Interactive High-Dimensional Data Visualization

Andreas BUJA, Dianne COOK, and Deborah F. SWAYNE

We propose a rudimentary taxonomy of interactive data visualization based on a triad of data analytic tasks: finding Gestalt, posing queries, and making comparisons. These tasks are supported by three classes of interactive view manipulations: focusing, linking, and arranging views. This discussion extends earlier work on the principles of focusing and linking and sets them on a firmer base. Next, we give a high-level introduction to a particular system for multivariate data visualization—XGobi. This introduction is not comprehensive but emphasizes XGobi tools that are examples of focusing, linking, and arranging views; namely, high-dimensional projections, linked scatterplot brushing, and matrices of conditional plots. Finally, in a series of case studies in data visualization, we show the powers and limitations of particular focusing, linking, and arranging tools. The discussion is dominated by high-dimensional projections that form an extremely well-developed part of XGobi. Of particular interest are the illustration of asymptotic normality of high-dimensional projections (a theorem of Diaconis and Freedman), the use of high-dimensional cubes for visualizing factorial experiments, and a method for interactively generating matrices of conditional plots with high-dimensional projections. Although there is a unifying theme to this article, each section—in particular the case studies—can be read separately.

Key Words: Brushing; High-dimensional projections; Multiple linked views; Plot matrices; Real-time graphics; Taxonomy of data visualization.

1. INTRODUCTION: ZOOLOGY OF MULTIVARIATE DATA VISUALIZATION

The goal of this presentation is two-fold:

- To propose a "zoology" of some of the major data visualization efforts to date.
- To show off some of our own "zoo" of data visualization techniques.

With a "zoology" we mean a taxonomy that brings order to seemingly disconnected or competing visualization efforts. With our own "zoo" we mean the XGobi system that contains our own collection of critters. In the same way that animals in a zoo are instances

---

Andreas Buja is Member of Technical Staff, AT&T Bell Laboratories, Murray Hill, NJ 07974-0636; e-mail: andreas@research.att.com. Dianne Cook is Assistant Professor, Department of Statistics, Iowa State University, Ames, IA 50011-1210. Deborah F. Swayne is Research Scientist, Bellcore, Morristown, NJ 07960-6438.

Journal of Computational and Graphical Statistics, Volume 5, Number 1, Pages 78–99

of species of the Linnean taxonomy, our own XGobi implementations are instances of species of a visualization taxonomy.

It is useful to develop a taxonomy for data visualization, not only because it brings order to disjointed techniques, but because it clarifies and interprets ideas and purposes behind the techniques. In addition, a taxonomy may trigger the imagination to dream up new and as yet undiscovered techniques.

A caveat is necessary for any taxonomy: it will do injustice to some species that are not easily placed or that have been forgotten or ignored by the taxonomist. Oversight and incompleteness on our part should be taken as challenges to correct or refine the taxonomy.

The taxonomy we propose classifies not so much individual techniques but approaches and aspects thereof. A concrete technique may therefore draw on several parts of the visualization taxonomy.

At the root of the taxonomy is a division of data visualization into two areas:

- *rendering*, or what to show in a plot; and
- *manipulation*, or what to do with plots.

The first area, rendering of multivariate data, comprises all decisions that go into production of a static image. The major rendering decision concerns the basic type of plot, which can serve as a start for further subdivision:

- *scatterplots*, where cases are represented by locations of points;
- *traces*, where cases are represented as functions of a real parameter, such as parallel coordinate plots (Inselberg 1985; Wegman 1990) and Andrews curves (Andrews 1972); and
- *glyphs*, where cases are represented as complex symbols whose features are functions of the data, such as trees and castles (Kleiner and Hartigan 1981), stars (Newton 1978), shape coding (Beddow 1990), Chernoff faces (Chernoff 1973). [We think of glyphs as location independent representations of cases. For successful use of glyphs, however, some sort of suggestive layout is often essential because comparison of glyph shapes is what this type of rendering primarily affords. If glyphs are used to enhance a scatterplot, the scatterplot takes over the layout function. For an example, see Carr and Nicholson's (1988) ray glyphs.]

We limited ourselves to multivariate visualization and omitted from this list important techniques for visualizing 1-dimensional distributions (histograms, density plots, cdf plots, Q-Q plots, jitter plots, boxplots), time series (time series plots), and functions of one and two variables (perspective plots, level plots).

We structured the described catalog of types of plots according to our own idea of the intrinsic relations among plotting techniques. Because they are not at the heart of our own interest, this part of the taxonomy is getting short shrift. In XGobi we stick to basic scatterplots, possibly enhanced with color, trivial glyphs, and line drawings.

The second area—here imprecisely called "manipulation"—refers to

- how we operate on individual plots; and
- how we organize multiple plots.

The purpose of these manipulations is to support the search for structure in data. In the practice of data visualization, there usually exists a larger context of open-ended problem solving. In such contexts, data visualization systems are most useful if they provide plot manipulation tools that support extensive searching.

The question then is to identify and classify useful plot manipulations. An organizing principle for plot manipulations is in terms of the basic search tasks that these manipulations are supposed to support. Here is a set of three tasks that we feel are fundamental to data exploration:

- *Finding Gestalt*: Local or global linearities and nonlinearities, discontinuities, clusters, outliers, unusual groups, discreteness, and so on, are examples of Gestalt features that can be of interest.
- *Posing queries*: This is a natural task after initial Gestalt features have been found and their identification and characterization is desired. Queries can concern individual cases as well as subsets of cases. The goal is essentially to find intelligible parts of the data.
- *Making comparisons*: Two types of comparisons are frequently made in practice—comparisons of variables or projections, and comparisons of subsets of the data. In the first case one compares views "from different sides," in the second case one compares views "of different slices" of the data. In either case it is likely that large numbers of plots are generated; it is then a challenge to organize the plots in such a way that meaningful comparisons are possible.

These tasks can be supported by a variety of tools, but a natural pairing of tasks with near-optimal categories of tools may be the following:

- *Finding Gestalt: Focusing individual views*. By focusing we mean any operation that is an extension of manipulating a camera, such as deciding from which side to look at the object and in which magnification and detail. Focusing views includes choosing the variables or (more generally) the projections for viewing, but also choosing aspect ratio, zoom, and pan.
- *Posing queries: Linking multiple views*. In graphical data analysis it is natural to pose queries graphically, for example with the familiar brushing techniques—coloring or otherwise highlighting a subset of the data means issuing a query about this subset. It is then equally natural that the response to the query be given graphically. This is achieved by showing information about the highlighted subset in other views. It is therefore desirable that the view where the query is posed and the views that present the response are linked. Ideally, responses to queries are instantaneous.
- *Making comparisons: Arranging many views*. It is often a powerful informal technique to arrange large numbers of related plots for simultaneous comparison. The

most useful arrangements are matrix-like, such as in scatterplot matrices of pairwise variable plots, but other arrangements can be useful as well.

The notions of focusing single views and linking multiple views were discussed by Buja, McDonald, Michalak, and Stuetzle (1991). The notion of arranging many views is a new addition. In the following sections we discuss these three categories of tools in turn.

1.1 FOCUSING INDIVIDUAL VIEWS

Focusing determines what Gestalt of the data is seen. The meaning of focusing depends very much on the type of rendering chosen. For scatterplots, focusing includes the choice of projection, aspect ratio, zoom, and pan. For trace techniques such as parallel coordinates and Andrews curves, focusing includes the choice of variables, their order, their scale, and the scale and aspect ratio of the plot. For glyph techniques, focusing includes the choice of variables and their mapping to glyph features, as well as the layout of the glyphs on the plotting surface.

All aspects of focusing can be subject to interactive control in visualization systems. Some focusing choices are discrete, such as which variables to display, but other choices are continuous, and these can be subjected to real-time control and animation. In scatterplots, for example, the choice of projection, aspect ratio, pan, and zoom all have underlying continuous parameters that can be changed in small steps and at rapid speed, resulting in animations that can possibly be controlled by the user's input.

The gain resulting from animation and real-time control is not incremental but a quantum leap:

- Motion can *add at least one spatial dimension* to the visual perception of Gestalt, as exemplified by real-time rotations in 3-D data spaces.
- A general gain from continuous motion of any kind is *object continuity*: It allows a viewer to keep track of plotted objects as the animation moves smoothly through a sequence of plots.
- Real-time control, as opposed to precomputed animation, affords *searching and tuning* of views with visual feedback.

Two systems that provide animated projections in the form of 3-D data rotations and grand tours are Lisp-Stat (Tierney 1990) and XGobi (Swayne, Cook, and Buja 1991), both freely available. The systems offering 3-D rotations alone are too numerous to be listed. Some references to higher-than-3-D projection techniques are Asimov (1985), Buja and Asimov (1986), Hurley and Buja (1990), Cook, Buja, Cabrera, and Hurley (1995).

A related animation method was proposed by Young, Kent, and Kuhfeld (1988). They use linear interpolation to travel between two 3-D spaces.

A lesser known application of real-time animation is to scaling, zooming, and panning: It allows one to search a potentially infinite plotting plane with the equivalent of a microscope that has shifting focus and varying magnification. In addition, it allows one to search among aspect ratios, which is important for some time series applications (Buja, Asimov, Hurley, and McDonald 1988, p. 282f).

1.2 LINKING MULTIPLE VIEWS

Multiple linked views are the optimal framework for posing queries about data. A user should be able to pose a query graphically, and a computer should be able to present the response graphically as well. Both query and response should occur in the same visual field. This calls for a mechanism that links the graphical query to the graphical response. A graphical user interface that has such linking mechanisms is an implementation of the notion of "multiple linked views."

In principle, answers to graphical queries could be found in a single view by first marking the queried objects, then changing the focus of the view and looking up whatever information is desired about the queried objects. There are two problems with this sequential approach: (1) The delay between query and response cuts down on the number of queries a user is willing to entertain; and (2) the Gestalt of the data in the query view is lost when the user looks up information in other views.

Recently, Shneiderman (1994) promoted the notion of interactive graphical data base query under the name "dynamic queries." Some of us have also interpreted brushing in multiple linked views as data base query (Buja et al. 1991, accompanying video). This interpretation joins two others that have been previously proposed. Here is a list:

- *Brushing as conditioning* (Becker and Cleveland 1987): Because brushing a scatterplot defines a clause of the form $(X, Y) \in B$, the selection is interpreted as conditioning on the variables X and Y. The idea is that, by varying the area B, such conditioning allows one to examine the dependence of other variables on the "independent" variables X and Y.
- *Brushing as sectioning* (Furnas and Buja 1994): Brushing projections with thin point-like or line-like brushes can be seen as forming geometric sections with hyperplanes in data space. From such sections the true dimensionality of the data can be inferred under certain conditions.
- *Brushing as data base query*: Brushing operations have an interpretation in terms of the logic of query clauses. For example, a simple placement of a rectangular brush defines a conjunction $a_1 < X < a_2$ and $b_1 < Y < b_2$; but accumulating cases by "painting" with a moving brush corresponds to forming the disjunction of all clauses traced out by the moving brush. The question arises then how flexible brushing is as a way of defining query clauses. The difficulty of formulating complex clauses has been a problem in the human interface of data base languages.

The major advantages of brushing multiple linked views over conventional data base queries with a language interface are the following:

1. The query is issued graphically based on visual information. There is, for example, no danger of placing an empty query: We correct the brush placement until the query "looks right."
2. The answer to the query is given graphically. In contrast to textual answers that tend to flood a screen if unexpectedly large queries are issued, graphical presentation has a much greater capacity for showing digestible information about large subsets of the data.

A problem with a brushing interface to data base query is that it may be difficult to

form certain clauses involving many variables. Because most brushing interfaces use an accumulating mode of painting, it is easy to form disjunctions (unions) of any kind, but it may be unintuitive to form conjunctions (intersections) involving many variables.

Certain problems of language interfaces are nonexistent in a graphical interface. For example, there is often no need for a complement operation, as long as the views show all the data. If one gets interested in the complement of a selected subset marked with red color, one simply looks for the cases that are not red. No additional query is necessary.

1.3 ARRANGING MANY VIEWS

The purpose of arrangement of views is to facilitate meaningful comparisons. An example is the scatterplot matrix, which arranges a potentially large number of pairwise variable plots in a sensible way. In general, we include here any tool for arranging large numbers of plots that have a semantic connection to each other.

Besides pairwise variable plots, arrangement techniques have proven useful for large collections of so-called conditional plots. Such plots show structured subsets that are essentially defined by combinations of levels of discrete or discretized variables. In a two-factor situation, for example, it is natural to arrange the conditional plots in a matrix arrangement such that cell (i, j) contains a plot of the subset defined by the ith level of the first factor and the jth level of the second factor. For more than two factors, one has to resort to nesting of some of the factors within others. These ideas are implemented in different ways in two sophisticated systems: TempleMVV (Mihalisin, Timlin, and Schwegler 1991), and Trellis (Becker and Cleveland in press). Conditional plots have an obvious interpretation in terms of data base queries. Each factor combination constitutes a data base query, and a collection of conditional plots arises from a collection of queries with factorial structure. Therefore, an arrangement of conditional plots is meant to facilitate graphical comparison of a collection of precomputed queries with special structure.

Another system in the category of arrangement techniques has been built by Eddy and Mockus (1995) in a very different context. Their aim is to provide an image browser for a large collection of images from NASA's Voyager mission. Their approach is to arrange stamp-sized renditions of images on a screen and provide intelligent rearrangement tools for visual clustering of images, for example.

Plot arrangement has not been recognized yet as a powerful concept in data visualization, at least not by a wider statistics audience. Yet the potential for ideas in this area is great. One visualization group at Bell Labs, for example, is experimenting with 3-D arrangements of conditional plots (Anupam, Dar, Leibfried and Petajan 1995).

2. XGOBI

We are going to demonstrate instances of the three categories of tools with the XGobi software. Most of the tools were explicitly built into XGobi, but others appeared serendipitously as afterthoughts through creative application of existing tools.

XGobi is a software system for high-dimensional data visualization with dynamic

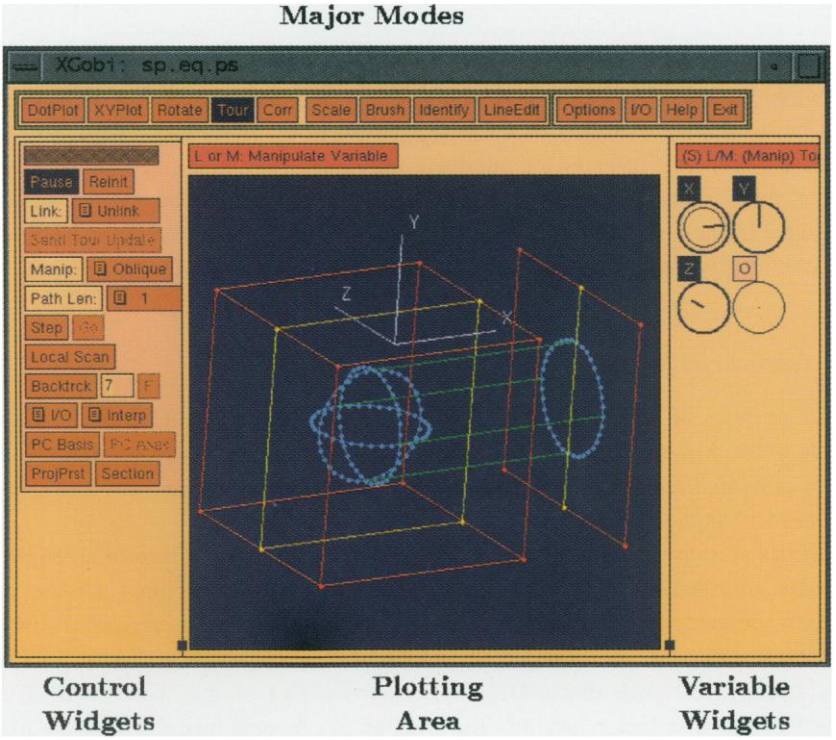

Figure 1. Example of an XGobi Window. The plotting area shows points and lines. The major modes, such as "DotPlot," "XYPlot," and "Tour" are accessible through buttons on the top bar. Subordinate to each major mode, the left side shows a number of buttons and sliders for controlling example animation speed. The right side has the so-called variable widgets, which are representations of the variables to indicate the relative position of the current projection plane with regard to the variable unit vectors. The tripod in the plotting area is redundant with the lines in the variable circles. Also, the variable boxes serve as an interface for variable selection and transformation. The gray bar at the very top is provided by the window system for window manipulations.

graphics, implemented in the X Window System (Trademark of MIT). XGobi's rendering is tailored to scatterplots—that is, cases are mapped to points on the screen. The cases are presented as simple glyphs, such as circles or crosses of various sizes and colors, and they can be labeled with text strings if desired. Lines of various colors can be drawn by specifying that certain pairs of points be connected.

An instance of XGobi is represented to the user by a window such as the one shown in Figure 1. The particular plot shown in this window is a reminder that, although conceived as a scatterplot instrument, XGobi can be used to draw surprisingly complex plots.

An XGobi window has four major components:

- In the center is a plotting area, showing a scatterplot or line drawing. This area is not passive; it can respond to mouse actions for labeling, brushing, panning, zooming, rotation.
- At the top is a horizontal bar of buttons for the main functions, such as X-Y

plots, 3-D rotation, and tours.

- On the left is a vertical stack of buttons and other controls that are specific to the current main function. For example, in the tour function this area contains among other things a slider for speed control and a button to initiate projection pursuit.
- On the right is a set of so-called variable widgets—one widget per variable—to select, deselect, and transform variables. In addition, the variable widgets indicate how each variable contributes to the current plot; a horizontal vector in the variable widget indicates the associated variable is plotted horizontally, for example.

In much of our following demonstrations we will cover up the controls on the left in order to reduce the visual clutter of the figures.

Each instance of XGobi runs in its *own process*, and multiple XGobi processes can be run simultaneously. They can be linked to each other under some minimal conditions on the data they present. *Linking of multiple windows* exists for various tasks:

- labeling;
- glyph brushing of points;
- color brushing of points;
- color brushing of lines;
- erase brushing; and
- tour motion.

In each case, linking means that a receiving window undergoes the same changes as a sending window. The receiver takes on the same labels, the same glyphs and point colors, the same line colors, the same erased points and lines, or the same high-dimensional projection as the sending window.

Another set of powerful tools in XGobi are dynamic projection mechanisms:

- 3-D rotations;
- grand tours; and
- correlation tours.

The 3-D rotation function provides a few standard rotation modes in 3-spaces spanned by variable triples. The grand tour produces oblique rotations in higher than 3-D; the correlation tour is a simplification of the grand tour whereby selected variables are either x-variables or y-variables, and 1-D projections of the y-variables are plotted against 1-D projections of the x-variables. In both tour functions, there are three control modes:

- random travel on a continuous path;
- real time projection pursuit; and
- manual projection control.

The last item, manual projection control, is a recent addition that has proven to be extremely powerful. Its implementation will be documented elsewhere. Many of the following demo images have been produced by making extensive use of manual controls.

Projection mechanisms are, of course, at the heart of the *focusing tools* available in XGobi.

This leaves the question of what the *arrangement tools* in XGobi are. The answer is that none are explicitly built into it. Yet there is a way to use (or abuse?) high-dimensional rotations to achieve intelligent arrangements of conditional plots. The idea is to create in a *single* XGobi window a view/projection that can be interpreted as a matrix arrangement of subsets of the data. The trick that makes this work is, unexpectedly, the use of discrete variables: Rotating discrete variables into a projection has the effect of grouping the points according to the levels of the variables. For details, see Section 5.

We proceed with a slew of demos that we hope will be more convincing than long-winded verbal explanations. Some of the demo data are new, some are stale, and some are artificial.

3. FOCUSING TOOLS FOR FINDING GESTALT

3.1 EXAMPLE 1: 3-D AND 4-D ROTATION—THE LASER DATA

The following data are courtesy of Paul Tukey who first analyzed them with his own 3-D viewing software. The data were obtained by Bellcore engineers in an experiment whose purpose was to establish the performance characteristics of a new laser. This laser was powered by two input currents. The output of the laser was a beam with a certain energy and wavelength. Energy and wavelength are functions of the two input currents. It was known that erratic behavior of the beam could arise for certain combinations of input currents because of quantum theoretical effects. The problem was finding areas of stable operation in terms of the two currents where energy and wavelength of the beam would show predictable and smooth behavior. The variables were: "current 1," "current 2" (the predictors), "energy," and "wavelength" (the responses). There were 64 measurements. Scatterplots of the response variables versus the predictor variables are shown in Figure 2.

Analyzing energy as a function of current 1 and current 2 first, we find that the marginal plots of energy against the individual currents are astonishingly uninformative: One would expect energy to be a monotone increasing function of both currents, but the indications of monotonicity from the plots in Figure 2 are weak. Perceptually the strongest structure arises from the fact that current 2 has been measured discretely at six equispaced levels.

One of the most efficient ways of analyzing these data is applying 3-D rotations to the space generated by current 1, current 2, and energy. The rotating image on a computer screen is striking. It suggests that most of the points lie on a concave response surface energy = f(current 1,current 2). Unfortunately, this is entirely lost in any still image, such as the one shown in the top left of Figure 3. If we enhance the plot with lines, however, the structure becomes somewhat apparent even in a static plot. In the top right plot of Figure 3, we take advantage of the six discrete levels of current 2 and trace a curve on the response surface by holding the value of current 2 fixed. The projections in the top row of Figure 3 are the same. The energy curves with constant values of current 2 indicate the overall concavity of the response surfaces. An interesting deviation from concavity is found in the second curve from the right—the kink, marked blue, in the upper part was interpreted by experts as a quantum theoretical effect. Less visible is an outlier, marked

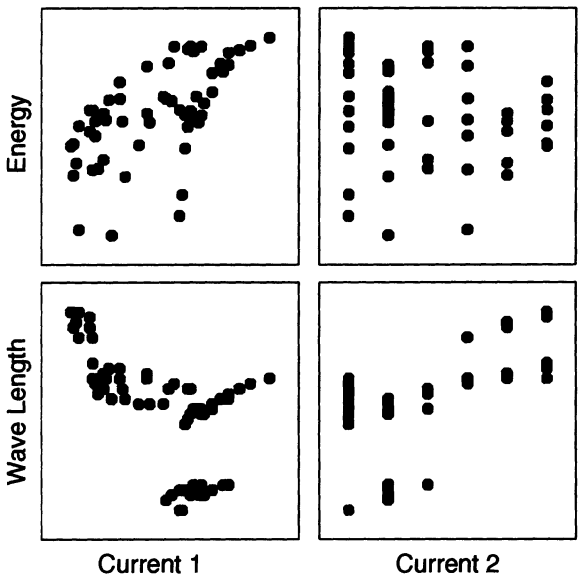

Figure 2. The Laser Data: Responses Plotted Against Predictors.

red, in the upper end of the second curve from the left. This outlier is elusive because it stands out in such a way that the concave surface surrounds it like the palm of a hand. This point does not seem to have an interpretation and is considered a bad observation.

We bring the remaining variable, wavelength, into the picture by literally rotating this variable into view. The result is a 2-D projection of the full 4-D data space, shown in the bottom row of Figure 3. A striking effect is the appearance of a jagged pattern in the second and third curve from the right. Because these jaggies must be due to wavelength, we just found wavelength instabilities. This is obviously an undesirable effect because, for these values of the input currents, it is impossible to predict the wavelength of the beam. Besides instabilities, we also find abrupt level changes of the wavelength; see the two leftmost as well as the rightmost curves, where the change is marked green.

This example should illustrate the fact that 2-D projections of 4-D data (often misleadingly called "4-D rotations") may actually be interpretable. At the end of the analysis we had all the answers the customer was interested in: finding areas of stable/unstable operation as a function of the two input currents.

We will revisit these data in a later section and re-analyze them with arrangements of conditional plots.

3.2 EXAMPLE 2: PROJECTIONS OF HIGH-DIMENSIONAL CUBES—A THEOREM BY DIACONIS AND FREEDMAN

In the previous section we had a direct glimpse of 4-D data. In this section we want to exercise our high-dimensional intuition by projecting some fairly intuitive objects, namely high-dimensional cubes, down to two dimensions. Figure 4 shows random projections of

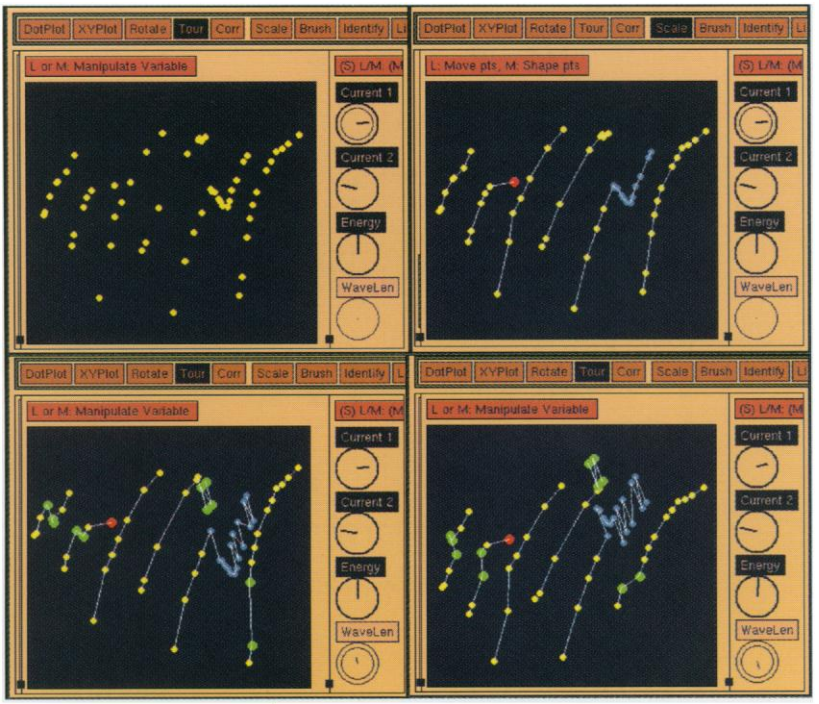

Figure 3. Rotations of the Laser Data in 3-D and 4-D. Top row: 3-D, in the space of "Current 1," "Current 2," and "Energy." With the connecting lines, the energy dip (blue) and the odd observation (red) are visible. Bottom row: 4-D with "Wavelength" rotated into the projection. The wavelength instabilities (blue and green) show up as jaggies. On the far left and far right are also change points in the wavelength (green).

the vertices of cubes of dimensions 3 through 9, respectively.

It is not too hard to gain an understanding of a 4-D cube. It has two opposite faces formed by 3-D cubes, in the same way as a 3-D cube has two opposite faces formed by 2-D cubes, that is, squares. Similarly, a 5-D cube has two opposite faces consisting of 4-D cubes, and so on. The number of ways in which opposite faces can be found equals the dimension. In a 3-D cube, we have three ways to find opposite faces, namely front-back, left-right, and up-down; in a 4-D cube we have four ways of finding opposite faces; and so on.

This is all fine, except that high-dimensional cubes get unwieldy quickly. With 2^p vertices in p dimensions, a cube as a multivariate data set can get quite large. In addition, a confusing effect sets in: Most random projections of the vertices of a high-dimensional cube look amazingly as if they were distributed according to a bivariate standard normal distribution. This is not coincidental! According to a theorem by Diaconis and Freedman (1984), most random projections of many types of objects show a point distribution that approaches a bivariate standard normal distribution as the dimension p goes to infinity. The sequence of random projections of cubes in Figure 4 bears this out visually. For a related discussion, see Cook et al. (1995, sec. 3.2.2).

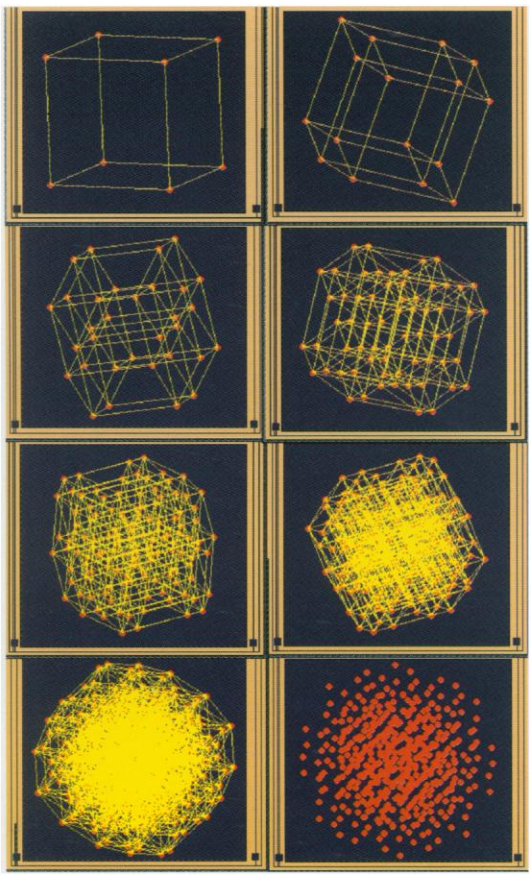

Figure 4. Cubes of Dimensions 3–9 in Eight XGobi Windows. The 9-D cube is shown with and without edges.

3.3 EXAMPLE 3: HAMPEL'S USE OF A 4-DIMENSIONAL CUBE IN AN EXPERIMENTAL DESIGN

Despite the experience of the previous section, higher-dimensional cubes can be handy objects for representing certain types of data. A case in point are data from factorial experiments. In particular, the observations of 2^p designs are in an obvious 1-to-1 correspondence with the vertices of a p-dimensional cube: In a 2^4 design, for example, the vertex with coordinates $(0, 1, 1, 0)$ corresponds to the observation with factors 2 and 3 at the high level and factors 1 and 4 at the low level. This fact inspired Frank Hampel (1970s, personal communication 1994) to re-analyze the 2^4 fermentation data of Johnson and Leone (1964, p. 187) with a novel graphical idea. [Note on the data: This is really a 2^5 design, but Johnson and Leone averaged the high and low levels of factor 3. Hence the labeling of the factors as x_1, x_2, x_4, and x_5.]

Hampel drew a specific projection of a 4-D cube, namely a set of four small squares arranged as vertices of a larger square; he then sheared the projected vertices in the 45-degree direction by moving them with an amount proportional to the response. The

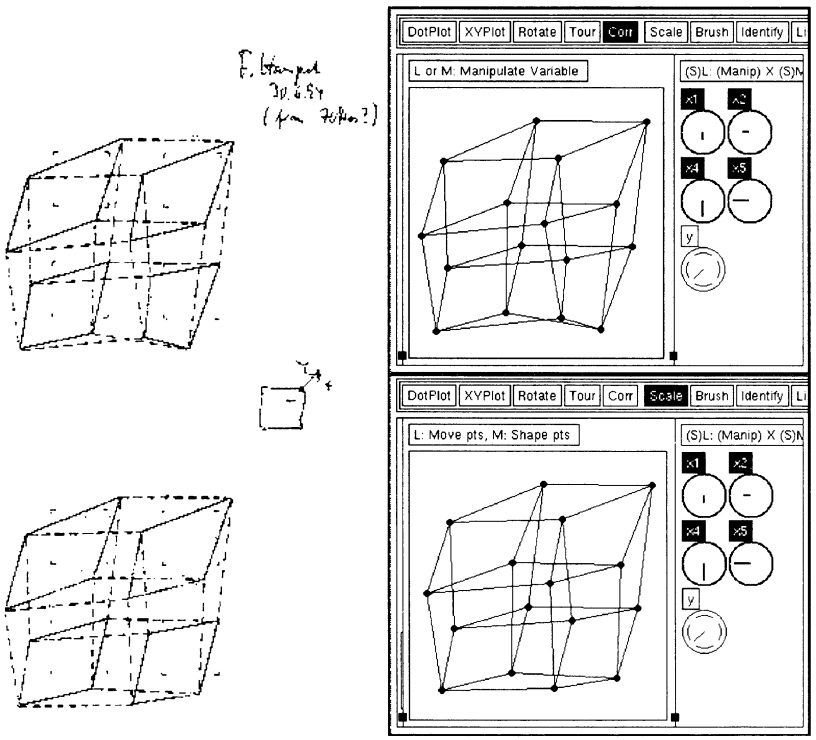

Figure 5. The Fermentation Data of Johnson and Leone (1964). Left: Hampel's 4-D cube for the 2^4 experiment. Right: Our reconstruction in XGobi. Top: The original data; the two points at the bottom right are confused. Bottom: The corrected data. (Drawings on left courtesy of F. Hampel.)

resulting graph, hand-drawn by Hampel, is shown in the top left of Figure 5. The projections of the vertices before shearing are marked with small angles. In the top right, we show our reconstruction of this plot in XGobi.

The point of the graph is to show that there exists a local interaction, as reflected by a crossing of lines at the bottom of the graph. It turns out that this local interaction could have been caused by a confusion of two values (cases 6 and 8 in Johnson and Leone's [1964] listing of the data, tab. 15.11, p. 187). The corrected data are shown in the bottom graph of Figure 5, again in Hampel's drawing, and in our reconstruction in XGobi; the offensive crossing has disappeared. There still exist some systematic deviations from parallelism, indicating the existence of interactions, but these are no longer of the flagrantly local kind. Hampel attributes the discovery of the confusion of two values to Cuthbert Daniel, who presented his analysis in a lecture at the University of California at Berkeley in 1968. Later, in the 1970s, Hampel reconstructed the analysis with his own tool, the sheared 4-D cube.

Hampel's idea behind this graph is the following: If the data have only main effects, then the shearing of the cube (in the 45-degree direction, say) with values proportional to the responses produces an image where all 2-D faces are parallelograms. This is easily seen by examining how a single main effect shears the cube. The two opposite 3-D

faces corresponding to the lower and the upper level of the factor are moved rigidly by differing amounts, leaving parallelograms intact. If some of the 2-D faces are no longer parallelograms, it must be due to interactions. The power of this method is that nonstandard localized interactions can be detected, such as the "interaction" caused by a confusion of two values.

Following are some details about our reconstruction of Hampel's graph in XGobi. As can be seen from the variable widgets on the right of Figure 5, we generated five variables, four of them being dummy codings of the four factors x_i $(i = 1, 2, 4, 5)$, the fifth variable being the response. The dummy variables generate the vertices of the 4-D cube. In recreating Hampel's sheared projection, we started with a square representing the projection of the cube on the factors x_5 and x_4, and rotating factors x_2 and x_1 slightly into the projection. The factors had to be projected to the left and down in order to match Hampel's drawing. Finally, shearing was achieved by partially rotating the response into the projection along the 45-degree direction to the left and down. The counterintuitive orientations of the variables indicate that Hampel held his 20-year-old drawing upside down when he added his handwritten signature and date in June of 1994. The little square in the middle right shows how he remembered doing the shearing.

4. LINKED VIEWS FOR GRAPHICAL QUERIES

4.1 EXAMPLE 1: LINKED SCATTERPLOT BRUSHING

This technique is standard by now. We illustrate it with an equally standard data set, namely the livability ratings of the 329 metropolitan areas of the U.S. as provided by the Rand McNally "Places Rated Almanac" (Boyer and Savageau 1985). The reason for using such worn-out data is that in our demos they still represent one of the most accessible examples to broad audiences. Some of our viewers suggested that this particular application should be sold as a "moving advisor" to folks who are planning to move to another part of the country. Linked brushing offers unbeatable ease for locating areas with favorable climate, crime rate, housing costs, and so forth, on the one hand, and querying geographical areas for their livability on the other hand. The data are not without idiosyncrasies, however, which only increases the charm of exploring this particular data set.

Of the nine livability criteria that make up the variables of this data set, we only look at two. In Figure 6 we show a plot of "housing costs" against "climate and terrain" on the left, and "latitude" against "longitude" with points roughly outlining a map of the USA on the right. A brush was used to color various interesting groups of metro areas in the left-hand plot. The linked colors in the map identify the metro areas geographically:

- Red: This is a cluster with exceptionally high climate ratings but high housing costs. The map shows them to be coastal California, which is not too unexpected.
- Green: This cluster has only slightly lower climate ratings but considerably lower housing costs. It is mostly a set of cities in the Pacific Northwest, where climate is mild but housing is much more affordable. Here is a first idiosyncrasy of the climate ratings: They do not measure cloud cover, the major unpopular feature of

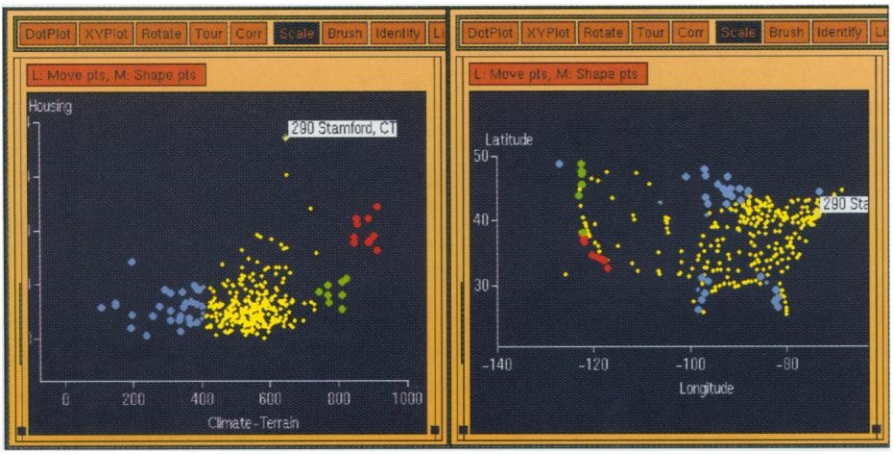

Figure 6. The Places Rated Data. Red is coastal California, green the Pacific Northwest, and blue the extreme climates.

the climate in the Pacific Northwest. The problem is not rain; contrary to common belief, Seattle has less precipitation than New York.

- Blue: The areas with unfavorable climate fall in two categories: The cities in southern Texas and the Gulf Coast of Florida are rated low because of hot summers, and the many places in the northern Midwest are rated low because of cold winters. The latter are joined by Anchorage, shown off the coast of Washington State, and a couple of places in Maine. Thus, the climate ratings measure extremes of temperature either way, hot or cold.
- A label was placed to identify the most expensive place in the U.S.: Stamford, Connecticut. Its label also appears in the map. Stamford as a metropolitan area has a reputation for being expensive, but it is also known to have poverty. Because the rating of housing costs seems to measure only the costs of home ownership, the costs of renting in low-cost areas are ignored.

4.2 EXAMPLE 2: LINKED TREE BRUSHING

This is a less common example of querying in multiple linked views. We illustrate it with the most common of all, namely Fisher's Iris data. The reason for this disingenuous choice of data is that we do not want to preempt a forthcoming publication by Koschat and Swayne (in press) for which this display was developed.

Figure 7 shows three views. The bottom is a plot of a hierarchical clustering tree. The colors reflect the true species: "Iris setosa," "Iris versicolor," and "Iris virginica."

We computed the tree in the S language with average-linkage clustering. We then wrote an S function which converted the clustering results to a data set that could be read into XGobi. This data set is of size $n + (n - 1) = 150 + 149 = 299$, because the original 150 points are joined by 149 points for the nonterminal branches of the tree. Our function did not only that; it also computed branch averages of all other variables

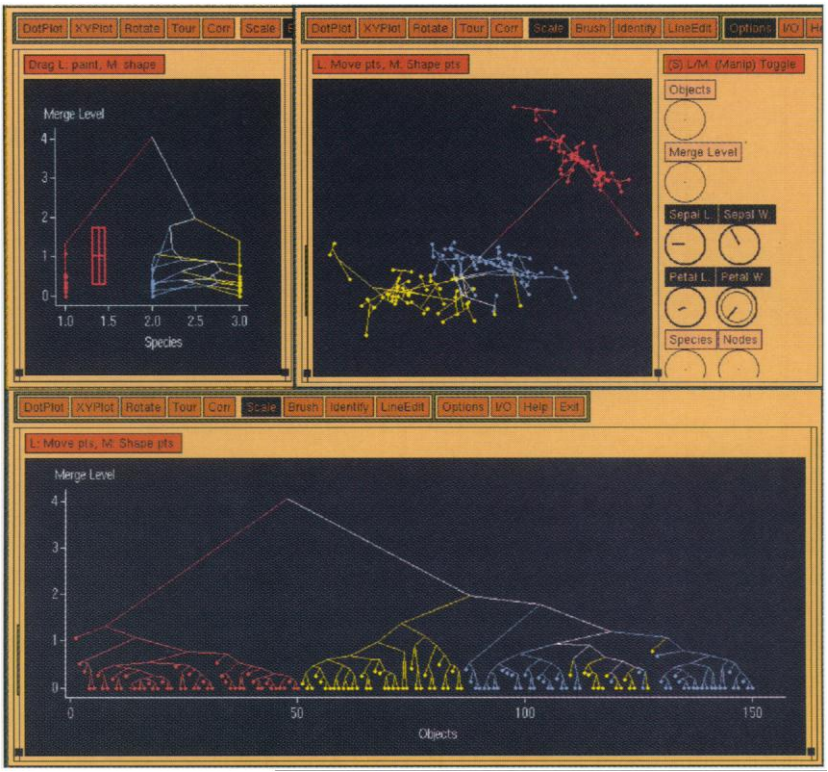

Figure 7. Fisher's Iris Data. Top left: The merge level of the tree is plotted against the true species, numbered from 1 to 3. Note that all variables, including "Species," have values not only for the observations but for the branching points of the tree as well. The red rectangle is the brush, which paints both points and lines. Top right: An oblique projection of the four length and width variables. Again, these variables have values for the branching points. Bottom: The hierarchical clustering tree, drawn as a plot of "Merge Level" against "Object" number.

for all the 149 nonterminal branches. This artifice made it possible to join the original multivariate data with the coordinates "object" and "merge level" that describe the tree layout.

We added two more variables: a variable "species" of three levels for the true class membership, and a variable "nodes" of two levels that distinguishes between real data points and artificial branch centers.

The result is a single data set from which three XGobis could generate the three linked views shown in Figure 7. In all views, we can inscribe the links that arise from the clustering tree, and we can smoothly rotate back and forth between views of the tree and projections of the original variables.

The top left plot of Figure 7 shows "merge level" plotted against true species membership ("grp"). This is the active plot that we used for brushing, and the red brush is still shown. The criss-crossing of tree-branches indicates that the blue and yellow species have some overlap, but not the red species. The top right plot shows an oblique projection of the four original variables, where the tree is inscribed in the manner explained previously.

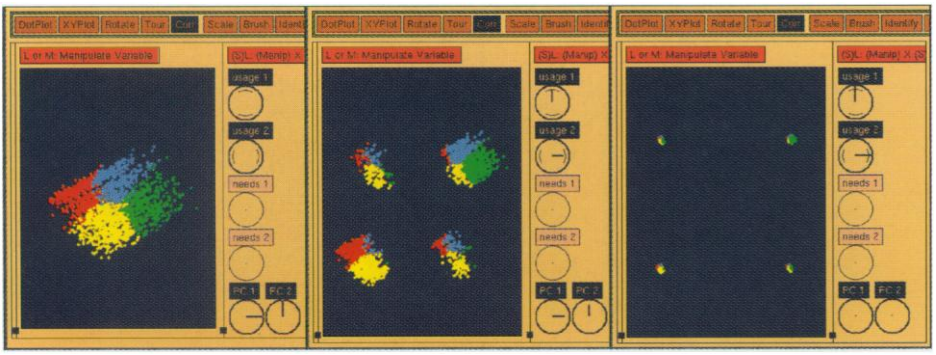

Figure 8. The Market Segmentation Data. Left: A plot of the first two principal components. Center: A projection halfway between the principal components plot and the four-point plot of the binary usage variables. Right: The four-point plot of the binary usage variables.

With a few interactive data rotations, it is easily possible to disentangle the messy area of overlap somewhat and establish a 1-to-1 correspondence with the groupings suggested by the clustering tree in the lower plot.

This tool is useful for teaching of hierarchical clustering methods and for presentation of clustering results. We refer to the Koschat and Swayne (in press) article for a fresh data example.

5. ARRANGEMENT TOOLS FOR VIEW COMPARISON

In this section we give a couple of examples for the use of matrix arrangements of conditional plots in XGobi. Unlike TempleMVV [Trademark Mihalisin Associates, Inc.] (Mihalisin, Timlin, and Schwegler 1991) and Trellis (Becker and Cleveland in press), arrangements of conditional plots have not been built into XGobi. Rather, matrix arrangements are created using high-dimensional data rotations. The reason why this can be done is best explained in terms of a generic example:

> *Consider two* continuous *variables X and Y and two* binary *variables U and V, making up a 4-D data set (X, Y, U, V). A plot of Y versus X will show the joint marginal distribution of (X, Y). See the left hand plot in Figure 8 for an example. By contrast, a plot of V versus U will show just four multiply overplotted points, corresponding to the four combinations of the binary levels of each variable. An example is in the right hand plot of Figure 8.*
>
> *Consider now a 4-D rotation that moves the (X, Y) plot half way to the (U, V) plot: The result is a plot that looks like the center of Figure 8. This is essentially a conditional plot of (X, Y) given the four pairs of values of (U, V) in turn.*

This trick can be easily executed in XGobi with manually controlled projections. It yields of course only a crude approximation to the elaborate conditional plots of TempleMVV and Trellis. For example, no axes and frames are provided, the flexibility of plot enhancement is limited, and unlike Trellis no overlapping "shingles" (a generalization of levels) are provided. On the plus side, however, manually controlled high-dimensional projections allow us to travel smoothly and rapidly between various

Figure 9. The Market Segmentation Data, in a (2 × 2) × (2 × 2) Arrangement. Horizontal: "Usage 2" nested within "Usage 1." Thus, the columns correspond left to right to the low-low, low-high, high-low, high-high levels of "Usage 1" and "Usage 2." Vertical: "Needs 2" nested within "Needs 1." The rows, from the bottom up, correspond to level combinations in the same order as the columns.

conditioning schemes. The user never thinks in terms of high-dimensional rotations but in terms of moving a discrete variable into the view.

One of the strengths of the TempleMVV and Trellis systems is their handling of more-than-two-way conditioning—that is, more than two discrete or discretized variables. The method they adopt is essentially mapping factorial to nested designs: If, for example, an additional binary variable W is given, one obtains a 4 × 2 matrix arrangement by nesting the levels of W within the levels of U. This is easily mimicked with 4-D rotations by moving more than 2 discrete variables into the view.

5.1 EXAMPLE 1: INTERPRETING A MARKETING SEGMENTATION ANALYSIS

The following data example is from a market segmentation study at AT&T. The goal of market segmentation is to divide customers into homogeneous groups based on information that is relevant for marketing. Because the market of the present study was in telecommunications, the following variables seemed relevant: telephone usage, telecommunications needs, and some auxiliary variables. The problem at hand was to understand the four segments produced by the k-means algorithm that was applied to these data.

The data are courtesy of D. Paul and N. Ruotolo at AT&T. We show in Figures 8

and 9 a subset of the variables, namely two binary variables that indicate high or low levels of two kinds of telephone usage, and two binary variables that indicate certain telecommunication needs. We also include the two largest principal components of the full data (containing many more variables).

The left-hand plot of Figure 8 shows the first two principal components. The colors represent the four segments found by the k-means algorithm. We note in passing that, contrary to common belief, k-means did not find clusters in the usual sense, simply because no clusters existed in these data. It found a sensible partition, though, which may be reasonable for segmentation purposes.

The center plot of Figure 8 shows a conditional plot of the principal components given the two usage variables: The large concentration of green and blue in the top right group indicates that the green and blue segments are largely high users, and similarly the red and yellow concentration in the bottom left indicates that these two segments are largely low users.

Figure 9 shows a conditional plot of the principal components given all four binary variables. Horizontally, we condition on the usage variables, vertically on the needs variables, where the second variable is nested within the first variable. In other words, the four columns indicate the groups corresponding to the low-low, low-high, high-low, high-high levels of the usage variables, and similarly the four rows from the bottom up indicate the respective level combinations for the needs variables.

From this 4×4 conditional plot we can easily read off the following interpretations of the segments:

- red: low on both usage variables and low on both needs variables,
- yellow: low on both usage variables, high on "needs 2,"
- blue: high on both usage variables, low on both needs variables,
- green: high on both usage variables, high on "needs 2."

With this conditional plot we were able to correlate two partitionings of the data. The first partition is given by the four segments and coded in color; the second partition is given by conditioning on four binary variables, producing 16 disjoint subsets that are laid out in a 4×4 matrix.

An alternative method for correlating the two partitionings would have been in terms of numerical tables of counts. We think, though, that the color-coded conditional plots present a much more immediate route to the qualitative features of the data.

The role of the principal components in this analysis was quite inessential; they were used mostly as a convenient jittering mechanism. A minor insight from their use is that the eight small groups in the two middle columns fall mostly along the boundaries of the segments. Thus, the "interior" cases that do not lie near the boundaries of the segments are more likely to follow the interpretations given above.

5.2 EXAMPLE 2: THE LASER DATA REVISITED

We have another look at the laser data of Section 3.1 by means of conditional plots. The goal of the exercise is to compare the curves on the energy and wavelength

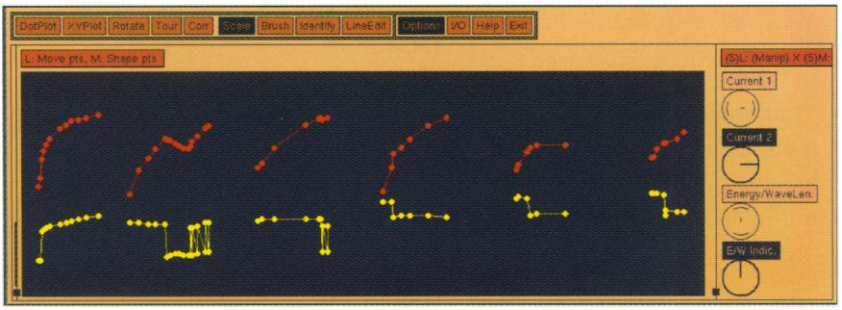

Figure 10. Conditional Plots of the Laser Data, Given the Levels of "Current 2." Red: Energy curves as a function of "Current 1." Yellow: Wavelength curves, featuring instabilities and change points.

surfaces traced by the discrete levels of current 2. To this end, we ran the data through a preprocessing step that allowed us to draw energy and wavelength curves on top of each other. The trick was to double the number of cases and to create a combined variable "energy/wavelen" whose values would be either standardized energies or standardized wavelengths, depending on whether a point belonged to an energy curve or a wavelength curve. In addition, we included a dummy variable to indicate whether a point was part of an energy or a wavelength curve.

The result after a simple 4-D rotation is shown in Figure 10. The top row contains the energy curves and the bottom row the wavelength curves, one per level of current 2, each as a function of current 1. Vertical comparison shows on what stretch of an energy curve a wavelength instability (jaggies) or a jump occurs.

As a microscope-like tool for reading detail off the curves, these conditional plots are superior to the data rotations of Section 3.1. In order to create them, though, a certain amount of knowledge about the data has to be assumed. As tools for first exploratory steps, the crude data rotations of Section 3.1 assume less on the analyst's part.

6. CONCLUSIONS

We attempted in this article to bring a system into today's research in interactive visualization of high-dimensional data. To this end, we proposed a triad of concepts that may help us understand the playing field a little better. These concepts cover three broad classes of view manipulations: (1) focusing single views; (2) linking multiple views; and (3) arranging many views. The tasks that are supported by these manipulations are (1) search for Gestalt; (2) graphical data base queries; and (3) graphical plot comparisons. Canonical examples in each category are, respectively, (1) projection controls; (2) linked scatterplot brushing; and (3) matrix arrangements of plots. We emphasize that these are *only* examples, albeit important ones.

In our data illustrations we hoped to show that the canonical examples—projecting, linking, and arranging—can be given some interesting twists:

- 4-D rotations of data can lead to unexpectedly interpretable findings, as in the

laser data. The same insights can always be found by other ways as well, but the effortless interpretability of 4-D rotations was surprising even to us who believe in the 4-dimensional.

- Who would have thought that asymptotic theorems for dimensions that go to infinity are relevant for data visualization? The disconcerting effect described by the Diaconis–Freedman theorem is certainly visible in nine dimensions.
- We found a piece of prehistory in Hampel's hand-drawn projection of a sheared 4-D cube, with which he graphically analyzed a 2^4 factorial experiment. We successfully reconstructed Hampel's drawing with computer bits.
- Although we think of our XGobi system as a scatterplot viewer, we twisted it into showing us a precomputed clustering tree, which was then linkable to projections of the data.
- Matrix arrangements of conditional plots can be mimicked by rotating discrete variables into a scatterplot. Thus, discrete variables can have a use in a scatterplot viewer!

The XGobi system, in which all these things can be done, is freely available from StatLib:

http://lib.stat.cmu.edu/general/XGobi/

[Received June 1995. Revised November 1995.]

REFERENCES

Andrews, D. F. (1972), "Plots of High-Dimensional Data," *Biometrics*, 28, 125–136.

Anupam, V., Dar, S., Leibfried, T., and Petajan, E. (1995), "DataSpace: 3-D Visualization of Large Databases," in *Proceedings of the IEEE Symposium on Information Visualization*, pp. 82–88.

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

Becker, R. A., and Cleveland, W. S. (1987), "Brushing Scatterplots," *Technometrics*, 29, 127–142. Also in *Dynamic Graphics for Statistics*, eds. W. S. Cleveland and M. E. McGill, Belmont, CA: Wadsworth, pp. 201–224.

——— (in press), "Trellis Displays," *Journal of Computational and Graphical Statistics*.

Boyer, R., and Savageau, D. (1985), *Places Rated Almanac*, Chicago: Rand McNally & Company.

Buja, A., and Asimov, D. (1986), "Grand Tour Methods: An Outline," in *Proceedings of the 17th Symposium on the Interface between Computer Science and Statistics*, pp. 63–67.

Buja, A., Asimov, D., Hurley, C., and McDonald, J. A. (1988), "Elements of a Viewing Pipeline for Data Analysis," in *Dynamic Graphics for Statistics*, eds. W. S. Cleveland and M. E. McGill, Belmont, CA: Wadsworth, pp. 277–308

Buja, A., McDonald, J. A., Michalak, J., and Stuetzle, W. (1991), "Interactive Data Visualization Using Focusing and Linking," in *Proceedings Visualization '91*, pp. 156–163.

Carr, D. B., and Nicholson, W. L. (1988), "Explor4: A Program for Exploring Four-Dimensional Data Using Stereo-Ray Glyphs, Dimensional Constraints, Rotations, and Masking," in *Dynamic Graphics for Statistics*, eds. W. S. Cleveland and M. McGill, Bellmont, CA: Wadsworth, pp. 309–329.

Chernoff, H. (1973), "The Use of Faces to Represent Points in k-Dimensional Space Graphically," *Journal of the American Statistical Association*, 68, 361–368.

Cook, D., Buja, A., Cabrera, J., and Hurley, C. (1995), "Grand Tour and Projection Pursuit," *Journal of Computational and Graphical Statistics*, 4, 155–172.

Diaconis, P., and Freedman, D. (1984), "Asymptotics of Graphical Projection Pursuit," *The Annals of Statistics*, 12, 793–815.

Eddy, W. F., and Mockus, A. (1995), "An Interactive Icon Index—Images of the Outer Planets," *Journal of Computational and Graphical Statistics*, 5, TK–TK.

Furnas, G. W., and Buja, A. (1994), "Prosection Views: Dimensional Inference Through Sections and Projections" (with discussion), *Journal of Computational and Graphical Statistics*, 3, 323–385.

Inselberg, A. (1985), "The Plane With Parallel Coordinates," in *The Visual Computer*, 1, New York: Springer Verlag, pp. 69–91.

Johnson, N. L., and Leone, F. C. (1964), *Statistics and Experimental Design* (vol. II,) New York: John Wiley.

Kleiner, B., and Hartigan, J. A. (1981), "Representing Points in Many Dimensions by Trees and Castles," *Journal of the American Statistical Association*, 76, 260–269.

Koschat, M. A., and Swayne, D. F. (in press), "Interactive Graphical Methods in the Analysis of Customer Panel Data," (with discussion) *Journal of Business and Economic Statistics*.

Hurley, C., and Buja, A. (1990), "Analyzing High-Dimensional Data with Motion Graphics," *SIAM Journal on Scientific and Statistical Computing*, 11, 1193–1211.

Mihalisin, T., Timlin, J., and Schwegler, J. (1991), "Visualization and Analysis of Multivariate Data: A Technique for all Fields," in *Proceedings Visualization '91*, pp. 171–178.

Newton, C. M. (1978), "Graphics: From Alpha to Omega in Data Analysis," in *Graphical Representation of Multivariate Data*, *Proceedings of the Symposium on Graphical Representation of Multivariate Data*, ed. P. C. C. Wang, New York: Academic Press, pp. 59–92.

Shneiderman, B. (1994), "Dynamic Queries for Visual Information Seeking," *IEEE Software*, 11, 70–77.

Swayne, D. F., Cook, D., and Buja, A. (1991), "XGobi: Interactive Dynamic Graphics in the X Window System with a Link to S," in *ASA Proceedings of the Section on Statistical Graphics*, pp. 1–8.

Tierney, L. (1990), *LISP-STAT, An Object-Oriented Environment for Statistics and Dynamic Graphics*, New York: Wiley.

Wegman, E. J. (1990), "Hyperdimensional Data Analysis Using Parallel Coordinates," *Journal of the American Statistical Association*, 85, 664–675.

Wegman, E. J., and Carr, E. B. (1993), "Statistical Graphics and Visualization," in *Handbook of Statistics*, 9, 857–958, ed. C. R. Rao, Amsterdam: Elsevier.