Visualization and Knowledge Discovery for High Dimensional Data

Alfred Inselberg[*] Multidimensional Graphs Ltd [†]
&
School of Mathematical Sciences
Tel Aviv University, Israel
aiisreal@math.tau.ac.il

Abstract

The goal here is to present a multidimensional visualization methodology and its applications to Visual and Automatic Knowledge Discovery in a coherent paper. Visualization provides *insight through images* and can be considered as a collection of application specific mappings:

$$ProblemDomain \longrightarrow VisualRange.$$

For the visualization of multivariate problems a multidimensional system of *Parallel coordinates* (abbr. ||-coords) is constructed which induces a one-to-one mapping between subsets of N-space and subsets of 2-space. The result is a rigorous methodology for doing and *seeing* N-dimensional geometry. We start with an overview of the mathematical foundations where it is seen that from the display of high-dimensional datasets the search for multivariate *relations* among the variables is transformed into a 2-D pattern recognition problem. This is the basis for the application to *Visual* Knowledge Discovery which is illustrated in the second part with real dataset of VLSI production. Then a recent geometric classifier is presented and applied to 3 real datasets. The results compared to those of 23 other classifiers have the least error. The algorithm, has quadratic computational complexity in the size and number of parameters, provides comprehensible and explicit rules, does *dimensionality selection* – where the minimal set of *original* variables re-

*Senior Fellow San Diego SuperComputing Center, California, USA

†36A Yehuda Halevy Street, Raanana 43556, Israel

quired to state the rule is found, and orders these variables so as to optimize the clarity of separation between the designated set and its complement.

Finally a simple *visual* economic model of a real country is constructed and analyzed in order to illustrate the special strength of ||-coords in modeling multivariate relations by means of hypersurfaces.

Do it in Parallel!

*I*n 1854 a cholera epidemic raging in London motivated Dr. J. Snow to search for clues. On a map of the neighborhood he placed dots at the locations of the recorded deaths. By a stroke of good fortune, the map also had the positions of the drinking water wells. The concentration of dots in the vicinity of just one of the wells was *visually* striking. He had the handle of the suspect well changed and the epidemic stopped! Apparently, the disease was being transmitted by contact with the handle. This true story is widely considered as an early success of visualization [22]; the picture provided insight which no one had gleaned from the tabular presentation of the data. Since nothing succeeds like success, others also chose to present and analyse their data visually. But They hit a wall: how to display data having many more than 2 variables? In due course lots of ingenious methodologies for visually encoding finite multivariate point sets were developed but having serious shortcomings (see [22] for a beautiful non-technical survey). By and large, they are laborious, lose information, have high representational complexity (which limits the number of variables that can be handled), can not represent *relations* among the variables, each variable may require different treatment etc. A *conceptual* breakthrough is needed.

The approach taken here is, in the spirit of Descartes, based on a coordinate system but differing in an important way. In geometry **parallelism**, which does not require a notion of angle, rather than orthogonality is the more fundamental concept. This, coupled with the fact that orthogonality "uses-up" the plane very fast, was the inspiration in 1959 for "Parallel Coordinates" whose systematic development began in 1977 (see [8] for a recent review). The goal is the visualization of multidimensional geometry and multivariate problems **without loss of information.**

In the Euclidean plane R^2 with xy-Cartesian coordinates, N copies of the real line labeled $\bar{X}_1$, $\bar{X}_2$, ..., $\bar{X}_N$ are placed equidistant and perpendicular to the x-axis. They are the axis of the **Parallel Coordinate** system for R^N all having the same positive orientation as the y-axis. A point C with coordinates $(c_1,c_2,...,c_N)$ is represented by the

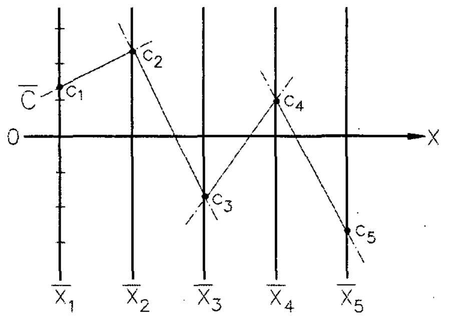

Figure 1: Polygonal line $\bar{C}$ represents the point $C = (c_1, c_2, c_3, c_4, c_5)$.

complete polygonal line $\bar{C}$ (i.e. the lines of which only the segments between the axes are usually shown) whose N vertices are at $(i-1, c_i)$ on the $\bar{X}_i$-axis for $i = 1, \ldots, N$ as shown in Fig. 1. In

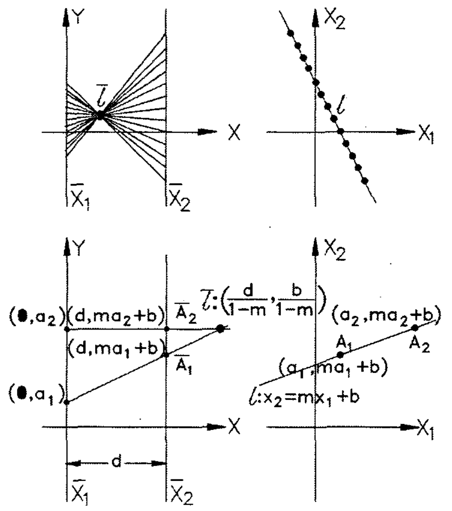

Figure 2: In 2-D parallel coordinates induce a point $\longleftrightarrow$ line duality.

this way, a 1-1 correspondence between points in R^N and planar polygonal lines with vertices on the parallel axes is established.

At first a relation, typically involving infinitely many points, is represented by the *envelope* [3] of the corresponding infinite family of polygonal lines representing the points of the relation. Let's look at the situation in 2-D where a point on the plane is represented by a segment between the x_1 and x_2 axis and, in fact, by the *whole line* containing the segment. In Fig. 2, the distance between the parallel axes is d. The line

$$l : x_2 = mx_1 + b, \qquad (1)$$

is a collection of points A. They are represented by the infinite collection of lines $\bar{A}$ (the image of a set F in ||-coords is denoted by $\bar{F}$) on the xy plane which when $m \neq 1$ intersect at the *point* with xy-coordinates:

$$\bar{l} : (\frac{d}{1-m}, \frac{b}{1-m}). \qquad (2)$$

This point represents the linear *relation*, Equation (1), and is the *envelope* of the family of lines $\bar{A}$. The two parameters m and b specify completely both l and $\bar{l}$. So in 2-D ||-coords induce a *Point* $\rightleftharpoons$ *Line* duality (i.e. mapping points into lines and vice versa). But there is a "little problem" when $m = 1$. This is because dualities properly reside in the *Projective* P^2 [5] and not in the

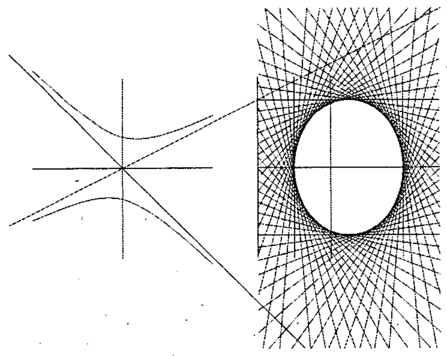

Figure 3: Hyperbola (point-curve) $\rightarrow$ Ellipse (line-curve).

Euclidean plane. As $m \rightarrow 1$ the point $\bar{l} \rightarrow \infty$ in the direction with slope b/d. So a line with $m = 1$ is mapped into a *direction* which provides the full information about the line. That is, lines with $m = 1$ are mapped into the *ideal points* of P^2.

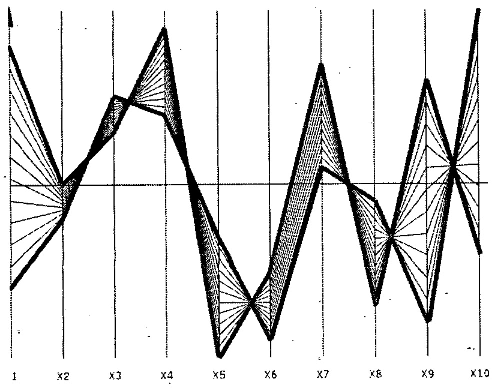

Figure 4: Line Interval in 10-D. Heavier polygonal lines represent the endpoints.

The role of the envelopes is seen clearly in finding the image(representation) $\bar{r}$ of a curve r. The image is obtained as the *envelope* of the lines representing the points of r. We distinguish by referring to the original curves as *point-curves* and their images as *line-curves*. In Fig. 3 we see one of the 6 ways that conics(point-curves) map into conics(line-curves). Actually, this is a special case of a much more general result [10] involving convex sets. If you are still with us, congratulations. You are certifiably brave and perhaps incorrigibly masochistic. But we've only just begun!

Indexing

Dualities between points and lines do not usually generalize nicely in N-space but they do in ||-coords. Consider a line l in N-Dimensional space described by

$$l_{i-1,i} : x_i = m_i x_{i-1} + b_i, \qquad (3)$$

for i=2,...,N. In the $x_{i-1}x_i$-plane the relation labeled $l_{i-1,i}$ is a line in 2-D and therefore by Equation (2), is represented by the point:

$$\bar{l}_{i-1,i} : \{\frac{(i-2)(1-m_i)+1}{1-m_i}, \frac{b_i}{1-m_i}\}, \qquad (4)$$

where, the distance between adjacent axes is taken as 1. So a line in N-D can be uniquely represented by N-1 such points one for each pair $(i-1, i)$.

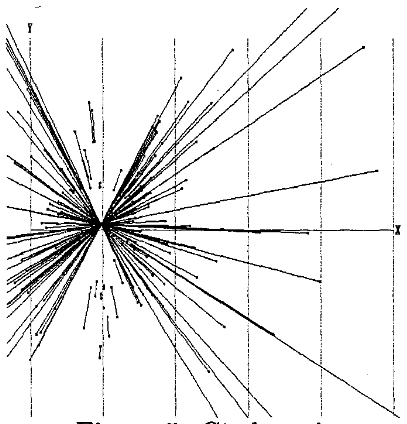

Figure 5: Coplanarity.

A polygonal line passing through all these N-1 points necessarily represents a point on the line l since the pair of y-coordinates a_{i-1}, a_i of its vertices simultaneously satisfy Equation (3) for $i = 2,...,N$. In Fig. 4 several polygonal lines representing points on an *interval* of a line in 10-D are shown. The *indexing* of the points in Equation (4) *is an essential part of the representation.* Since the display space is at a premium the indexing is usually not included in the picture. But in principle it must be accessible from some database. A number of construction algorithms based on this representation, *and crucially depending on the indexing*, have been found including one for obtaining the minimum distance between two lines and a Collision Avoidance algorithm for Air Traffic control [9].

Recursion

It turns out that the three points $\bar{l}_{i,j}, \bar{l}_{j,k}, \bar{l}_{i,k}$ are always collinear. Let's denote by $\bar{L}$ the line on which the three points are on. This property is essential for the representation of higher dimensional p-flats in N-space (that is planes of dimension $2 \leq p \leq N-1$). Let's illustrate it by taking a plane π_2 in 3-D. For a line $l \subset \pi_2$ the points $\bar{l}_{1,2}, \bar{l}_{2,3}, \bar{l}_{1,3}$ are obtained, which by the 3pt collinearity property yield a line $\bar{L}$. Similarly for another line $l' \subset \pi_2$ the corresponding line $\bar{L}'$ is obtained. The point of intersection, $\bar{\pi}_{1,2,3} = \bar{L} \cap \bar{L}'$, has the remarkable property that for any other line $l'' \subset \pi_2$ the corresponding line $\bar{L}''$ also intersects at $\bar{\pi}_{1,2,3}$. This condition characterizes *coplanarity* and is shown in Fig. 5. It can be shown in this way that π_2 can be represented by two points with *3 indices* (since the

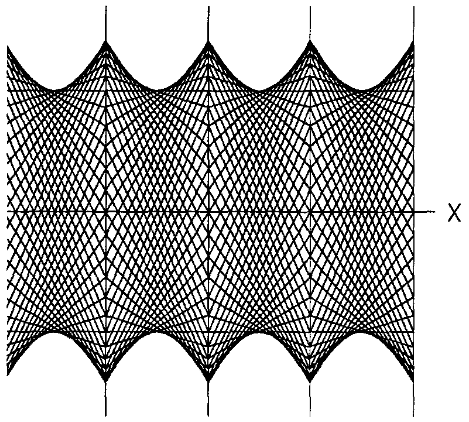

Figure 6: A Sphere in 5-D

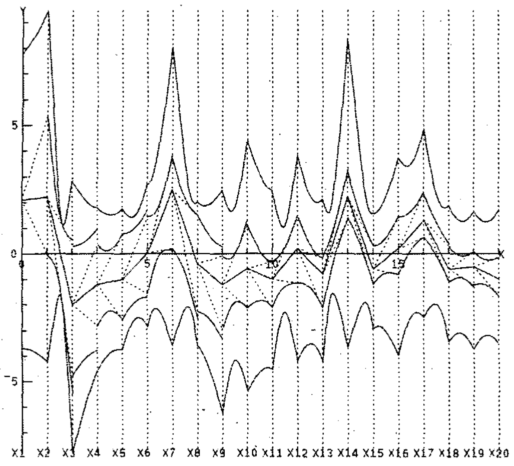

Figure 7: A Convex Hypersurface in 20-D and a point constructed with the interior point algorithm.

plane is described by a linear equation in 3 variables) and which distinguish them from two points with *two indices* representing a line in 3-D.

To recoup, we started with points (which are 0-flats i.e. 0-dimensional) on a 2-flat π_2 then constructed lines (1-flats) on π_2 and from them obtained the representation of the 2-flat still in terms of points. This outlines the recursive (on the dimensionality) construction which enables the representation of higher dimensional objects (see Eickemeyer[6]).

Polytopes & Hypersurfaces

A multidimensional object, represented in ||-coords, can still be recognized after it has been acted on by *projective transformation* (i.e. translation, rotation, scaling and perspective). Using Eickemeyer's representation, convex polytopes in N-D can be directly visualized (Chatterjee [4]).

Certain classes of smooth hypersurfaces can be represented and visualized. see Fig. 6, and there is an efficient and widely applicable algorithm for finding and *displaying* interior points see Fig. 7. In short, the representation in ||-coords of non-trivial multidimensional objects can be accomplished without loss of information, and provides *visible features* revealing the geometrical properties of the object being represented.

Visual Data Mining – A Case Study

We saw that parallel coordinates transform multivariate relations into 2-D patterns, a property that is well suited for visual data exploration and analysis. For this reason several software (including commercial) tools (i.e. VisuLab(Hinterberger[18]), VisDB(Keim [14]), Xmdv(Ward[15]), XGobi (Swayne, Cook, Buja, [20]), Parallax (Avidan [1]) etc include ||-coords. This type of application hinges on :

- an informative display of the data,
- good choice of queries, and

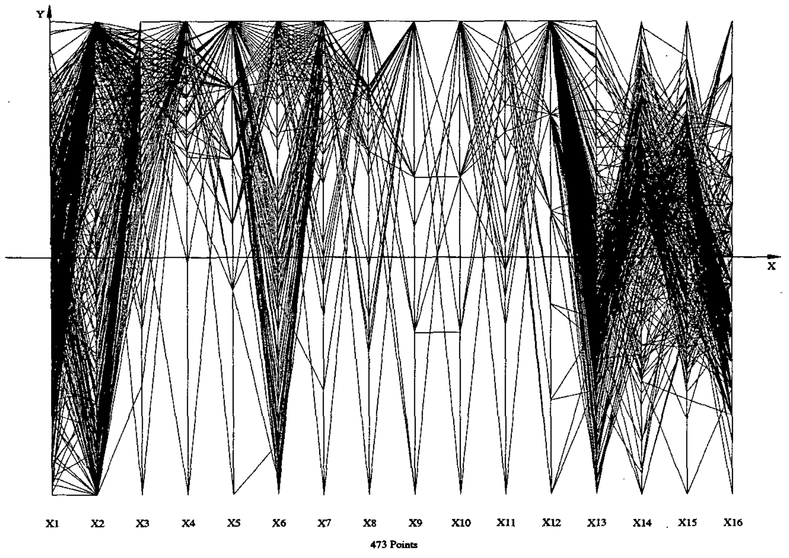

Figure 8: The full dataset

- skillful *interaction* by the user of the display in search of patterns corresponding to relationships among the variables in the data.

To appreciate all this an actual case study is described. The dataset, displayed in Fig. 8, consists of production data of several batches of a specific VLSI chip with measurements of 16 parameters involved in the process. The parameters are denoted by $X1, X2, \dots, X16$. The *yield*, as the % of useful chips produced in the batch, is denoted by $X1$, and $X2$ is a measure of the *quality* (given in terms of speed performance) of the batch. Ten different categories of *defects* are monitored and the variables' scales of $X3$ through $X12$ are inverted so that 0 (zero) amount appears at the top and increasing amounts appear proportionately lower. The remaining $X13$ through $X16$ denote some physical parameters.

Since the goal here is to raise the yield, $X1$, while maintaining high quality, $X2$, we have a case of multi-objective optimization due to the presence of more than one objective. The production specialists believed that it was the presence of defects which prevented high yields and qualities. So their purpose in life was to keep on pushing – at considerable cost and effort – for *zero defects*.

With this in mind the result of our first query is shown in Fig. 9 where the batches having the highest $X1$ and $X2$ have been isolated. This in an attempt to obtain *clues*; and two real good ones came forth. Notice the resulting range of $X15$ where there is a significant separation into two clusters. As it turns out, this gap yielded important insight into the physics of the problem. The other clue is almost hidden. A careful comparison – and here interactivity of the software is essential – between Fig. 8 and Fig. 9 shows that some batches which were high in $X3$ (i.e. due to the inverted scale low in that defect) *were not included in the selected subset.* That casts some doubt into the belief that zero defects are the panacea and motivates the next query where we search for batches having zero defects in at least 9 (excluding $X3$ where we saw that there are problems) out of the 10 categories. The result is shown in Fig. 10 and is a shocker. There are 9 such batches and *all of them have poor yields and for the most part also low quality*! That this was not questioned

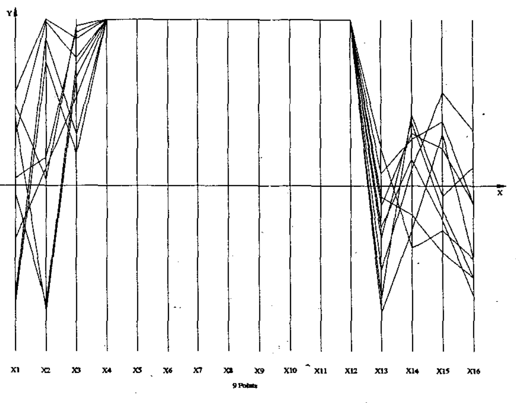

Figure 10: The batches with zero in 9 out of ten defect types.

and discovered earlier is surprising. We scrutinize the original picture Fig. 8 for visual cues relevant to our objectives and our findings so far. And ... there is

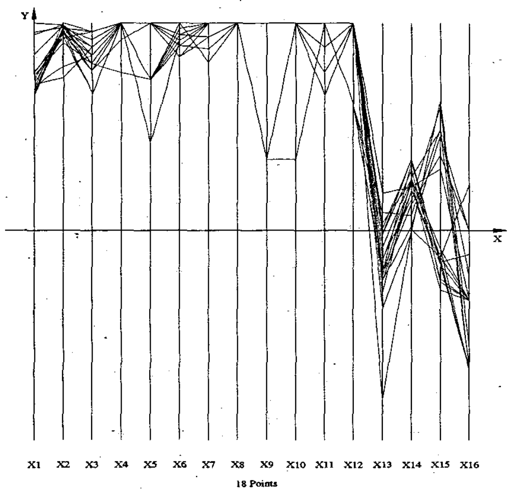

Figure 9: The batches high in Yield, $X1$, and Quality, $X2$.

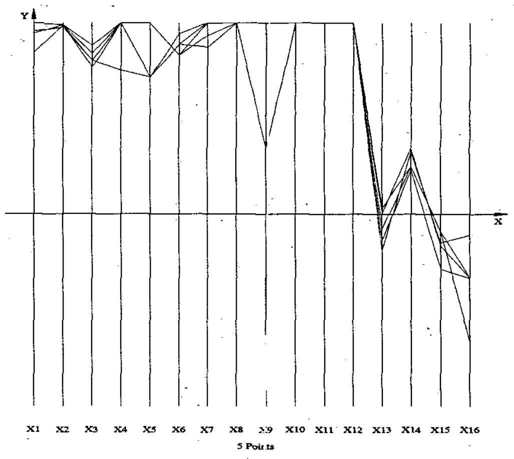

Figure 11: The batches with the highest Yields do not have the lowest defects of type $X3$ and $X6$.

one staring us in the face! Among the 10 defects $X3$ through $X12$ whatever $X6$ is, it's graph is very different than the others. It shows that the process is much more sensitive to variations in $X6$ than the others. For this reason, we chose to treat $X6$ differently and remove its zero defect constraint. This query (not shown) showed that the very best batch (i.e. highest yield with very high quality) *does not have zeros (or the lowest values) for* $X3$ *and* $X6$; a most "heretical" finding. It was confirmed by the next query which isolated the cluster of batches with the top yields (note the gap in $X1$ between them and the remaining batches). These are shown in Fig. 11 and they confirm that small amounts (the ranges can be clearly delimited) of $X3$ and $X6$ type defects are essential for high yields and quality.

Returning to the subset of data which best satisfied the objectives, Fig. 9 in order to explore the gap in the range of $X15$, we found that the cluster with the high range of $X15$ gives the lowest (of the high) yields $X1$, and worse it does not give *consistently* high quality $X2$, whereas the cluster corresponding to the lower range has the higher qualities and the full range of the high yield. It is evident that the small ranges of $X3$, $X6$ close to (but not equal to) zero, together with the short (lower) range of $X15$ provide *necessary* conditions for obtaining high yields and quality. This is also indicated in Fig. 11. By a stroke of good luck these 3 can also be checked *early* in the process avoiding the need of "throwing good money after bad" (i.e. by continuing the production of a batch whose values of $X3$, $X6$ and $X15$ are not in the small "good" ranges we have found).

These findings were significant and differed from those found with other methods for statistical process control[2]. This approach has been successfully used in a wide variety of applications from the manufacture of printed circuit boards, PVC and Manganese production, financial data, determining "skill profiles" (i.e. as in drivers, pilots), etc.

Automation

Though it is fun to undertake this type of exploration, the level of skill and patience required tends to limit the number of effective users. It is not surprising then that the most persistent requests and admonitions have been for tools which, at least partially, automate the knowledge discovery process. Hence the motivation for the, recently found [11], geometric algorithm which is presented next.

Classification is a basic task in data analysis and pattern recognition and an algorithm accomplishing it is called a **Classifier** [13], [19], [21]. The input is a dataset P and a designated subset S. The output is a characterization, that is a set of conditions or rules, to distinguish elements of S from all other members of P the "global" dataset. The output may also be that there is insufficient information in the dataset to provide the desired distinction.

With parallel coordinates a dataset P with N variables is transformed into a set of points in N-dimensional space. In this setting, the designated subset S can be described by means of a hypersurface which encloses just the points of S. In practical situations the strict enclosure requirement is dropped and some points of S may be omitted ("false negatives"), and some points of $P - S$ are allowed ("false positives") in the hypersurface. The description of such a hypersurface is equivalent to the rule for identifying, within some acceptable error, the elements of S. This is the *geometrical* basis for the classifier presented here. The algorithm accomplishing this entails:

- use of an efficient "wrapping" algorithm to enclose the points of S in a hypersurface S_1 containing S and typically also some points of $P - S$; so $S \subset S_1$, of course such an S_1 is not unique [1],
- the points in $(P - S) \cap S_1$ are isolated and the wrapping algorithm is applied to enclose them, and usually also a few points of S_1, producing a new hypersurface S_2 with $S \supset (S_1 - S_2)$,
- the points in S not included in $S_1 - S_2$ are next marked for input to the wrapping algorithm, a new hypersurface S_3 is produced containing these points as well as some other points in $P - (S_1 - S_2)$ resulting in $S \subset (S_1 - S_2) \cup S_3$,
- the process is repeated alternatively producing upper and lower containment bounds for S; termination occurs when an error criterion (which can be user specified) is satisfied or when convergence is not achieved.

Basically, the "wrapping" algorithm is a fast way of producing a hypersurface enclosing tightly a given point set. The kind of surface produced is a convex-hull approximation. The efficiency of the version implemented here is due to the use of the ||-coords representations of N-dimensional objects applied in the description of the resulting hypersurface [8]. To summarize, initially the wrapping S_1 encloses all the points of $S = S_0$. Then in the attempt to remove all extraneous points a *cavity* is created by the subsequent wrapping. Such cavities are generically denoted by S_{2n} for $n = 1, 2, \ldots$. Usually some of the

[1]To avoid unnecessary verbiage by a statement $S_j \subset S_k$ we also mean that the set of points enclosed in the hypersurface S_j is contained in the set of points enclosed by the hypersurface S_k.

points of S are enclosed in S_{2n}, so a correction follows with a S_{2n+1}, the hypersurfaces with odd subscript, which enclose and add these points to the previous approximation for the enclosure of S. Such a correction may also add some points of $P - S$ which need to be subsequently removed, or better reduced, to provide an increasingly tighter bound. So the process entails bounding the designated set S alternately from above and below providing, in case of convergence, an increasingly better approximation for S. It can and does happen that the process does not converge when P does not contain sufficient information to characterize S. It may also happen that S is so "porous" (i.e. sponge-like) that an inordinate number of iterations are required.

At step r the output is the description of the set S_r which consists of:

- a list of the minimum number of variables needed to describe S *without loss of information.* Unlike other methods, like the Principal Component Analysis (PCA), the classifier discards only the redundant variables. It is important to clarify this point. A subset S of a multidimensional set P is not necessarily of the same dimensionality as P. So the classifier finds the dimensionality of S in terms of the original variables and retains only those describing S. That is, it finds the *basis* in the mathematical sense of the smallest subspace containing S, or more precisely the current approximation for it. This basis is the minimal set M_r of variables needed to describe S. We call this dimensionality **selection** to distinguish it from dimensionality *reduction* which is usually done *with* loss of information. Retaining the original variables is important in the applications where the domain experts have developed intuition about the variables they measure. The classifier presents M_r *ordered according to a criterion which optimizes the clarity of separation.* This may be appreciated with the example provided in the attached figure, in addition,

- the current approximation of the rule stated in terms of conditions on the variables M_r, which constitutes the description of the current hypersurface, is obtained.

So on convergence, say at step $2n$, the description of S is provided as :

$$S \approx (S_1 - S_2) \cup (S_3 - S_4) \cup \ldots \cup (S_{2n-1} - S_{2n})$$

this being the terminating expression resulting from the algorithm.

The implementation allows the user to select a subset of the available variables and restrict the rule generation to these variables. In certain applications, as in process control, not all variables can be controlled and hence it would be useful to have a rule involving such variables that are "accessible" in a meaningful way. There are also two options available :

- either minimize the number of variables used in the rule, or
- minimize the number of steps, in terms of the unions and (relative) complements, in the rule.

In the first case, when the first hypersurface S_1 is found, the variables occurring in its description are the minimum number of variables needed to describe S. From this point on the algorithm can be restricted to use only these. If convergence is achieved a rule involving this minimal set of variables is obtained; we fondly refer to this variation as **Enclosed Cavities** and abbreviate it by **EC**. By contrast, when the algorithm is allowed to operate on all the initially selected variables **at each step**, the number of operations in the terminating expression is reduced. This variation of the classifier is called **Nested Cavities** (abbr. **NC**). Clearly the minimal set of variables needed to specify S is not given by **NC**. In practice, the reduction in the number of steps between **EC** and **NC** turns out to be substantial.

In the 3 cases presented next the dimensionality was lowered significantly not only by **EC** but also by **NC** to less than half and in one case to about 1/4 of the original variables.

One of the difficult problems in using parallel coordinates for viewing a specific dataset is to somehow find an axes permutation which is "good" (i.e. provides rich visual cues on what may be true or not) about the **specific** dataset. There is an inherent ordering emerging from dimensionality selection which, as we see below, answers this need well. This ordering is completely dataset specific. Further, since the algorithm is *display independent* there is no inherent limitation as to the size and number of variables in the dataset. The most significant limitation then in visual data mining is finally overcome. The visual aspects can now be used for displaying the result as well as exploring the salient features of the distribution of data brought out by the classifier.

This is not the right forum to analyze the computational complexity and other intricacies of the algorithm. It is worth pointing out that achieving an "optimum", in the sense of minimizing the number of cavities, turns out to be an NP-complete problem. Still the next best thing is done here in terms of discovering the cavities in order of decreasing size. Other relevant aspects are:

- an approximate convex-hull boundary for each cavity is obtained,

- utilizing properties of the representation of multidimensional objects in ||-coords, a very low polynomial worst case complexity of $O(N^2|P|^2)$ in the number of variables N and dataset size $|P|$ is obtained; it is worth contrasting this with the often unknown, or unstated, or very high (even exponential) complexity of other classifiers,
- an intriguing prospect, due to the low complexity, is that the rule can be derived in near real-time making the classifier **adaptive** to changing conditions,
- the minimal subset of variables needed for classification is found,
- the rule is given explicitly in terms of conditions on these variables, in terms of included and excluded intervals, and provides "a picture" showing the complex distributions with regions where there is data and "holes" with no data; that can provide significant insights to the domain experts,

Results

Three datasets, benchmarks in classification, are used to test the classifier. The results are then compared to those obtained with other well-known classifiers.

On the classifiers

During 1990-1993, Michie, Spiegelhalter and Taylor [16], on behalf of the ESPRIT program of the European Union, made extensive studies of several classifiers applied to diverse datasets. About 23 different classifiers were applied to about 20 datasets for comparative trials in the **StatLog** project. This was designed to test classification procedures on large-scale commercially important problems in order to determine suitability of the various techniques to industry. There were three main types of classifiers used:

1. **Extension to Linear Discrimination :** *Discrim, Logdisc, Quadisc, SMART, Backdrop, Cascade and DIPOL92.*
2. **Decision Trees and Rule-Based Methods :** *NewID, AC^2, Cal5, C4.5, CART, IndCART, Baytree, CN2, ITRule*
3. **Density Estimates :** *NaiveBay, CASTLE, ALLOC80, K-NN, RBF, Kohonen and LvQ.*

Data, Results and Comparisons

Satellite image data

The dataset used in *Statlog* is from a region in Australia. It consists of multispectral values and associated classification according to ground type and can be found in the *Statlog* ftp site. Each frame consists of four digital images of the same scene in different spectral bands, two in the visible and two in the near infra-red region. There are 36 variables (the attributes) and the class attribute for six (6) soil types i.e. the six classes to be characterized by the classifier(s). The data has 4435 samples(data items) for the training set and 2000 samples in the test set. By way of explanation, for validation the dataset is partitioned into *training* and *testing* subsets, the "popular" proportions being about 2/3 to 1/3. The rule is derived, by the classifier, on the training set and tested on the remainder of the data; the error pertains to the false "positives" and "negatives". The comparative results are shown in Table 1 below. An important observation, is that in a great many cases, S_2 turned out to be the hypersurface requiring the largest number of variables for its definition. We conjecture that this is an indication of the existence of many "borderline" cases (i.e. close elements in the class S and it's complement), or it may suggest that the class definition may be "fuzzy".

Rank	Classifier	Error rate %	
		Train	Test
1	**NC**	**4.3**	**9.0**
2	k-NN	8.9	9.4
3	LVQ	4.8	10.5
4	DIPOL92	5.1	11.1
5	RBF	11.1	12.1
6	ALLOC80	3.6	13.2
7	IndCART	2.3	13.8
8	CART	7.9	13.8
9	Backprop	11.2	13.9
10	Baytree	2.0	14.7
11	CN2	1.0	15.0
12	C4.5	4.0	15.0
13	NewID	6.7	15.0
14	Cal5	12.5	15.1
15	Quadisc	10.6	15.5
16	AC^2		15.7
17	SMART	12.3	15.9
18	Cascade	11.2	16.3
19	Logdisc	11.9	16.3
20	Discrim	14.9	17.1
21	Kohonen	10.1	17.9
22	CASTLE	18.6	19.4
23	NaiveBay	30.8	28.7
24	ITrule	Failed	Failed

Table 1: Summary of the *StatLog* results and comparison with the **Nested Cavities (NC)** classifier for the satellite image data. The error is averaged over the six classes.

Vowel recognition data

The data collection process involves digital sampling of speech with acoustic signal processing, followed by recognition of the phonemes, groups of phonemes and words. The goal here is a speaker-independent rule based on 10

Rank	Classifier	Testing Mode	Test Error Rate %
1	**Nested Cavities (NC)**	**Cross validation**	**7.9**
2	CART-DB	Cross validation	10.0
3	**Nested Cavities (NC)**	**Train & Test**	**10.5**
4	**Enclosed Cavities (EC)**	**Cross validation**	**13.9**
5		**Train & Test**	**13.9**
6	CART	Cross validation	21.8
7	k-NN	Train & Test	44.0
8	RBF	Train & Test	46.5
9	Multi-layer perceptron	Train & Test	49.4
10	Single-layer perceptron	Train & Test	66.7

Table 2: Summary of classification results for the vowel dataset.

variables of 11 vowels occurring in various words spoken (recorded and processed) by 15 British male and female speakers. Deterding [7] collected this dataset of vowels and which can be found in the CMU benchmark repository on the WWW. There are 528 entries for training and 462 for testing. Three other types of classifiers were also applied to this dataset: neural networks and k-NN by Robinson & Fallside [12], and Decision trees by Shang and Breiman [17]. For the sake of variety both versions of our classifier were used and a somewhat different error test procedure was used. The results are shown in Table 2.

Monkey neural data

We have decided to include the result on this dataset due to its interesting and unusual features. Here there are two classes to be distinguished consisting of pulses measured on two types of neurons in a monkey's brain (poor thing!). The experiment was conducted at the Yale Medical School and we received the data from Prof. R. Coiffman's[2] group working on this classification problem. There are 600 samples with 32 variables. Remarkably, convergence was obtained and required only 9 of the 32 parameters. The resulting ordering shows a striking separation. In the attached figure the first pair of variables x_1, x_2 originally given is plotted showing no separation. In the adjoining plot the best pair x_{11}, x_{14}, as chosen by the classifier's ordering, shows remarkable separation. The discovery of this manually would require constructing and inspecting a scatterplot with 496 pairs ...!

[2]He was the recipient of the National Medal of Science in Mathematics for the year 2000.

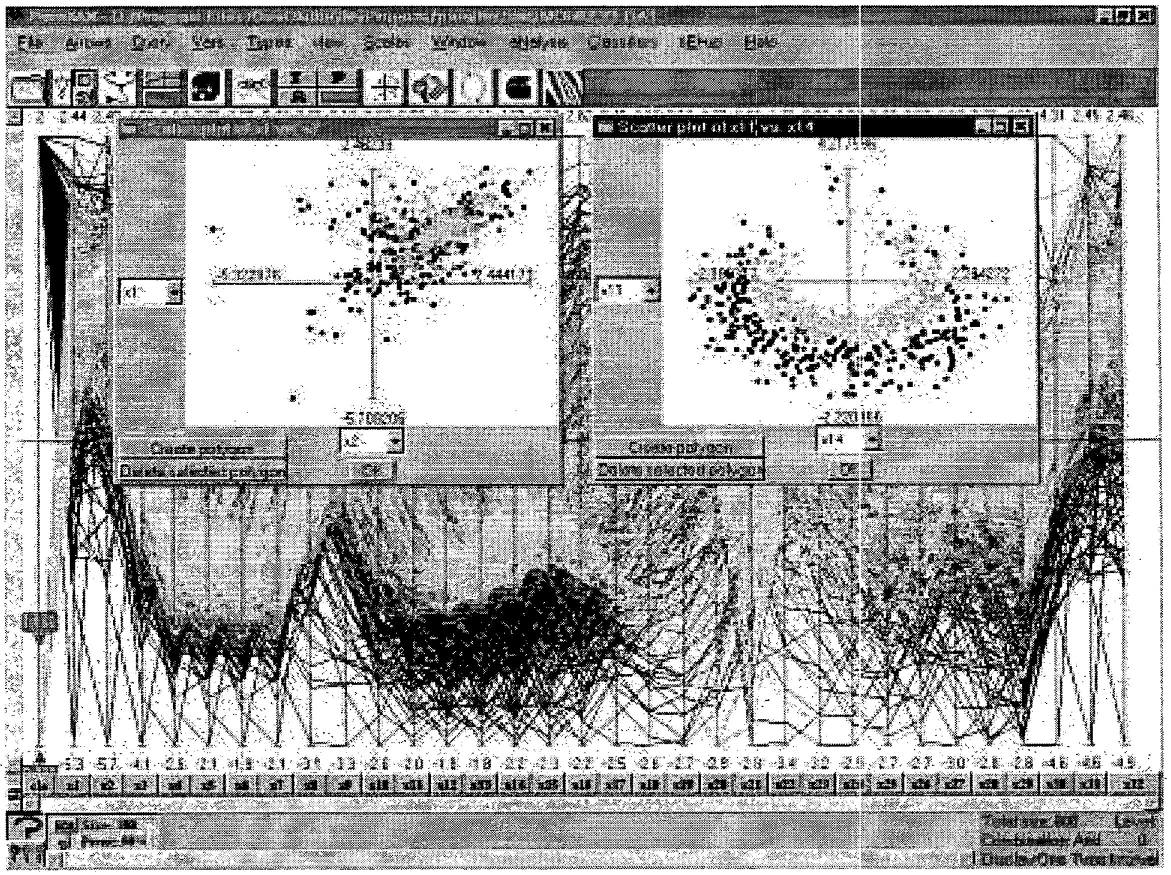
Figure 12: The monkey dataset showing the separation achieved by two of the 9 out 32 parameters obtained from the dimensionality selection.

The result shows that the data consists of two "banana-like"[3] clusters in 8-D one (the complement in this case) enclosing the other (class for which the rule was found). Note that the classifier can actually describe highly complex regions. It can build and "carve" the cavity shown. It is no wonder that separation attempts in terms of hyperplanes or nearest-neighbor techniques can fail badly on such datasets. The rule gave an error of 3.92 % using train-and-test with 66 % of the data for training) and impressed the Yale group – not an easy feat!

The geometric formulation combined with the results on the representation of multidimensional objects in ||-coords gave a classifier with remarkably low computational complexity. This makes feasible the classification of truly large in size and number of variable datasets,

[3]Perhaps the monkey was dreaming about bananas during this fateful experiment ...

i.e. the scalability problem is overcome, something we hope to test with suitable partners in the near future. The low complexity, enables the derivation of the rule in near real-time, and *then* apply it to incoming data, rendering the classifier **adaptive** to changing conditions. The rules provided are explicit, and "visualizable" and yield dimensionality selection which choses and orders the minimal set of variables needed to state the rule **without loss of information**. As it often happens, new questions have been raised on, termination criteria, automatic approaches to overfiting, interpretation of the "geometry" of the dataset as described by the rule and others.

Visual & Computational Models

$\mathcal{F}$inally we illustrate the methodology's ability to model multivariate relations in terms of hypersurfaces – just as we model a relation between two variables by a planar region. Then by using the interior point algorithm, as shown for example in Fig. 7, with the model we can do trade-off analyses, discover sensitivities, understand the impact of constraints, and in some cases do optimization. For this purpose we shall use a dataset consisting of the outputs of various economic sectors and other expenditures of a particular (and real) country. It consists of the monetary values over several years for the **Agr**icultural, **Fish**ing, and **Min**ing sector outputs, **Manu**facturing and **Cons**truction industries, together with **Gov**ernment, Miscellaneous spending and resulting GNP; eight variables altogether. We will not take up the full ramifications of constructing a model from data. Rather, we want to illustrate how ||-coords may be used as a modeling tool. Using the Least Squares technique we "fit" a function to this dataset and we are not concerned at this stage whether the choice of function is a "good" choice or not. The function we obtained bounds a region in R^8 and is represented by the upper and lower curves shown in Fig. 13.

The picture is in effect a simplistic *visual* model of the country's economy, incorporating it's capabilities, limitations and interelationships among the sectors etc. A point interior to the region, satisfies all the constraints simultaneously, and therefore represents (i.e. the 8-tuple of values) a *feasible economic policy* for that country. Using the interior point algorithm we can construct such points. It can be done interactively by sequentially choosing values of the variables and we see the result of one such choice in Fig. 13. Once a value of the first variable is chosen (in this case the agricultural output) within it's range, the dimensionality of the region is reduced by one. In fact, the upper and lower curves between the 2nd and 3rd axes correspond to the resulting 7-dimensional hypersurface and show the *available* range of the second variable (Fishing) reduced by the constraint. In fact, this can be seen (but not shown here) for the rest of the variables. That is, due to the relationship between the 8 variables, a constraint on one of them impacts all the remaining ones and restricts their range. The display allows us to experiment and actually **see** the impact of such decisions "downstream". By interactively varying the chosen value for the first variable we found, that it not possible to have a policy that favors Agriculture without also favoring Fishing and vice versa.

Proceeding, a very high value from the available range of Fishing is chosen and it corresponds to very low values of the Mining sector. By contrast in Fig. 13 we see that a low value in Fishing yields high values for the Mining sector. This inverse correlation was examined and it was found that the country in question has a large number of *migrating* semi-skilled workers. When the fishing industry is doing well most of them are attracted to it leaving few available to work in the mines and vice versa. The comparison between the two figures shows the *competition for the same resource* between Mining and Fishing. It is especially instructive to discover this interactively. The construction of the interior point proceeds in the same way.

Let us move over to Fig. 7 where the same construction is shown but for a more complex 20-dimensional hypersurface ("model"). The intermediate curves (upper and lower) also provide valuable information and "previews of coming attractions". They indicate a

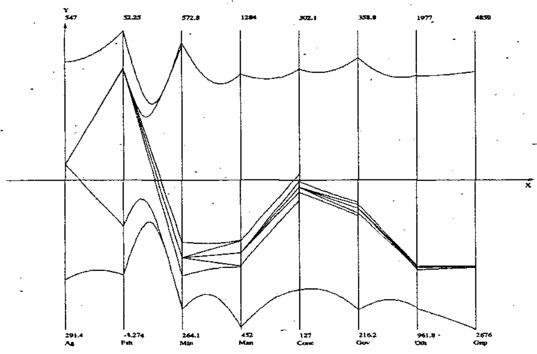

Figure 13: Model of a country's economy

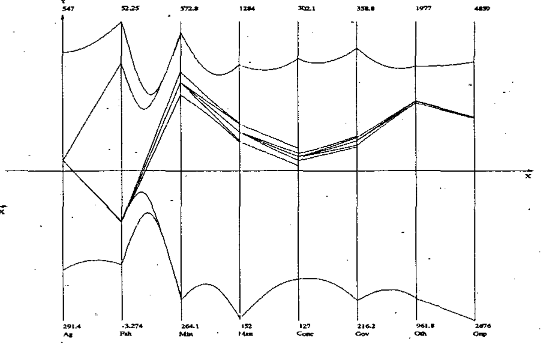

Figure 14: Competition for labor between the Fishing & Mining sectors – compare with previous figure

neighborhood of the point (represented by the polygonal line) and provide a feel for the local curvature. Note the narrow strips between $X13$, $X14$ and $X15$ (as compared to the surrounding ones), indicating that for this choice of values these 3 are the *critical* variables where the point is "bumping the boundary". A theorem guarantees that a polygonal line which is in-between all the intermediate curves/envelopes represents an interior point of the hypersurface and all interior points can be found in this way. If the polygonal line is tangent to anyone of the intermediate curves then it represents a *boundary point*, while if it crosses anyone of the intermediate curves it represents an *exterior point*. The later enables us to see, in an application, the first variable for which the construction failed and what is needed to make corrections. By varying the choice of value over the available range of the variable interactively, sensitive regions (where small changes produce large changes downstream) and other properties of the model can be easily discovered. Once the construction of a point is completed it is possible to vary the values of each variable and see how this effects the remaining variables. So one can do *trade-off analysis* in this way and provide a powerful tool for, Decision Support, Process Control and other applications. As new data becomes available the model can be updated with the Decision Making being based on the most recent information.

References

[1] T. Avidan and S. Avidan. Parallax - a data mining tool based on parallel coordinates. *Comput. Statit.*, 13-1:65, 1998.

[2] E.W. Bassett. Ibm's ibm fix. *Industrial Computing*, 14(41):23–25, 1995.

[3] V. G. Boltyanskii. *Envelopes, R.B.Brown translator (original in Russian).* Pergamon Press, New York, 1964.

[4] A. Chatterjee. *Visualizing Multidimensional Polytopes and Topologies for Tolerances.* Ph.D. Thesis USC, 1995.

[5] H. S. M. Coxeter. *The Real Projective Plane.* Third Edition, Springer-Verlag, New York, 1992.

[6] J. Eickemeyer. *Visualizing p-flats in N-space using Parallel Coordinates.* Ph.D. Thesis UCLA, 1992.

[7] Deterding D. H. *Speaker Normalization for Automatic Speech Recognition.* Ph.D. Thesis, Cambridge University, 1989.

[8] A. Inselberg. Don't panic ... do it in parallel! *Comput. Statit.*, 14:53–77, 1999.

[9] A. Inselberg and B. Dimsdale. Multidimensional lines ii: Proximity and applications. *SIAM J. of Applied Math.*, 54-2:578–596, 1994.

[10] A. Inselberg, M. Reif, and T. Chomut. Convexity algorithms in parallel coordinates. *J. ACM*, 34:765–801, 1987.

[11] A. Inselberg and Avidan T. *The* Automated *Multidimensional Detective, in Proc. of IEEE Information Visualization '99, 112-119.* IEEE Comp. Soc., Los Alamitos, CA, 1999.

[12] Robinson A. J. and Fallside F. *A Dynamic Connectionist Model of Phoneme Recognition.* Proc. of 1st European Neural Network Conf. (nEURO), 1988.

[13] Quinlan J.R. *C4.5 : Programs for Machine Learning.* Morgan Kaufman, 1993.

[14] D. A. Keim and H. P. Kriegel. Visualization techniques for mining large databases: A comparison. *Trans. Knowl. and Data Engr.*, 8-6:923–938, 1996.

[15] A. R. Martin and M. O. Ward. *High dimensional brushing for interactive exploration of multivariate data, Proc. IEEE Conf. on Visualization, Atlanta, GA, 271-278.* IEEE Comp. Soc., Los Alamitos, CA, 1995.

[16] Spiegelhalter D.J. Michie D. and Taylor C.C. *Machine Learning , Neural and Statistical Classification.* Ellis Horwood series in AI, 1994.

[17] Shang N. and Breiman L. *Distribution based trees are more accurate.* Proc. of 1st Inter. Conf. on Neural Info. Proc. (ICONIP96), 133, 1996.

[18] C. Schmid and H. Hinterberger. *Comparative Multivariate Visualization Across Conceptually Different Graphic Displays, in Proc. of 7th SSDBM.* IEEE Comp. Soc., Los Alamitos, CA, 1994.

[19] Fayad U. M. Smyth P., Piatesky-Shapiro G. and Uthurusamy R. *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press, 1996.

[20] D. F. Swayne, D. Cook, and A. Buja. *XGobi : Interactive Dynamic Graphics in the X Window System.* JCGS, **7-1**, 113-130, 1998.

[21] Mitchell T.M. *Machine Learning.* McGraw-Hill, 1997.

[22] E. R. Tufte. *The Visual Display of Quantitative Information.* Graphic Press, Connecticut, 1983.