car ratings, which are then standardized. It will then produce a report (listing) of these data, followed by a visualization and some summary


```
(def car-ratings (load-data "car-ratings.lsp"))
(def std-car-ratings
     (standardize-data
      :dialog nil
      :mean 0
      :stdv 1))
     (send std-car-ratings :report-data)
     (send std-car-ratings :visualize-data)
     (send std-car-ratings :summarize-data)
(def car-prefs (load-data  "car-pref14.lsp"))
(def pca-car-prefs
     (principal-components
           :dialog nil
           :corr t))
     (send pca-car-prefs :report-model)
     (send pca-car-prefs :visualize-model)
     (send pca-car-prefs :create-data-objects
           :dialog nil
           :scores t
           :coefs  t
           :input  t)
```

Figure 12 - Batch Mode

statistics. The system then loads data about car preferences which are then submitted to a principal components analysis. A report and a visualization is produced of the results and then output data objects are created. This batch code corresponds to the analyses discussed in Section 4.2 on workmaps and in Section 4.3 on the command line interface.

4.6 Tight Coupling of All Environments

The five data analysis features discussed above are tightly coupled, as can be seen from the previous sections. The guidance diagrams used by novice analysts generate commands that are identical to those typed by sophisticated users with the command line interface. The graphical interface used by competent analysts also generates the same commands. The commands, in turn, generate the structured analysis diagram and perform the data analysis. These commands can be used in batch files.

It is possible to switch between the several kinds of environments at any time. When the sophisticated user moves into an unfamiliar type of data analysis, or when the analyst looses track of the overall structure of the analysis, the analyst can switch from the command line interface to the graphical interface, with the entire structured history of the analysis session being presented. Similarly, the moderately competent analyst can switch guidance diagrams on or off as desired.

5 Conclusion

In this paper we have discussed three major aspects of multivariate statistical visualization, namely data visualization, model visualization and analysis visualization. We believe that data analysis systems of the 21st century will incorporate the methods we have presented, and that they will help the data analyst have a more insightful, productive and satisfying experience, enabling them to more clearly “see what the data seem to say”.

6 References

Asimov, D. (1985), "The Grand Tour: A Tool for Viewing Multidimensional Data," SIAM J. Scientific and Statistical Computing, 6, 128-143.

Becker, R.A. and Cleveland, W.S. (1986), Brushing Scatterplots. Unpublished manuscript, AT&T Bell Laboratories.

Buja, A. and Asimov, D. (1986). “Grand Tour Methods: An Outline,” Computer Science and Statistics: Proceedings of the 17th Symposium on the Interface, Elsevier, Amsterdam.

Carr, D.B., Littlefield, R.J., Nicholson, W.L. and Littlefield, J.S. (1987), “Scatterplot Matrix Techniques for Large N,” J. Amer. Statistical Assn., 82, 424-436.

Chambers, J.M. (1981), “Some thoughts on expert software”. Proc. 13th Symp. Interface of Comp. and Stat., 36-40.

Cleveland, W.S. & McGill, M.E. (1988), “Dynamic Graphics for Statistics”. Wadsworth, Belmont, CA.

Donoho, A.W., Donoho, D.L. and Gasko, M. (1986), MACSPINTM: A Tool for Dynamic Display of Multivariate Data. Wadsworth, Inc., Monterey, Calif.

Donoho, A.W., Huber, P.J., Ramos, E. and Thoma, H. (1982), “Kinematic Display of Multivariate Data,” Proceedings, NCGA Computer Graphics '82.

Faldowski, R.A. (1992) *Interactive Graphical Modeling.* Ph.D. Dissertation Proposal, Univ. N. Carolina Psychometrics Lab.

Fisherkeller, M.A., Friedman, J.H. and Tukey, J.W. (1974), "An Interactive Multidimensional Data Display and Analysis System," SLAC PUB 1408. Stanford Linear Accelerator Center, Stanford, Calif.

Friedman, J.H., McDonald, J.A. and Stuetzle, W. (1982), “An Introduction to Real Time Graphical Techniques for Analyzing Multivariate Data,” Proc. NCGA Computer Graphics '82.

Friedman, J.H. and Tukey, J.W. (1974), "A Projection Pursuit Algorithm for Exploratory Data Analysis," IEEE Transactions on Computers, IC-23.

Gabriel, K.R. (1981), "Biplot Display of Multivariate Matrices for Inspection of Data and Diagnosis," in: V. Barnett (ed.), Interpreting Multivariate Data. Wiley, London.

Gabriel, K.R. and Odoroff, C.L. (1986), “ANIMATE: An Interactive Color Statistical Graphics System for Three Dimensional Displays,” Computer Graphics '86 Conference Proc., 3, 723-731.

Gale, W.A. (1988), “Artificial Intelligence and Statistics”. Addison Wesley, Reading Massachusetts.

Gale, W.A. & Pregibon, D. (1982), “An expert System for Regression Analysis”. Proc. 14th Symp. on Interface of Comp. and Stat., 110-117.

Guttman, L. (1968), A general nonmetric technique for finding the smallest coordinate space for a configuration of points. “Psychometrika”, 33, 469-506.

Huber, P.J. (1985), “Projection Pursuit,” Ann. Statis., 13, 435-475.

Huber, P.J. (1987), "Experiences with Three-Dimensional Scatterplots," J. Amer. Statistical Assn., 82, 448-452.

Hurley, C. & Buja, A. (1990) Analyzing High-Dimensional Data with Motion Graphics. SIAM J. Scientific and Statistical Computing, 11, 1193-1211.

Jacobowitz, D. (1975), “Development of Semantic Structures”. Unpublished Doctoral Dissertation. Department of Psychology, University of North Carolina at Chapel Hill.

Kuhfeld, W.F. (1986), “Metric and Nonmetric Plotting Models,” Psychometrika, 51, 155-161.

Lubinsky, D.J. & Pregibon, D. (1988), “Data Analysis as Search”, J. Econometrics, 38, 247-268.

Lubinsky, D.J. (1989), “Data Analysis Strategy Representation”, Proc. 21st Symp on Interface of Comp and Stat.

Lubinsky, D.J., Young, F.W. & Frigge, M.L. (1990) “Representing and Using Data Analysis Strategies”. Technical Report, Bell Telephone Laboratories, Holmdel, NJ.

McDonald, J.A. and Pedersen, J. (1985), “Computing Environments for Data Analysis (I. Introduction; II. Hardware),” SIAM J. Sci. Statistical Computing, 6, 1004-1012, 1013-1021.

McFarlane, M.M. (1992), “Interactive Graphical Modeling for Multidimensional Scaling. Unpublished Master’s Thesis, UNC Psychometric Laboratory, Chapel Hill NC.

Nicholson, W.L. and Carr, D.B. (1984), “Looking at More than Three Dimensions,” in: Billard (ed.), Computer Science and Statistics: The Interface, North Holland Press, 201-209.

Oldford, W. & Peters, S. (1988), “DINDE: Towards more Sophisticated Software Environments for Statistics.” Siam J. Sci. Stat. Comput., 9, 191-211.

Pregibon, D. & Gale, W.P. (1984), “REX: An expert system for regression analysis.” Proc. COMPSTAT 84, 242-248.

Schiffman, S.S., Reynolds, M.L., & Young, F.W. (1981), “Introduction to Multidimensional Scaling: Theory, Methods and Algorithms”. New York: Academic Press.

Stuetzle, W. (1987), "Plot Windows," J. Amer. Statistical Assn., 82, 466-475.

Tierney, L. (1991), “Lisp-Stat: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics”. Wiley, New York.

Tufte, E.R. (1983), The Visual Display of Quantitative Information. Graphics Press, Cheshire, Conn.

Tukey, J.W. (1977), Exploratory Data Analysis. Addison-Wesley, Reading, Penn.

Tukey, J.W. and Tukey, P.A. (1980), "Graphical Display of Data Sets in 3 or More Dimensions," in: V. Barnett (ed.), Interpreting Multivariate Data, Wiley, London, 189-275.

Young, F.W. (1992) “ViSta: The Visual Statistics System”. UNC Psychometric Laboratory, Chapel Hill NC.

Young, F.W., Faldowski, R.A. & Harris, D.F. (1992) The Spreadplot: A graphical spreadsheet of Algebraically Linked Dynamic Plots. *ASA Proceedings of the Section on Statistical Graphics*, (in press)

Young, F.W. and Hamer, R.M. (1987), Multidimensional Scaling: History, Theory and Applications. Erlbaum, Hillsdale, N.J.

Young, F. W., Kent, D. P. and Kuhfeld, W. F. (1988). “Dynamic Graphics for Exploring Multivariate Data,” in Cleveland, W. S. and McGill, M. E. (eds.): Dynamic Graphics for Statistics. Wadsworth, Inc., Belmont, Calif

Young, F.W. & Rheingans, P. (1991a) Visualizing Structure in High-Dimensional Data. *IBM Journal of Research and Development*. **35**, 97-107.

Young, F.W. & Rheingans, P. (1991b) Visualizing Multivariate Data with VISUALS/Pxpl. (Video). *IBM Journal of Research and Development*. **35**, (video supplement).

Young, F.W. & Smith, J.B. (1991) Towards a Structured Data Analysis Environment: A Cognition-Based Design. In: W., Buja, A., & Turkey, P. (Eds.) Computing and Graphics in Statistics. *IMA Volumes in Mathematics and it Applications*, **36**, Springer Verlag. 252-279.