Exploring N-Dimensional Databases

Jeffrey LeBlanc, Matthew O. Ward

Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609

Norman Wittels

Civil Engineering Department
Worcester Polytechnic Institute
Worcester, MA 01609

Abstract

The ability of researchers in the scientific and engineering community to generate or acquire data far outstrips their ability to analyze it. This problem is even more pronounced when the data is of high dimensionality. Visualization has been identified as a critical technique for exploring data sets, but the visualization tools developed to date have mostly concentrated on the display of low (one to four) dimensional data. Ideally a tool for examining N-dimensional data should allow the presentation of the data in a way that can be intuitively interpreted and allow the display of arbitrary views and subsets of the data. The work presented in this paper describes the creation of such a tool using a technique which we term *dimensional stacking.*

1 Introduction

Modern scientific applications are generating larger and larger quantities of data at an ever increasing rate. There is a critical need for tools to aid in the display and interpretation of this data. Some applications generate values that are the result of a many-dimensional function. The display of 2-D, 3-D, and even 4-D data to aid understanding can be readily accomplished using currently available graphics techniques, but little progress has been made in developing graphical techniques for higher dimensional data.

When data is displayed, only 2 or 3 dimensions are generally shown at one time. With new advances in rendering technology, animation can be used to display a fourth dimension along the time axis, and techniques such as translucency [FUC85] can provide more insight into 3-D volumes. We are interested in displaying information of higher dimensionality. The display methods should be dynamic, efficient, and consider what elements of graphical display people best perceive [CLE85]. The traditional 2-D display works so well [TUF83] that we have attempted to exploit it analyzing higher dimensional data.

Researchers have been studying was to display dimensional data for some time. During the 1970s, the increased power available from computer graphics allowed the implementation of display and analysis methods for such data that were devised years earlier, but had been difficult to implement due a lack of sufficient computational power [FIE79]. In addition, new methods for data mapping and analysis began to be devised based on these prior methods [LEE77, TUK77].

One of the most traditional methods of displaying data is the simple coordinate plot: a single data point representing a value at some point in space. Friedman and Stuetzle researched the display of multidimensional data plots on a screen in ways directed towards statistical analysis [KOL82]. More recent work has dealt with producing 2-D scatterplots of "core" subsets of N-dimensional data [BEZ88].

Other display methods use symbols to represent data points in multi-dimensional space. Among the types of symbols used have been faces [CHE73] and trees [KLE81]. This type of iconographic representation has been expanded to use symbols juxtapositioned to create a "texture" map to display data [BER89, GRI89].

With the power of modern graphics hardware, it is possible to render very complex, and even animated, images on a screen [FAR87] to visualize multidimensional data. 3-D data can be rendered as realistic objects which can be manipulated as if they were physical objects using the latest interface devices [FEI90].

These techniques, while effective, suffer from certain drawbacks. While the interpretation of plots and the early symbol representations is fairly straightforward, the ratio of the amount of information represented to the space used to actually represent it (the data-ink ratio [TUF83]) is generally low. 3-D representations display a large amount of data, but by definition a solid model is a volume and not all parts can be viewed at once. A compromise between the two is needed: a method which displays data effectively without obscuring potentially important information.

A recent technique, which serves as a starting point for our work, assigns variables a "speed" and embeds one range of values within another[MIL90]. This has been used to display graphs of mathematical functions involving three or four variables using a 2-D plot. Color and orientation are used to differentiate the various dimensions. While adequate for these sparse function plots, this technique shares the shortcomings of the other plotting techniques in that the amount of information displayed is fairly small given the amount of screen space used.

The system we present uses the technique of *dimensional stacking*, described in detail under Development below, to "collapse" an N-dimension space down into a 2-D space and then render the values contained therein. Each value can then be represented as a pixel or rectangular region on a 2-D screen whose intensity corresponds to the data value at that point.

2 Purpose and Goals

In analyzing data we are typically interested in detecting such phenomena as local extrema, trends, discontinuities, anomalies, and correlations. For high dimensional data, this can be difficult without the proper tools to aid in the analysis. One should have the capability to examine N-dimensional data in a fashion that allows these features to become apparent. A graphical tool is the obvious way to do this since the human brain is well adapted to processing visual data in parallel and performing rapid qualitative pattern matching.

The primary goal of this research is to create a tool that enables the user to project data of arbitrary dimensions onto a 2-D image. Of equal importance to this is the ability to control the viewing parameters, so that one can interactively adjust what ranges of values each dimension takes and the form in which the dimensions are displayed. This will allow an intuitive feel for the data to be developed as the database is explored.

3 Development

In displaying N-dimensional data, we begins with several basic premises. The data to be displayed has some number of dimensions N. Each of these dimensions consists of some finite set of values (we can approximate ranges for infinite values with a finite set). Each point in this N-dimensional space possesses a value which falls within some range $< V_{min}, V_{max} >$.

A screen also has some finite number of distinct points, or pixels. In rendering information on a screen, each element of information may map into one or more pixels. The value of the element may be represented as a color and/or intensity value at each pixel, or perhaps as a geometric shape or screen position.

In traditional 3-D graphics, images are rendered by specifying viewing parameters and projecting (using parallel or perspective projection) the 3-D object onto a 2-D viewing surface. A view is generally defined by specifying a point to view the object from, a viewing direction, an orientation, and clipping parameters to define how much of the 3-D world is included in the image.

In our research we extend this strategy to N-dimensional information. Initially we have considered the display of parallel orthographic projections; that is, the view vector is either parallel or perpendicular with all dimensional axes. The view displayed for a data point is a grey scale determined by the relationship of the data value to the range of values it can take. This is currently performed by linear interpolation, but we are studying alternative mappings of values to intensities or colors. The view specification and clipping parameters are discussed below.

A technique which we term *dimensional stacking* is used to map the N-dimensional space into a 2-D space. Assume N dimensions $< D_1, D_2, ..D_n >$. Then consider a traditional 2-D display for the pair of dimensions $[D_1, D_2]$: each element of it represents a discrete value. If the cardinality of either dimension is less than the screen resolution, each element will map to a rectangular region of pixels. We can then use this area as a virtual screen for dimensions $[D_3, D_4]$, thus projecting 4-D data on a 2-D display. We can continue recursively stacking dimensions until the accumulated size exceeds the screen resolution. To clarify this process an example is provided below.

During this dimensional compression the dimensions can be seen as traveling at different "speeds" [MIL90], with the inner ones traveling "faster", that is repeating themselves more, than the outer ones. Thus, the two outer, or "slowest", dimensions divide the image into sections as determined by the cardinality of the dimensions. Then the dimensions that move a level "faster" divide up each of those sections into sub-sections. This process repeats for all the dimensions. For example, given a 4-D function whose dimensions (alternating "horizontal" and "vertical") have cardinality of 2, 3, 5, and 6, the slower dimensions could be stacked into a 2x3 display, each element of which is a 6x5 display. Many other combinations are possible.

A view is specified by giving each dimension a speed, or level, and a horizontal/vertical orientation. Clipping and/or zooming is accomplished by selecting the range of values to display for each dimension. Thus, for a dimension with values $< S_0, S_1, ..S_m >$, the user specifies a subrange of values to be displayed. Once an initial view is specified, the user can then change the ordering of dimensions and ranges of values interactively to produce different views.

If there are an odd number of dimensions we add in a "blank" dimension of cardinality 1 to balance the display. When this is treated as one of the slowest dimensions the display is effectively a row or column, with each section then divided according to the order of the other dimension. Since the capability to change the grouping of dimensions exists, the blank dimension can be swapped in to one of the faster dimensions, producing a striped effect.

The process for computing which value in N-space corresponds to a point (x, y) on the 2-D image is accomplished with the following algorithm:

1. For each dimension, calculate its ***stack cardinality***, which is the product of the cardinalities of the faster dimensions with the same orientation.

2. Divide the X coordinate value by the stack cardinality of the slowest horizontal dimension. This results in the index for that dimension.

3. The remainder is now divided by the stack cardinality of the next fastest horizontal dimension, giving its index.

4. Continue dividing the remainders by the stack cardinalities of successively faster horizontal dimensions until either the fastest dimension is reached or a zero remainder is produced. We now have indices for all horizontal dimensions.

5. Repeat the process using the Y coordinate and the vertical dimensions.

Given the 4-D data above, where the dimensions of cardinality (2, 3, 5, 6) are assigned the speed orientation (slow/horizontal, slow/vertical, fast/vertical, fast/horizontal), respectively, consider the image point (8, 4): dividing the X coordinate by the stack cardinality of the slowest horizontal dimension, 6, yields 1 and remainder 2. Since there are no faster dimensions, the process is complete, the X value 8 mapping into the horizontal dimensional values $\{1, 2\}$. A similar process for Y maps it into the values $\{0, 4\}$ of the vertical dimensions. The functional value of dimensions at location $\{1, 0, 2, 4\}$ is then fetched and mapped onto the display at the point (8, 4).

The implementation of the system was done in C and X-Windows on a DECstation 3100 workstation. Input to the program consists of a file containing the number of dimensions, their labeling, their range, a list of the distinct values each dimension can take, and the N-dimensional data to be displayed. The interface is a pull-down menu system. Options allow for the dynamic re-association of the speed and orientation of the dimensions, display of the current associations, selection of any subrange for each selection (including the fixing of a dimension to a single value), re-display of the data if viewing parameters are changed, and saving a view as an image file.

The dimension to speed/orientation assignment is performed by selecting the dimensions from a menu in the order, slowest to fastest, that the user wishes them to be displayed, with alternating horizontal and vertical orientations. To select a subrange for a dimension, the user again chooses the dimension from a menu listing and then indicates the minimum and maximum values from a sub-menu containing that dimension's possible values. Future enhancements will allow enhanced subset specification and reordering.

When displaying the data the system will scale the image to the largest size possible for the screen. If, even at a single pixel level, the range of the data is too large to be displayed, the system will generate an error message and ask that the view be respecified before attempting to redisplay. An alternative approach would be to create an image larger than the screen size and provide panning and shrinking operations. Tick marks are put along the sides corresponding to the dimension ranges to aid in the interpretation of the data; different colors and sizes of marks are associated with different dimensions and speeds.

4 Results

The system described above has proven to be a useful tool for visualizing N dimensional data. It was designed to display data in a simple yet versatile manner, and we feel that is precisely what it does. As an extension on the idea of the traditional 2-D plot, it is easy to interpret. Since the display of the data is done using a 2-D representation no information will be hidden, as can occur if a 3-D representation were used. The data-ink ratio [TUF83] is close to one, meaning there is very little wasted space for the amount of data displayed.

The following is an example of how the system functions. Shown in Figures 1 through 4 is a simple set of data with values monotonically increasing from 0.0 through 10.0. These values are normalized into a range of 0.0 to 255.0, thus mapping them to the grey levels of our system (the printer further reduces this to a range of 0 to 31). As in the example above, there are 4 dimensions with cardinality 2, 3, 5, and 6, respectively. If we think of this as a 4-D array, we assign the value at location

$$\{i, j, k, m\}$$

using the formula

$$1.0 + \frac{255.0 * counter}{C_i * C_j * C_k * C_m}$$

where *counter* is

$$i + j * C_i + k * C_i * C_j + m * C_i * C_j * C_k$$

and $\{C_i, C_j, C_k, C_m\}$ are the cardinalities of dimensions $\{i, j, k, m\}$, respectively.

Figure 1 shows the display with the dimensions of cardinality 2 and 3 as the slower dimensions and those of cardinality 5 and 6 as the faster dimensions. Note the tick lines which mark the changes in value of each dimension. Figure 2 is similar, but the range of the dimension of cardinality 5 is limited to a subrange of 3. Since the range is smaller, the image was scaled to screen size, displaying larger elements. Figure 3 is also similar to Figure 1, but the orientation of the slow dimensions has been reversed. Finally, Figure 4 shows what happens when the fast dimensions of Figure 1 are exchanged with the slow ones.

The following data provides a practical example of our technique. One of the problems being examined at the WPI Center for Image Understanding is the automated detection of surface distress (cracks) in pavement images [WIT90]. During the summer of 1989, a data base with over 21000 records was synthesized to model how a crack would appear given certain conditions of sunlight and pavement aggregate. In its most condensed form, the data base contains approximately 1 megabyte of raw data, a sizable amount to have to sift through looking for highs, lows, and trends.

The seven dimensions associated with this data base are the depth and width of the crack, Phi and Theta values indicating the position of the sun relative to the crack, the ratio of direct sunlight to ambient light, the reflectivity of the aggregate surrounding the crack, and the reflectivity of the material at the bottom of the crack. Since there are an odd number of dimensions, a blank dimension was added. The number generated using the values in these dimensions represents the contrast between the crack and the pavement surface. Values of the function approximately range from 0.0 (undetectable) to approximately 6.0 (easily visible).

Table 1 shows the cardinality and values of each dimension and their initial arrangement for display in Figure 5. The level indicates the speed of the dimension, with 1 being the fastest, or innermost, dimension and 4 being the slowest, or outermost. The totally black areas within the data are the results of certain reflectivity combinations that were not generated in the database. The lighter areas are places where the contrast between the crack and the surrounding aggregate are particularly high. By rearranging the ordering of the dimensions and changing the displayable ranges of values, these areas of high contrast can be grouped together, showing what values in the data would model a particularly visible crack (see Figure 6 and Table 2). This method is being used to optimize the design of lighting systems for automated pavement inspection.

5 Conclusions

We have presented a new and useful tool for the display and analysis of N-dimensional data based on a technique we term dimensional stacking. We believe the technique embodied by this system will make such analysis easier and more efficient than previous methods of data projection.

By experimentation with this system, an intuitive understanding was developed for the pavement crack data base as to what relationships exist between the various parameters and the values they describe. We feel that work with other sets of multi-dimensional data will yield similar results.

Future enhancements for the system will include the ability to specify a non-orthogonal view for the data, the ability to smoothly pan across the data, and rubber-band window zooming. An extension to include more of the standard 2-D image operations, such as thresholding and gradient techniques, will also be incorporated at a future date.

References

[BER89] Bergeron, R.D., and Grinstein, G., "*A Reference Model for the Visualization of Multidimensional Data*", Eurographics 89, 1989.

[BEZ88] Bezdek, J.C., and E.R. Chiou, "*Core Zone Scatterplots: A New Approach to Feature Extraction for Visual Displays*", Vision, Graphics, and Image Processing, 41, 1988, 186-209.

[CHE73] Chernoff, H., "*The Use of Faces to Represent Points in k-Dimensional Space Graphically*", Journal of the American Statistical Association, 76, June 1973, 361-368.

[CLE85] Cleveland, William, "*The Elements of Graphing Data*", Wadsworth Advanced Books and Software, 1985.

[FAR87] Farrel, E.J., "*Visual Interpretation of Complex Data*", IBM Systems Journal, 26, No. 2, 1987.

[FEI90] Feiner, S., and C. Beshers, "*Visualizing n-Dimensional Virtual Worlds with n-Vision*", Computer Graphics, Vol 24 number 2, March 1990.

[FIE79] Fienberg, S.E., "*Graphical Methods in Statistics*", American Statistician, 33, November 1979, 165-178.

[FUC85] Fuchs, H., et al, "*Fast Spheres, Shadows, Textures, Transparencies, and Image Enhancements in Pixel-Planes*", Computer Graphics, 19, July 1985.

[GRI89] Grinstein, G., R. M. Pickett, and M. G. Williams, "*EXVIS: An Exploratory Visualization Environment*", Graphics Interface '89.

[KLE81] Kleiner, B., and J.A. Hartigan, "*Representing Points in Many Dimensions by Trees and Castles*", Journal of the American Statistical Association, 76, June 1981, 260-269.

[KOL82] Kolate, Gina, "*Computer Graphics Comes to Statistics*", Science, 217, Sept. 1982, 919-920.

[LEE77] Lee, R.C.T., Slagle, J.R., and Blum, H., "*A Triangulation Method for Sequential Mapping of Points from N-Space to Two-Space*", IEEE Trans. Computers, March 1977, 288-292.

[MIL90] Milhalisin, T., "*Graphing in Multiple Dimensions with MGTS*", SunTech Journal, Winter 1990.

[TUF83] Tufte, E.R. "*The Visual Display of Quantitative Information*", Graphics Press, Cheshire, Connecticut, 1983.

[TUK77] Tukey, J.W., "*Exploratory Data Analysis*", Addison-Wesley, Reading, MA, 1977.

[WIT90] Wittels, N., and El-Korchi, T., "*The Visual Appearance of Surface Distress in PCC Pavement: Pt. II: Crack Luminance*", Transportation Research Record, in press.

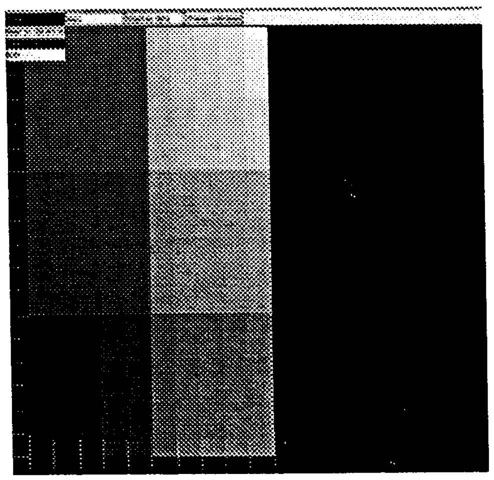

Figure 1

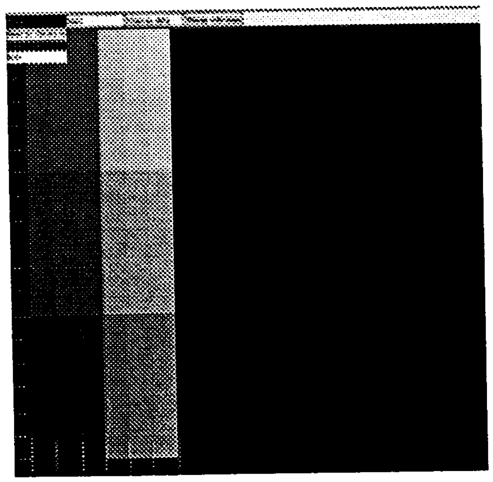

Figure 2

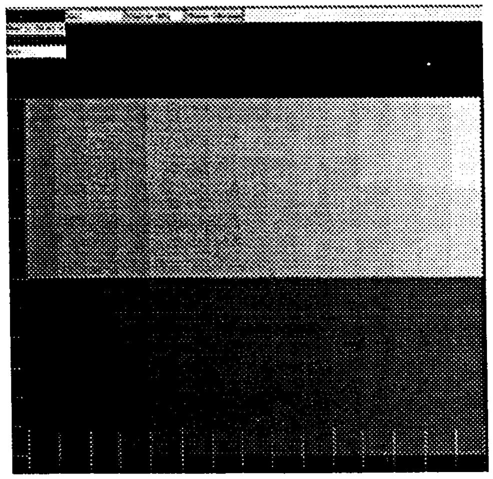

Figure 3

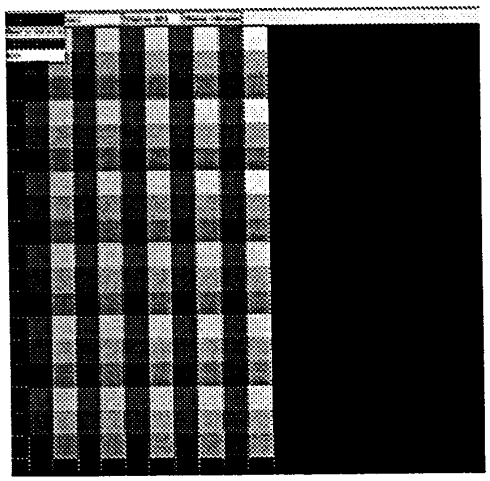

Figure 4

Figures 1 through 4 contain different views of the monotonically increasing function. Figure 1 shows the display with the dimensions of cardinality 2 and 3 as the slower dimensions (horizontal and vertical, respectively) and those of cardinality 5 and 6 as the faster dimensions. Figure 2 uses the same speed/orientation assignments, but the range of the dimension of cardinality 5 is limited to a subrange of 3. Figure 3 has the orientation of the slower dimensions reversed from their orientation in Figure 1. Finally, Figure 4 shows the image when the fast dimensions of Figure 1 are exchanged with the slow ones.

Table 1: Original speed/orientaton assignment and values for dimensions of Pavement Data

Level	Orientation	Name	Size	Real Values
1	Horizontal	Sun to Sky Ratio	6	$<\{0,1\},\{1,1\},\{1,2\},\{1,4\},\{1,8\},\{1,0\}>$
	Vertical	Reflectivity (crack)	10	$<\{30,30,30\},\{30,30,15\},\{30,30,45\},\{30,30,60\},\{30,15,30\},\{30,45,30\},\{30,60,30\},\{15,30,30\},\{45,30,30\},\{60,30,30\}$
2	Horizontal	Phi	6	$<0,15,30,45,60,75>$
	Vertical	Theta	5	$<0,30,45,60,90>$
3	Horizontal	Depth	3	$<10,15,30>$
	Vertical	Width	3	$<10,15,30>$
4	Horizontal	Reflectivity (aggregate)	2	$<15,30>$
	Vertical	Blank	1	$<0>$

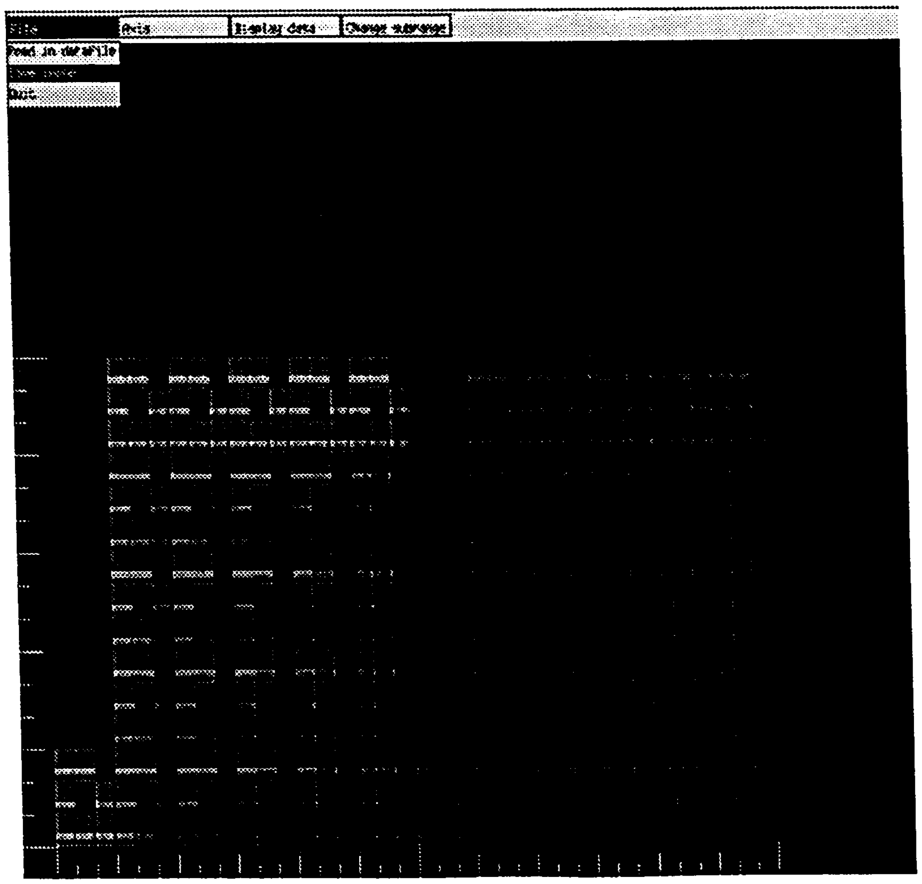

Figure 5: Image corresponding to Table 1

Table 2: Speed/orientation assignments and subranges producing strongest results

Level	Orientation	Name	Real Values
1	Horizontal	Sun/Sky Ratio	$< \{1,1\}, \{1,2\}, \{1,4\}, \{1,8\}, >$
	Vertical	Reflectivity (crack)	$< \{30,30,45\}, \{30,30,60\}, \{30,15,30\}$ $\{30,45,30\}, \{30,60,30\}, \{15,30,30\} >$
2	Horizontal	Depth	$< 10, 15, 30 >$ (Full range)
	Vertical	Width	$< 10, 15, 30 >$ (Full range)
3	Horizontal	Phi	$< 15, 30, 45, 60, 75 >$
	Vertical	Theta	$< 0, 30, 45, 60, 90 >$ (Full range)
4	Horizontal	Reflectivity (aggregate)	$< 15 >$
	Vertical	Blank	$< 0 >$

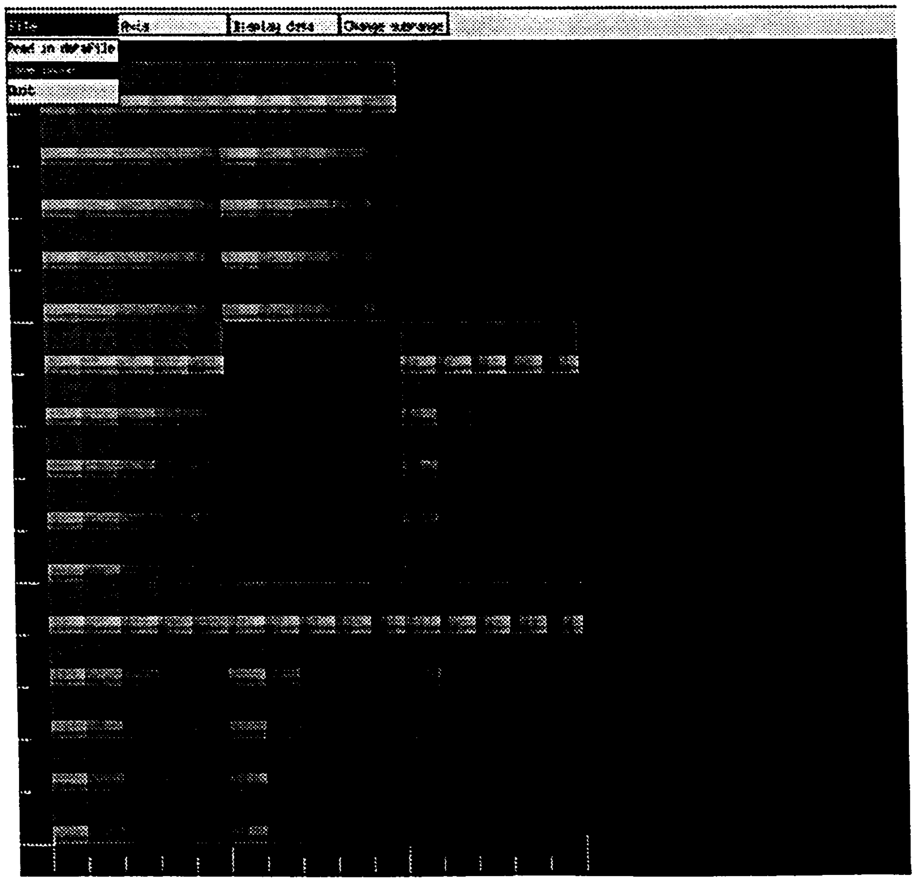

Figure 6: Image corresponding to Table 2