On Some Mathematics for Visualizing High Dimensional Data

Edward J. Wegman
Jeffrey L. Solka

Center for Computational Statistics
George Mason University
Fairfax, VA 22030

This paper is dedicated to Professor C. R. Rao on the occasion of his 80th birthday.

Abstract: The analysis of high-dimensional data offers a great challenge to the analyst because the human intuition about geometry of high dimensions fails. We have found that a combination of three basic techniques proves to be extraordinarily effective for visualizing large, high-dimensional data sets. Two important methods for visualizing high-dimensional data involve the parallel coordinate system and the grand tour. Another technique which we have dubbed saturation brushing is the third method. The parallel coordinate system involves methods in high-dimensional Euclidean geometry, projective geometry, and graph theory while the the grand tour involves high-dimensional space filling curves, differential geometry, and fractal geometry. This paper describes a synthesis of these techniques into an approach that helps build the intuition of the analyst. The emphasis in this paper is on the underlying mathematics.

1. Introduction

Seeing objects in high-dimensional spaces is often a fantasy of and a wish for many mathematicians. Often the revered mathematicians of the past were reputed to be able to see in their minds high-dimensional objects, which gave them insight beyond those of more ordinary mathematicians. Achieving such a feat has intrigued the present authors since we were mathematical juveniles. The combination of two techniques known respectively as *parallel coordinates* and the *d-dimensional grand tour* allows us to actually visually gain insight into hyperspace much like those revered mathematical geniuses of the past. A third method known as *saturation brushing* allows for visualization of relatively massive datasets. We have used these techniques in conjunction with each other for the analysis of data sets containing as many as 250,000 observations in as high as 18-dimensional space.

These techniques have been combined in an evolutionary series of softwares that one of us (EJW) has helped to design over the past decade or so. Our first effort dating from circa 1988 was a DOS program known as Mason Hypergraphics. This was followed around 1992 by a UNIX program known as ExplorN, and most recently in 2000 by a Windows 95/98/NT software known as Crystal Vision. We have written extensively about aspects of these ideas. See for example Wegman and Bolorforoush (1988), Wegman (1990), Miller and Wegman (1991), Wegman (1991), Wegman and Carr (1993), Wegman and Shen (1993), Wegman and Luo (1997), Solka, Wegman, Rogers, Poston (1997), Wegman, Poston and Solka (1998), Solka, Wegman, Reid and Poston (1998), and Wilhelm, Wegman and Symanzik (1999). Many of our ideas have appeared only in proceedings papers and technical reports, or have never been published at all. This paper is not intended as a general review paper of all high-dimensional data visualization methods, but as a synthesis of our approaches to visualizing high-dimensional data. Wegman, Carr and Luo (1993) provides a convienient introduction to many other high-dimensional data visualization techniques.

The present paper is intended to focus, not on the applications nor on the methodology itself, but rather on the underlying mathematics. The intent is to consolidate the mathematical developments that have been scattered over a wide variety of literature into one convenient paper. As indicated above, this paper is not an attempt to review all approaches to high dimensional visualization, but to synthesize specific techniques we have found useful. A companion paper focusing on applications of these methodologies to achieve a number of statistical tasks is being planned. However, the combination of mathematical tools, mainly geometric tools, is itself interesting, and in our view it is worthwhile to document these. It is especially of interest in a paper in honor of Professor C. R. Rao, who has contributed so much to make geometric methods an integral part of statistical analysis.

2. Parallel Coordinates

Parallel coordinate displays are a tool for visualizing multivariate data. They were introduced into the mathematics literature by Inselberg (1985) and suggested as a tool for high dimensional data analysis by Wegman (1990). Since then many additional refinements have been suggested. However, the basic parallel coordinate plot remains an intriguing mathematical object. Traditional Cartesian coordinates suffer from the fact that we live in three spatial dimensions. Beyond three dimensions all manner of artifices have been invented to represent higher-dimensional data, using time, color, glyphs, and so on. Regrettably these do not treat all variables in the same way and, hence, make comparing variables with different representations extremely difficult. The parallel coordinate plot device is based on the observation that problems associated with Cartesian plotting arise because of the orthogonality constraint. Because this is the case, in parallel coordinates, we simply give up orthogonality and draw the axes as parallel. Any number of parallel axes can be drawn in a plane[1]. If the data is d-dimensional, simply draw d parallel axes. A data vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$ is drawn by locating x_i on the i-th coordinate axis and simply joining the x_i to x_{i+1} by a line segment for $i = 1, \ldots, d-1$. There is in principle no upper bound on the dimension of the data that can be represented, although there are practical limits related to the resolution available on a computer screen and to the human eye. See Wegman (1990) for much more detail on interpretation and usage of these displays.

[1]Suggestions have also been made to replace the plane with a cylinder. In an attempt to overcome the pairwise adjacencies problem described in section 2.3, parallel axes are straight lines drawn on the cyclinder. In an attempt to deal with circular data, the parallel axes are actually drawn as parallel circles on the cylinder.

The power of and motivation for using parallel coordinate displays derives from the underlying connection with projective geometry. Axiomatic synthetic projective geometry is motivated by the asymmetry in Euclidean geometry induced by the parallel lines axiom. That is, **most** pairs of lines in a two-plane meet in a point, except if the lines are parallel. However, **all** pairs of points determine a line. In synthetic projective geometry, the parallel lines axiom is replaced with the axiom: **every pair of lines meets in a point**. Together with the other axioms of projective geometry, this axiom has the effect that any statement that is true about lines and points is also true when the words "line" and "point" are interchanged. This notion of duality between points and lines induces all types of additional dualities in a projective plane. Nondegenerate mappings between projective planes have the property of preserving certain geometric structures. In the case of transformations from Cartesian coordinate geometry to parallel coordinate geometry, this implies structure in Cartesian coordinates have a dual structure in parallel coordinates. The implication is that not only does a parallel coordinate display have the ability to uniquely map high-dimensional points into a planar diagram, but that the parallel coordinate display can be interpreted geometrically.

It is worth making a couple of observations. First synthetic projective geometry is an abstract mathematical construct. One model for synthetic projective geometry is the ordinary Euclidean plane supplemented by so-called ideal points. The ideal points are in one-to-one correspondence with the slopes of ordinary lines. The idea of this model is that "parallel lines meet at infinity." Hence all parallel lines having the same slope will meet at the same ideal point. The set of ideal points form the so-called ideal line. In this model, the projective plane thus has "regular" points, i.e. those from the Euclidean plane, and "ideal" points, which we have just described. Another model for the projective plane is the "cross cap," which is a hemisphere with opposite points on the equator topologically identified. Neither of these models for synthetic projective geometry is entirely satisfactory, since they make distinction among certain types of points. However the model, which regards the projective plane as an extended Euclidean plane, is extremely useful for data visualization purposes. Just as synthetic Euclidean geometry can be tied to the Cartesian coordinate system to form an analytic geometry, synthetic projective geometry can be tied to a coordinate system known as natural homogeneous coordinates to form an analytic projective geometry. We discuss these coordinates in section 2.2. For more details on projective geometry, see Wegman (2000).

Thus the intriguing aspect of parallel coordinate plots is the mathematical duality between Cartesian plots and parallel coordinate plots. For the purposes of the present mathematical discussion, we focus on just two dimensions. In this context we have just suggested that a point in an ordinary Cartesian plot is represented by a line in a parallel coordinate plot. Indeed, if we conceive of both the Cartesian two-dimensional plot and the parallel coordinate plot as representing two projective two-planes, we derive a number of interesting dualities.

2.1 Parallel Coordinate Geometry. The parallel coordinate representation enjoys some elegant duality properties with the usual Cartesian orthogonal coordinate representation. Consider a line $\mathcal{L}$ in the Cartesian coordinate plane given by $\mathcal{L}$: $y = mx + b$ and consider two points lying on that line, say $(a, ma + b)$ and $(c, mc + b)$. For simplicity of computation we consider the xy Cartesian axes mapped into the xy parallel axes as described in Figure 2.1. We superimpose a Cartesian coordinate axes tu on the xy parallel axes so that the y parallel axis has the equation $u = 1$. The point $(a, ma + b)$ in the xy Cartesian system maps into the line joining $(a, 0)$ to $(ma + b, 1)$ in the tu coordinate axes. Similarly, $(c, mc + b)$ maps into the line joining $(c, 0)$ to $(mc + b, 1)$. It is a straightforward computation to show that these two lines intersect at a point (in the tu plane) given by $\overline{\mathcal{L}}:\ (b(1-m)^{-1}, (1-m)^{-1})$. Notice that this point in the parallel coordinate plot depends only on m and b the parameters of the original line in the Cartesian plot. Thus $\overline{\mathcal{L}}$ is the dual of $\mathcal{L}$ and we have the interesting duality result that points in Cartesian coordinates map into lines in parallel coordinates while lines in Cartesian coordinates map into points in parallel coordinates.

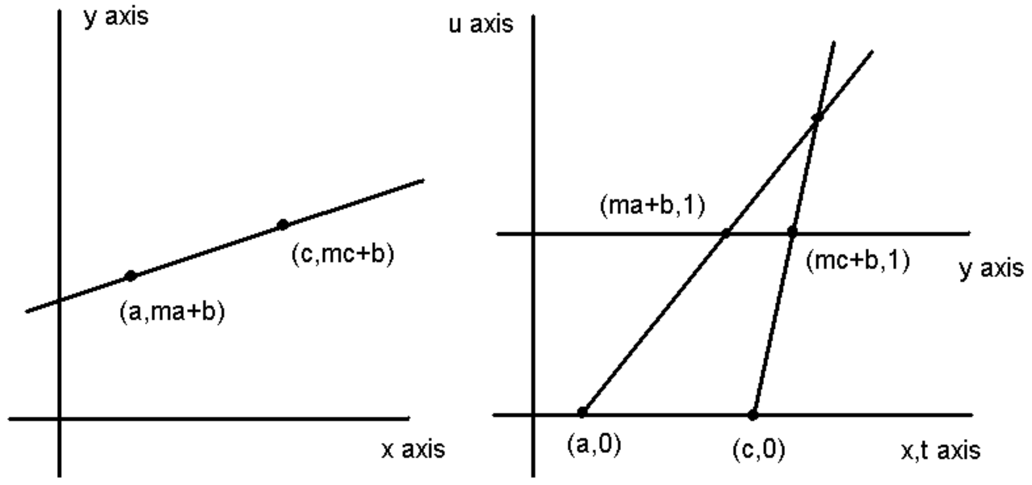

Figure 2.1 Illustrating the duality between points and lines in Cartesian and parallel coordinate plots

For $0 < (1-m)^{-1} < 1$, m is negative and the intersection occurs between the parallel coordinate axes. For $m = -1$, the intersection is exactly midway. A ready statistical interpretation can be given. For highly negatively correlated pairs, the dual line segments in parallel coordinates will tend to cross near a single point between the two parallel coordinate axes. The scale of one of the variables may be transformed in such a way that the intersection occurs midway between the two parallel coordinate axes in which case the slope of the linear relationship is negative one.

In the case that $(1-m)^{-1} < 0$ or $(1-m)^{-1} > 1$, m is positive and the intersection occurs external to the region between the two parallel axes. In the special case $m = 1$, this formulation breaks down. However, it is clear that the point pairs are $(a, a + b)$ and $(c, c + b)$. The dual lines to these points are the lines in parallel coordinate space with slope b^{-1} and intercepts $-ab^{-1}$ and $-cb^{-1}$ respectively. Thus the duals of these lines in parallel coordinate space are parallel lines with slope b^{-1}. We thus append the ideal points to the parallel coordinate plane to obtain a projective plane. These parallel lines intersect at the ideal point in direction b^{-1}. In the statistical setting, we have the following interpretation. For highly positively correlated data, we will tend to have lines not intersecting between the parallel coordinate axes. By suitable linear rescaling of one of the variables, the lines may be made approximately parallel in direction with slope b^{-1}. In this case the slope of the linear relationship between the rescaled variables is one.

2.2. Natural Homogeneous Coordinates and Conics. The point-line, line-point duality seen in the transformation from Cartesian to parallel coordinates extends to conic sections. To see this consider both the xy plane and the tu plane to be augmented by suitable ideal points so that we may regard both as projective planes. The representation of points in parallel coordinates is thus a transformation from one projective plane to another. Computation is simplified by an analytic representation. However, the usual coordinate pair, (x, y), is not sufficient to represent ideal points. Thus, for purposes of analytic projective geometry, we represent points in the projective plane by triples (x, y, z). As motivation for this representation, consider two distinct parallel lines having equations in the projective plane

$$ax + by + cz = 0 \text{ and } ax + by + c'z = 0. \tag{2.1}$$

Simultaneous solution yields $(c - c')z = 0$ so that $z = 0$. Thus when $z = 0$, the triple (x, y, z), i.e. $(x, y, 0)$, describes ideal points. The representation of points in the projective plane is by triples, (x, y, z), which are called natural homogeneous coordinates. If $z = 1$, the resulting equation is $ax + by + c = 0$ and so $(x, y, 1)$ is the natural representation of a point (x, y) in Cartesian coordinates lying on $ax + by + c = 0$. Notice that if $(\gamma x,\ \gamma y,\ \gamma)$ is any multiple of $(x, y, 1)$ on $ax + by + c = 0$, we have

$$a\gamma x + b\gamma y + c\gamma = \gamma(ax + by + c) = \gamma \cdot 0 = 0. \tag{2.2}$$

Thus the triple $(\gamma x, \gamma y, \gamma)$ equally well represents the Cartesian point (x, y) lying on $ax + by + c = 0$ so that the representation of a point in natural homogeneous coordinates is not unique. However, if γ is not 1 or 0, we can simply re-scale the natural homogeneous triple to have a 1 for the z-component and thus read off the Cartesian coordinates directly. If the z-component is zero, we know immediately that we have an ideal point.

Notice that we could equally well consider the triples (a, b, c) as natural homogeneous coordinates of a line. Thus, triples can either represent points or lines reiterating the fundamental duality between points and lines in the projective plane. Recall now that the line $\mathcal{L}$: $y = mx + b$ mapped into the point $\overline{\mathcal{L}}$: $(b(1-m)^{-1}$,$(1-m)^{-1})$ in parallel coordinates. In natural homogeneous coordinates, $\mathcal{L}$ is represented by the triple $(m, -1, b)$ and the point $\overline{\mathcal{L}}$ by the triple $(b(1-m)^{-1}, (1-m)^{-1}, 1)$ or equivalently by $(b, 1, 1-m)$. The latter yields the appropriate ideal point when $m = 1$. A straightforward computation shows for

$$A = \begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & -1 \\ 1 & 0 & 0 \end{bmatrix} \tag{2.3}$$

that $t = xA$ or $(b, 1, 1-m) = (m, -1, b)A$. Thus the transformation from lines in orthogonal coordinates to points in parallel coordinates is a particularly simple projective transformation with the rather nice computational property of having only adds and subtracts.

Similarly, a point $(x_1, x_2, 1)$ expressed in natural homogeneous coordinates maps into the line represented by $(1, x_1 - x_2, -x_1)$ in natural homogeneous coordinates. Another straightforward computation shows that the linear transformation given by $t = xB$ or $(1, x_1 - x_2, -x_1) = (x_1, x_2, 1)B$ where

$$B = \begin{bmatrix} 0 & 1 & -1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{2.4}$$

describes the projective transformation of points in Cartesian coordinates to lines in parallel coordinates. Because these are nonsingular linear tranformations, hence projective transformations, it follows from the elementary theory of projective geometry that conics are mapped into conics. This is straightforward to see since an elementary quadratic form in the original space, say $xCx' = 0$ where x' denotes x transpose, represents the general conic. Clearly then since $t = xB$, B nonsingular, we have $x = tB^{-1}$, so that $tB^{-1}C(B^{-1})'t' = 0$ is a quadratic form in the image space. An instructive computation involves computing the image of an ellipse $ax^2 + by^2 - cz^2 = 0$ with $a, b, c > 0$. The image in the parallel coordinate space is $c(t+u)^2 - bu^2 = av^2$, a general hyperbolic form.

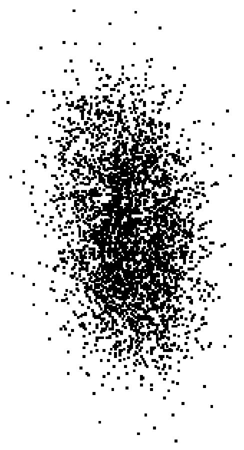

Figure 2.2a: One scatterplot of five-dimensional data showing elliptical cross section.

It should be noted that the solution to this equation is not a locus of points, but the natural homogeneous coordinates of a locus of lines, a line conic. The envelope of this line conic is a point conic. In the case of this computation, the point conic in the original Cartesian coordinate plane is an ellipse, the image in the parallel coordinate plane is as we have just seen a line hyperbola with a point hyperbola as envelope.

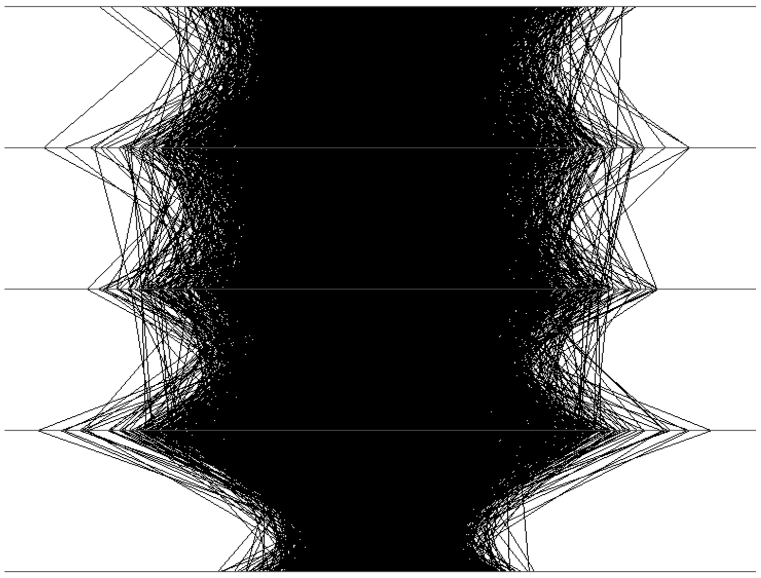

Figure 2.2b: Parallel coordinate plot of the same five-dimensional data showing the hyperbolic dual structure.

We mentioned the duality between points and lines and conics and conics. It is worthwhile to point out two other nice dualities. Rotations in Cartesian coordinates become translations in parallel coordinates and vice versa. Perhaps more interesting from a statistical point of view is that points of inflection in Cartesian space become cusps in parallel coordinate space and vice versa. Thus the relatively hard-to-detect inflection point property of a function becomes the notably more easy to detect cusp in the parallel coordinate representation. Inselberg (1985) discusses these properties in detail. It is well worth noting that the natural homogeneous coordinate representation is a standard device in computer graphics.

2.3 Permutation of the Axes for Pairwise Comparisons. One of the most common objections to parallel coordinate displays is the preferential positioning of adjacent axes. If the parallel coordinate axes are ordered from 1 through d, then there is an easy pairwise comparison of 1 with 2, 2 with 3 and so on. However, the pairwise comparison of 1 with 3, 2 with 5 and so on was not easily done because these axes were not adjacent. One simple mathematical question then is what is the minimal number of permutations of the axes in order to guarantee all possible pairwise adjacencies. Although there are $n!$ permutations, many of these duplicate adjacencies. Actually far fewer permutations are required.

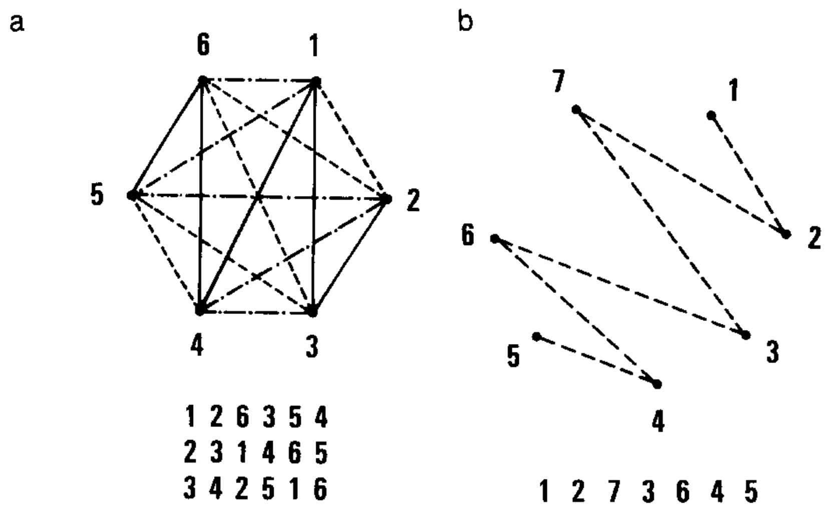

Figure 2.3 Illustrating the graph labeling for determining parallel coordinate permutations

A construction for determining the permutations is represented in Figure 2.3. A graph is drawn with vertices representing coordinate axes, labeled clockwise 1 to d. Edges represent adjacencies, so that vertex one connected to vertex two by an edge means axis one is placed adjacent to axis two. To construct a minimal set of permutations that completes the graph is equivalent to finding a minimal set of orderings of the axes so that every possible adjacency is present. Figure 2.3b illustrates the basic zig-zag pattern used in the construction. This creates an ordering which in the example of Figure 2.3b is 1 2 7 3 6 4 5. For d even this general sequence can be written as $1,\ 2,\ d,\ 3,\ d-1,\ 4,\ d-2, \ldots,\ (d+2)/2$ and for d odd as $1,\ 2,\ d,\ 3,\ d-1,\ 4,\ d-2, \ldots,\ (d+3)/2$.

An even simpler formulation is

$$d_{k+1} \ = \ (d_k + (-1)^{k+1} k) \, mod \, d, \ k = 1, 2, \ldots, d-1 \tag{2.5}$$

with $d_1 = 1$. Here it is understood that $0 \, mod \, d = d \, mod \, d = d$. This zig-zag pattern can be recursively applied to complete the graph. That is to say if we let $d_k^{(1)} = d_k$, we may define

$$d_k^{(j+1)} \ = \ (d_k^{(j)} + 1) \, mod \, d, \ j = 1, 2, \ldots, [\tfrac{d-1}{2}] \tag{2.6}$$

where $[\,\cdot\,]$ is the greatest integer function. For d even, it follows that this construction generates each edge in one and only one permutation. Thus $d/2$ is the minimal number of permutations needed to assure that every edge appears in the graph or equivalently that every adjacency occurs in the parallel coordinate representation. For d odd, the result is not exactly the same. We will not have any duplication of adjacencies for $j < [\,\frac{d-1}{2}]$. However, $j < [\,\frac{d-1}{2}]$ will not provide a complete graph. The case $j = [\,\frac{d-1}{2}]$ in equation (2.6) will complete the graph, but also create some redundancies. Nevertheless, it is clear that $[\,\frac{d+1}{2}]$ permutations are the minimal number needed to complete the graph and thus provide every adjacency in the parallel coordinate representation. Thus we have that the minimal number of permutations of the d parallel coordinate axes needed to insure adjacency of every pair of axes is $[\,\frac{d+1}{2}]$. These permutations may be constructed using formulas (2.5) and (2.6). It is worthwhile to point out that all possible pairs may be found in only $[\,\frac{d+1}{2}]$ distinct parallel coordinate plots, but for a scatterplot matrix, $\binom{d}{2} = \frac{d^2-d}{2}$ plots are required. One practical consequence is that for a fixed computer screen size, elements in the scatterplot matrix become difficult to see much more rapidly than the parallel coordinate plots. In general such permutation arguments are rendered unnecessary with the introduction of the grand tour.

3. The Grand Tour in *d*-dimensions

The grand tour is, in a sense, the generalization of rotations in high-dimensional space and is an invaluable tool for animating high-dimensional visualization. When used in conjunction with scatterplot matrix displays or with parallel coordinate displays, the grand tour allows the data analyst a variety of views for exploring the structure of data. The basic idea, introduced by Asimov (1985) and Buja and Asimov (1985), is to capture the popular sense of a grand tour. That is, to fully understand a subject item, one must examine it from all possible angles. This translates in a mathematical perspective to examining the data cloud from all possible angles. In the formulation introduced by Asimov and Buja, this meant to project into a set of two-planes dense in the d-dimensional space of the data. The idea is to move from one two-plane to the next so as to see the data from all possible angles. Not only should the set of two-planes be dense in the data space, but it is also required to move continuously (smoothly) from one two-plane to the next so that the human visual system can smoothly interpolate the data and track individual points and structures in the data. Hence the mathematics of the Asimov-Buja grand tour requires a continuous, space-filling path through the set of two planes in the d-dimensional data space. The idea then is to project the data onto the two-planes and view them in a time-sequenced set of two-dimensional images. The practical implementation is to step through the set of two-planes with a small step size in time rather than to move through the set of two-planes in some continuous sense. This type of grand tour was also studied by Buja, Hurley, and McDonald (1986), Cook, Buja and Cabrera (1991), Cook et al. (1993), Cook et al. (1995), Cook and Buja (1997), Furnas and Buja (1994), and Hurley and Buja (1990), .

Wegman (1991) formally suggested replacing the manifold of two-planes with a manifold of k-planes where $k \leq d$, d being the dimension of the data space, and discussed adapting the methods of Asimov-Buja for constructing a space-filling curve through the manifold of k-planes. The data is then projected into the k-plane and visualized using either a parallel coordinate display or a scatterplot matrix display. This method was actually implemented in the Mason Hypergraphics software (Wegman and Bolorforoush, 1988) much earlier. The approach formulated by Asimov is known as *the torus method* for reasons that shall become clear during our development of the grand tour mathematics. The geometric form of the standard two-torus imbedded in three space is of course the traditional doughnut shape. The generalization of the two-torus to higher dimensional space is harder to visualize. For this reason, it is somewhat easier to conceive of the basic structure of interest as a multidimensional hypercube. The the torus, then, is a hypercube with opposite faces identified. Because we want to deal with angles, we can tthink of the length of each side of the cube a 2π. We shall first formulate the Asimov-Buja winding algorithm, then a random curve algorithm, and finally a fractal algorithm. We also present a two-dimensional pseudo grand tour.

3.1 The Asimov-Buja Winding Algorithm in $\boldsymbol{d}$-space. Let $\boldsymbol{e}_j = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$ be the canonical basis vector of length d. The 1 is in the j-th position. The $\boldsymbol{e}_j$ are the unit vectors for each of the coordinate axes in the initial position. We want to do a general rigid rotation of these axes into a new position with basis vectors $\boldsymbol{a}_j(t) = (a_j^1(t), a_j^2(t), \ldots, a_j^d(t))$, where t is the time index. The strategy then is to take the inner product of each data point, say $\boldsymbol{x}_i$**,** $i = 1, \ldots, n$ with the basis vectors, $\boldsymbol{a}_j(t)$. This operation projects the data into the rotated coordinate system. By convention, d will refer to the dimension of the data and n will refer to the sample size of the data set. Of course, the j subscript on $\boldsymbol{a}_j(t)$ means that $\boldsymbol{a}_j(t)$ is the image under the generalized rotation of the canonical basis vector $\boldsymbol{e}_j$. Thus the data vector $\boldsymbol{x}_i$ is $(x_1^i, x_2^i, \ldots, x_d^i)$ so that the representation of $\boldsymbol{x}_i$ in the $\boldsymbol{a}_j$ coordinate system is

$$\boldsymbol{y}_i(t) = (y_1^i(t), y_2^i(t), \ldots, y_d^i(t)),\ i = 1, 2, \ldots, n \tag{3.1}$$

with

$$y_j^i(t) = \sum_{k=1}^{d} x_k^i a_j^k(t),\ j = 1, \ldots, d \text{ and } i = 1, \ldots, n. \tag{3.2}$$

The vector $\boldsymbol{y}_i(t)$ is a linear combination of the basis vectors representing the i-th data point in the rotated coordinate system at time t. It is also worth pointing out that $\boldsymbol{y}_i(t)$ is also a linear combination of the data. If one component of the vector is held out from the grand tour (i.e. a partial grand tour), then the partial grand tour lends itself to an interpretation in terms of multiple linear regression.

The general goal then is to find a generalized rotation $\boldsymbol{Q}$ such that $\boldsymbol{Q}(\boldsymbol{e}_j) = \boldsymbol{a}_j$. We can conceive of $\boldsymbol{Q}$ as either a function on the space of basis vectors or as a $d \times d$

matrix $\boldsymbol{Q}$ where $\boldsymbol{e}_j \times \boldsymbol{Q} = \boldsymbol{a}_j$. We implement this by choosing $\boldsymbol{Q}$ as an element of the special orthogonal group denoted by $SO(d)$ of orthogonal $d \times d$ matrices having determinant $+1$. Thus we must find a continuous space filling curve through $SO(d)$. We shall do this by a composite mapping from the real line, $\mathbb{R}$, to the p-dimensional hypercube $[0, 2\pi]^p$, i.e. $\alpha : \mathbb{R} \to [0, 2\pi]^p$, where $p = d(d-1)/2$. The components of $\alpha(t)$ are taken to be angles. The mapping β from $[0, 2\pi]^p$ onto $SO(d)$ is given by

$$\beta(\theta_{1,2}, \theta_{1,3}, \ldots, \theta_{d-1,d}) = R_{12}(\theta_{12}) \times R_{13}(\theta_{13}) \times \cdots \times R_{d-1,d}(\theta_{d-1,d}). \tag{3.3}$$

The fact that this is an onto mapping guarantees that the curve is space filling. See Asimov (1985). There are $p = \frac{1}{2}(d^2 - d)$ factors in the expression (3.3). These correspond to the $\binom{d}{2} = \frac{1}{2}(d^2 - d)$ distinct two-flats formed by the canonical basis vectors. In a general d-dimensional space, there are $d - 2$ axes orthogonal to each two-flat. Thus rather than rotating around an axis as we conventionally do in three-dimensional geometry, we must rotate in a two-plane in d-dimensional space. We let $R_{ij}(\theta)$ be the element of $SO(d)$ which rotates the $\boldsymbol{e}_i\boldsymbol{e}_j$ plane through an angle of θ. Thus

$$R_{ij}(\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & cos(\theta) & \cdots & -\,sin(\theta) & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & sin(\theta) & \cdots & cos(\theta) & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}, \tag{3.4}$$

where the cosines and sines are respectively in the i-th and j-th columns and rows. The restrictions on θ_{ij} are $0 \leq \theta_{ij} \leq 2\pi$, $1 \leq i < j \leq d$. The angles θ_{ij} are called the Euler angles. Finally, we construct $\alpha(t) = (\lambda_1 t, \lambda_2 t, \ldots, \lambda_p t)$ as the mapping from $\mathbb{R}$ to $[0, 2\pi]^p$ where, of course, $\lambda_j t$ is taken modulo 2π. $\lambda_1, \ldots, \lambda_p$ are taken to be linearly independent real numbers over the rational numbers. Thus we define $\boldsymbol{Q}_t = \beta(\alpha(t))$.

The fact that the λ are linearly independent over the rationals guarantees that no λ_j can be written in terms of the remaining λ_i. This guarantees that they are mutually irrational and that slopes through the hypercube cannot be multiples of one another. As mentioned earlier, opposite faces of a p-hypercube are topologically identified to construct a p-dimensional torus. Hence the terminology for the *torus method*. It is easy to see in two dimensions that opposite sides of a rectangle may be identified to form an ordinary torus. If we take $\lambda_1 = 1$ and λ_2 to be any irrational number, then the curve on the two-torus described by $\alpha(t)$ simply winds around the torus in a space filling curve. This is the origin of the idea of the *winding algorithm.*

The mapping from the p-torus to $SO(d)$ is onto, which guarantees that the image of the above curves are space filling. But there is a potential problem. We don't in general know how close to uniformly distributed the mapped curve is on $SO(d)$. The curve on the p-torus is equi-distributed, but in general the image may not be on $SO(d)$. In the two-plane formulation, this had been empirically a problem because some implementations of the torus method have given grand tours that behaved very non-unformly. In particular, the tour appears to dwell for a long time near certain axes while others appear rarely. Our experience with using this algorithm in the higher-dimensional formulation suggests empirically that this is less of a problem. While selected pairs of axes may appear relatively static, others are actually moving quite significantly. Thus the overall dynamic when visualizing high-dimensional plots is more satisfactory than when viewing 2-dimensional versions. Nonetheless, an adequate theoretical understanding of the overall dynamics of the torus algorithm is still an open question.

It is also worth pointing out that for a k-dimensional grand tour, one only needs the first k columns of a matrix in $SO(d)$. Thus from a computational point of view, there is no need to formulate the rotations in 2-planes that are to stay fixed. This simplifies the computational complexity somewhat, although it makes the overall algorithm more complicated with different algorithms for the computation of matrices in $SO(d)$ for each sub-dimension k. An interesting research question, posed by one of the referees, is whether overparametrizing improves the uniformity properties of the resulting grand tour. This also is an open question.

3.2 The Random Curve Algorithm. The key to the winding algorithm is the construction of the function $\alpha(t)$ which creates a space filling curve through the p-dimensional hypercube (or p-torus). The composition of α with β creates a space-filling curve through $SO(d)$. Alternate constructions which create a space filling curve through the p-dimensional hypercube can also be used to effect a space-filling curve through $SO(d)$. A simple way of doing this is to choose points at random in the hypercube. One initiates this algorithm by choosing two points at random in the hypercube, say $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, and creating a linear interpolant between them going from $\boldsymbol{s}_1$ to $\boldsymbol{s}_2$. Upon arriving at $\boldsymbol{s}_2$, we choose a third point, $\boldsymbol{s}_3$, and form the linear interpolant from $\boldsymbol{s}_2$ to $\boldsymbol{s}_3$. In general we have a sequence of points, $\boldsymbol{s}_j$, chosen randomly with linear interpolants between them. For any $\boldsymbol{x} \in [0, 2\pi]^p$ and any given, ϵ, eventually with probability one, for some $\boldsymbol{s}_j$, $\|\boldsymbol{x} - \boldsymbol{s}_j\| < \epsilon$. Thus eventually the random curve will pass arbitrarily close to any point in the p-hypercube.

Two caveats must be mentioned. Since opposite faces are identified, the shortest path between two points may not be through the hypercube but across a face of the cube. Since we are really interested in geodesics on the p-torus, one must not think in terms of staying strictly within the hypercube. This involves a slightly more complicated algorithm. In practice, a strict interpolation path within the hypercube also seems to be a quite satisfactory approach and will still pass within ϵ of any point with probability one. The second point to make is that, as with the winding algorithm, the random curve algorithm can, in principle, continue forever. Our original code in Mason Hypergraphics circa 1988 contained the winding algorithm. Our more recent codes in ExplorN and CrystalVision are based on the random curve algorithm. In practice, although theoretically these algorithms can go on forever, our experience has been that most structure in high-dimensional data shows up very rapidly, say within five or ten minutes of viewing the grand tour, and that it is unnecessary to continue the grand tour beyond that time.

3.3 The Fractal Curve Algorithm. In 1887, George Cantor demonstrated that any two finite-dimensional, smooth manifolds have the same cardinality regardless of their dimensions. In principle therefore $[0, 1]$ can be mapped bijectively onto $[0, 1]^d$. Many attempts to do this have been created most notable among these methods are the Peano curves and the Hilbert curves. The advantage of these curves are that they are fractal in character and hence, for a fixed fractal level, have a finite length and a fixed accuracy. By following a fractal curve through the hypercube, one can preselect an accuracy level and guarantee that the grand tour will terminate with a known time. To illustrate the computation, we will describe a two-dimensional Peano curve, a three-dimensional Hilbert curve, and our generalization to the d-dimensional Hilbert curve.

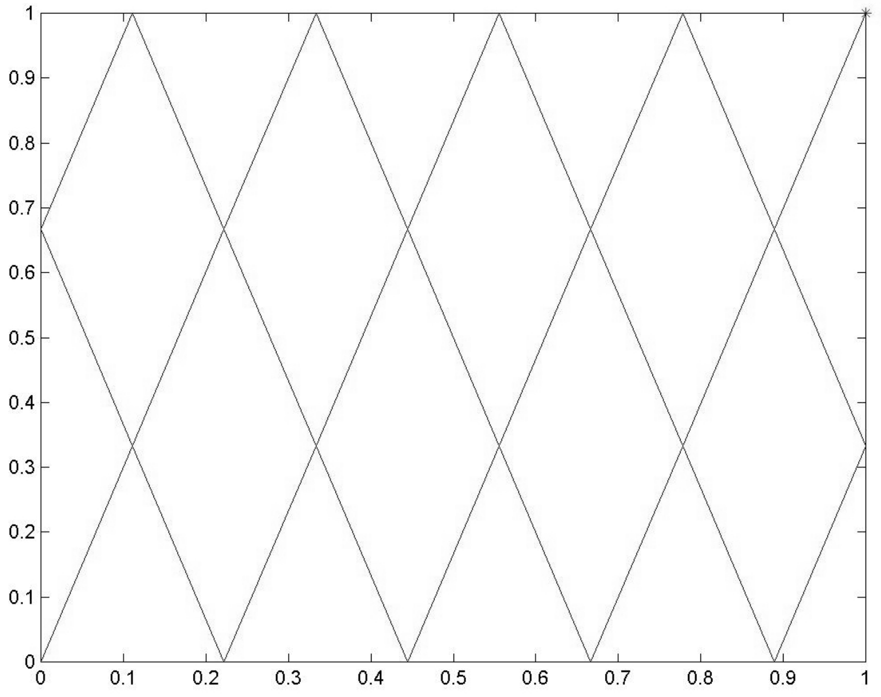

Figure 3.1 Third level Peano curve through $[0, 1]^2$.

We shall be concerned with ternary and octal expansions of fractional numbers between 0 and 1. We adopt the following notation for a ternary expansion:

$$0_3.t_1t_2t_3\cdots = \frac{t_1}{3} + \frac{t_2}{3^2} + \frac{t_3}{3^3} + \cdots,\ t_j = 0, 1 \text{ or } 2. \tag{3.5}$$

Similarly, for an octal expansion,

$$0_8.w_1w_2w_3\cdots = \tfrac{w_1}{8} + \tfrac{w_2}{8^2} + \tfrac{w_3}{8^3} + \cdots,\ w_j = 0, 1, 2, \ldots, 8. \tag{3.6}$$

The two-dimensional Peano curve of level m is given by the two vector

$$\beta(0_3.t_1t_2\cdots t_m) = (\,0_3.t_1(k^{t_2}t_3)(k^{t_2+t_4}t_5)\cdots \quad 0_3.(k^{t_1}t_2)(k^{t_1+t_3}t_4)\cdots). \tag{3.7}$$

Here $k(t) = 2 - t,\ t = 0, 1, 2$ and k^t is the t-th iterate of k. Of course the computation is carried out until m is satisfied. To make this computation concrete, let us consider a level 3 example. Implicitly the fourth position in the decimal expansion is 0. For illustration

$$\beta(0_3.021) = (\,0_3.0(k^2 1) \quad 0_3.(k^0 2)(k^1 0)\,) = (\,0_3.010 \quad 0_3.220\,) = \left(\tfrac{1}{9} \quad \tfrac{8}{9}\right).$$

The resulting sequence of points when joined determines a curve through in this case the square. Because there is considerable overlapping of points, a useful strategy is to join the midpoints of the line segments which results in the derived Peano curve in Figure 3.2.

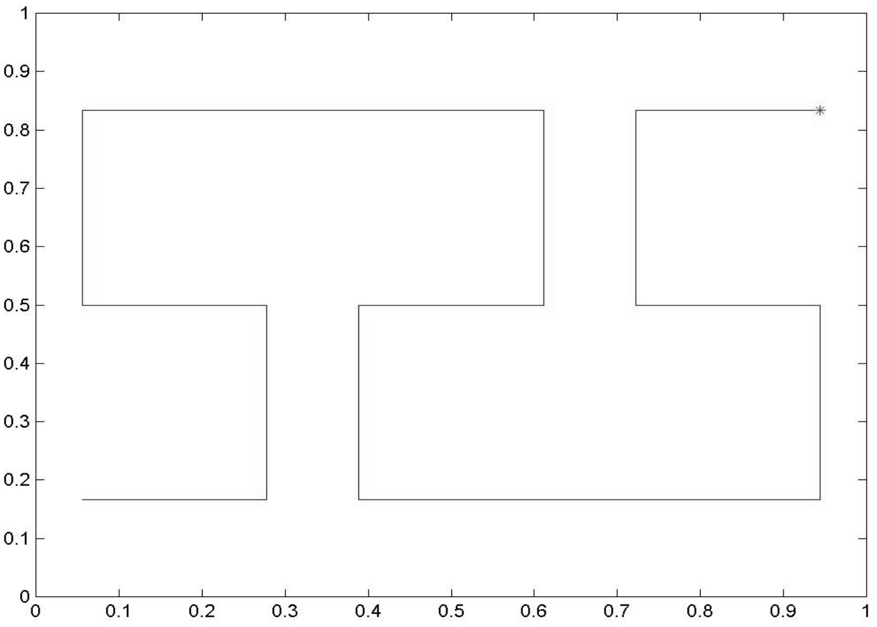

Figure 3.2 Derived Peano curve of level three.

The three-dimensional formulation of the Hilbert curve is somewhat more tedious. We define the following matrices:

$$H_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad H_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad H_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$H_3 = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \qquad H_4 = \begin{bmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad H_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$H_6 = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \qquad H_7 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix}.$$

We also construct the following column vectors:

$$h_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad h_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad h_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \qquad h_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad h_4 = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$$

$$h_5 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad h_6 = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} \qquad h_7 = \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}.$$

Then the three-dimensional Hilbert curve of level m is given by

$$\beta(0_8.w_1w_2\cdots w_m) = \sum_{j=1}^{m} \tfrac{1}{2^j} H_{w_0}H_{w_1}H_{w_2}\cdots H_{w_{j-1}}h_{w_j} \tag{3.8}$$

where H_{w_0} is the identity matrix. A three-dimensional level 2 Hilbert curve is given in Figure 3.3.

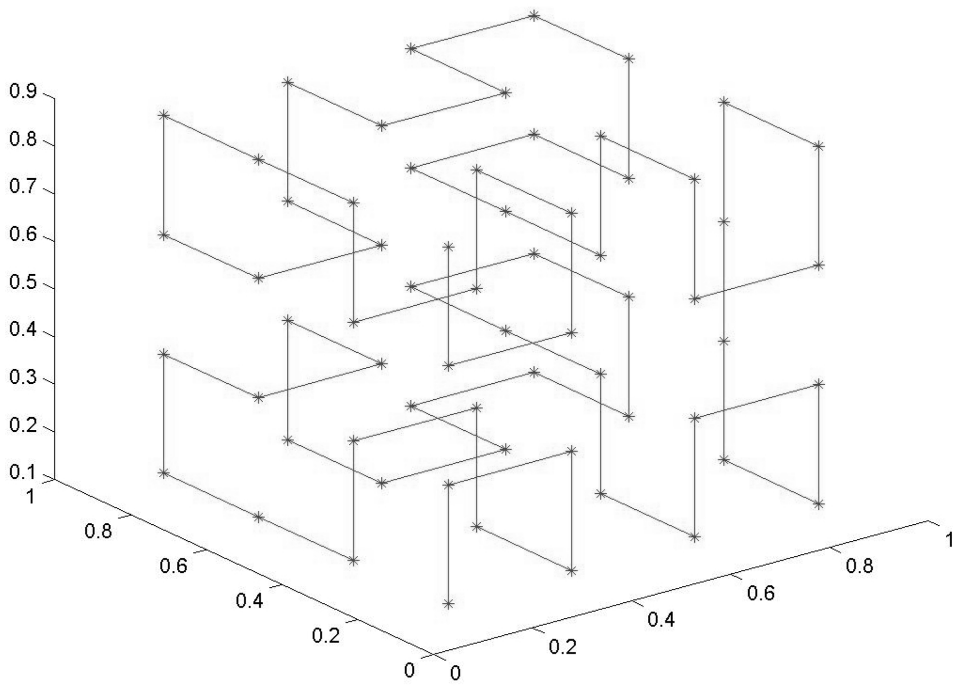

Figure 3.3 A 3-dimensional, level 2 Hilbert curve.

Both the Hilbert curve and the Peano curve can be extended to higher dimensions. See for example Solka et al. (1998). We give briefly the algorithm here. First let $k(x,n) = x$ if n is even and $k(x,n) = 2 - x$ if n is odd. Let $t_i = i$-th ternary digit of x. Let d be the dimension and let m be the level we desire. Let

$$\gamma_j(i) = k\Big(t_{di-(d-j)}, \quad \sum_{\delta=1,\, \delta \neq j \, mod \, d}^{di-(d-j)} t_\delta \Big) \,. \tag{3.9}$$

Then

$$\beta(x) = \left[\sum_{i=1}^{m} \frac{\gamma_1(i)}{3^i} \quad \sum_{i=1}^{m} \frac{\gamma_2(i)}{3^i} \quad \cdots \quad \sum_{i=1}^{m} \frac{\gamma_d(i)}{3^i} \right]. \tag{3.10}$$

As before this describes a series of points in $[0,1]^d$. These can be joined by line segments to form the general d-dimensional Peano curve and the midpoints of the line segments joined to form the d-dimensional derived Peano curve. By increasing the level m of the Peano curve we can come as close as desired to every point in the d-dimensional hypercube $[0,1]^d$. Further reading on Peano curves and related fractal curves can be found in Peano (1890), Steinhaus (1936) and Sagan (1994).

3.4 Andrews Plots, Parallel Coordinate Plots, and Tours. The Andrews plot (Andrews, 1972) was an early attempt to give a two-dimensional plot of multidimensional data. As such the Andrews plots have an interesting interpretation in connection with both parallel coordinates and the grand tour. To describe the Andrews plot, let $\boldsymbol{x}_i = (x_1^i, x_2^i, \ldots, x_d^i)$ be the data vectors, then the Andrews plot is given by

$$y_i(t) = \frac{x_1^i}{2^{1/2}} + x_2^i sin(t) + x_3^i cos(t) + x_4^i sin(2t) + x_5^i cos(2t) + \cdots. \tag{3.11}$$

Traditionally, the Andrews plot was constructed by plotting $y_i(t)$ versus t for $i = 1, \ldots, n.$ In brief the Andrews plot is a finite, hence, periodic Fourier expansion with the weights given by the components of the data vector. By plotting $y_i(t)$ versus t for every i in a static plot, one could group points that had similar curves. Because of the applicability of the Parseval relation, Andrews plots also have the property of preserving L_2 distances.

Andrews plots can be recognized in another sense. If one considers a one-dimensional plot of the $y_i(t)$ animated as a function of t (and this is a view recognized in Andrews, 1972), then the Andrews plot can be regarded as a one-dimensional tour. As a tour, the Andrews plot is a series of interpolations between various one-dimensional views of the data. In a similar way, the parallel coordinate plot can be viewed as a series of linear interpolations between one-dimensional projections of the data. Although these two plot devices have some similarity of interpretation in this sense, this interpretation misses the powerful geometric structure which motivated the parallel coordinate plot and lies at its intellectual roots.

Of course, the tour view of the Andrews plot also has a connection with the grand tour notion we have been examining. The Asimov-Buja grand tour was originally formulated as a series of projections into two-dimensional planes, not one-dimensional lines. The availability of multi-dimensional representations such as scatterplot matrices and parallel coordinates suggested the possibility full-dimensional grand tours. However, the two major criteria for grand tours are *continuity* and *space-filling*. The Andrews plot is a continuous tour, but as we shall see in the next section, it is not space filling.

3.5 A Pseudo-Grand Tour

As recently as 1990, the Andrews plot was characterized as a one-dimensional grand tour. See for example Crawford and Fall (1990). However, because of the familiar trigonometric identities,

$$sin(t_1 + t_2) = sin(t_1)\, cos(t_2) + cos(t_1)\, sin(t_2); \tag{3.12}$$

$$cos(t_1 + t_2) = cos(t_1)\, cos(t_2) - sin(t_1) sin(t_2) \tag{3.13}$$

and

$$cos^2(t) + sin^2(t) = 1, \tag{3.14}$$

Wegman and Shen (1993) showed that the Andrews plot was not a one-dimensional grand tour because it was not anywhere nearly space filling even in only one dimension. However, motivated by the Andrews plot, Wegman and Shen suggested a very fast algorithm for computing an approximate two-dimensional grand tour. Consider the d-dimensional data vector $\boldsymbol{x}_i = (x_1^i, x_2^i, \ldots, x_d^i)$. If d is not even augment the vector by one additional 0. We assume without loss of generality that d is even. Let

$$\boldsymbol{a}_1(t) = \sqrt{\tfrac{2}{d}}(sin(\lambda_1 t),\, cos(\lambda_1 t), \ldots, sin(\lambda_{\frac{d}{2}} t),\, cos(\lambda_{\frac{d}{2}} t)) \tag{3.15}$$

and

$$\boldsymbol{a}_2(t) = \sqrt{\tfrac{2}{d}}(cos(\lambda_1 t),\, -sin(\lambda_1), \ldots, cos(\lambda_{\frac{d}{2}} t),\, -sin(\lambda_{\frac{d}{2}} t)). \tag{3.16}$$

Here λ_j are as before linearly independent over the rationals. Note that

$$\|\boldsymbol{a}_1(t)\|_2^2 = \tfrac{2}{d} \sum_{j=1}^{\frac{d}{2}} (sin^2(\lambda_j t) + cos^2(\lambda_j t)) = 1,$$

$$\|\boldsymbol{a}_2(t)\|_2^2 = \tfrac{2}{d} \sum_{j=1}^{\frac{d}{2}} (cos^2(\lambda_j t) + (-sin)^2(\lambda_j t)) = 1,$$

and

$$< \boldsymbol{a}_1(t), \boldsymbol{a}_2(t) > \; = \tfrac{2}{d}\sum_{j=1}^{\frac{d}{2}} (sin(\lambda_j t)\, cos(\lambda_j t) - cos(\lambda_j t)\, sin(\lambda_j t)) = 0.$$

Thus $\boldsymbol{a}_1(t)$ and $\boldsymbol{a}_2(t)$ form an orthonormal basis for two-planes. They are not quite space filling because of the dependence between $cos(\lambda_j t)$ and $sin(\lambda_j t)$ implied by (3.14). However, the algorithm based on (3.15) and (3.16) is much more computationally convenient than the torus-based winding algorithms. Of course, it does not generalize to a full d-dimensional grand tour. A two-dimensional projection of the data onto the $\boldsymbol{a}_1\boldsymbol{a}_2$-plane can be accomplished by taking the inner product as in equation (3.2) with $j = 1, 2$.

4. Saturation Brushing

We have earlier mentioned saturation brushing as a technique for dealing with large data sets. A basic exposition of the saturation brushing idea can be found in Wegman and Luo (1997). While there is little in the way of mathematical underpinnings for the idea, it is appropriate for sake of completeness to briefly describe the idea here. When dealing with large data sets, overplotting becomes a serious problem. It is difficult to tell whether a pixel represents a single observation or perhaps hundreds or thousands of observations. The idea of saturation brushing is to desaturate a brushing color until it contains only a very small component of color and hence is very nearly black. Most modern computers have a so-called α-channel which allows for compositing of overplots. The α-channel is used computer graphics as a device for incorporating transparency. However, by using such a device to build up color intensity, we can obtain a visual indication of how much overplotting there is at a pixel. In effect, the brighter, more saturated a pixel is, the more overplotting.

5. Conclusions

We have used this combination of methods, i.e. parallel coordinate plots, scatterplot matrices, and full d-dimensional grand tours as well as partial grand tours, to analyze data sets ranging in dimension from 4 to 68 and ranging in data set size from as few as 22 points to as large as 280,000 points. An amazing amount of visual insight can be gained when these methods are applied in practical settings. Applications have included discovery of reasons for bank failures, discovery of hidden pricing mechanisms for commercial products such as cereals, discovery of physical structure of pi meson-proton collisions, creation of detection schemes for chemical and biological warfare agents, creation of the ability to detect buried landmines, demonstration of the impossibility of finding linear predictors of cost in a certain class of software development tools, and a host of other practical and interesting applications. We believe these combinations of techniques are both incredibly powerful from an applications point of view as well as having very interesting mathematical underpinnings.

Acknowledgements

The work of Dr. Wegman was supported by the Army Research Office under Grant DAAG55-98-1-0404, by the Office of Naval Research under Grant DAAD19-99-1-0314 administered by the Army Research Office, and by the Defense Advanced Research Projects Agency under Agreement 8905-48174 with The Johns-Hopkins University. The paper was completed while Dr. Wegman was an ASA/NSF/BLS Senior Research Fellow at the Bureau of Labor Statistics. Any opinions expressed in this paper are those of the authors and do not constitute policy of the Bureau of Labor Statistics. The work of Dr. Solka was supported by the NSWC ILIR Program through the Office of Naval Research and by the SWT Block Program of the Office of Naval Research.

The authors would like to express our gratitude to the referees who made insightful and very useful suggestions for improvements to the paper and our presentation of this material.

References

Andrews, D. F. (1972), "Plots of high dimensional data," *Biometrics,* 28, 125-136.

Asimov, D. (1985), "The grand tour: a tool for viewing multidimensional data," *SIAM J. Scient. Statist. Comput.*, 6, 128-143.

Buja, A. and Asimov, D. (1985), "Grand tour methods: an outline," *Computer Science and Statistics: Proceedings of the Seventeenth Symposium on the Interface*, D. Allen, ed., Amsterdam: North Holland, 63-67.

Buja, A., Hurley, C. and McDonald, J. A. (1986), "A data viewer for multivariate data," *Computer Science and Statistics: Proceedings of the Eighteenth Symposium on the Interface*, T. Boardman, ed., Alexandria, VA: American Statistical Association, 171-174.

Cook, D., Buja, A., and Cabrera, J. (1991), "Direction and motion control in the grand tour," *Computing Science and Statistics*, 23, 180-183.

Cook, D., Buja, A., Cabrera, J. and Swayne, D. (1993), *Grand Tour and Projection Pursuit* (a video), ASA Statistical Graphics Video Lending Library.

Cook, D., Buja, A., Cabrera, J., and Hurley, C. (1995), "Grand tour and projection pursuit," *Journal of Computational and Graphical Statistics*, 4(3), 155-172.

Cook, D. and Buja, A. (1997) "Manual controls for high-dimensional data projections," *J. Computat. Graph. Statist.*, 6, 464-480.

Crawford, S. L. and Fall, T. C. (1990), “Projection pursuit techniques for visualizing high-dimensional data sets," in *Visualization in Scientific Computing*, (G. M. Nielson, B. Shrivers, L. J. Rosenblum, editors), 94-108, Los Alamitos, CA: IEEE Computer Society Press.

Furnas, G. and Buja, A. (1994), "Prosection views: Dimensional inference through sections and projections," *J. Computat. Graph. Statist.*, 3, 323 - 353.

Hurley, C. and Buja, A. (1990), "Analyzing high dimensional data with motion graphics," *SIAM J. Sci. Statist. Comput.*, 11, 1193-1211.

Inselberg, A. (1985), "The plane with parallel coordinates," *The Visual Computer*, 1, 69-91.

Peano, G. (1890), "Sur une courbe qui remplit toute une aire plane," *Math. Annln.*, 36, 157-160.

Miller, J. J. and Wegman, E. J. (1991), "Construction of line densities for parallel coordinate plots," in *Computing and Graphics in Statistics* (A. Buja and P. A. Tukey, eds.), New York: Springer-Verlag, 107-123.

Sagan, H. (1994), *Space-Filling Curves*, New York: Springer-Verlag.

Solka, J. L., Wegman, E. J., Reid, L. and Poston, W. L. (1998), "Explorations of the space of orthogonal transformations from R^p to R^p using space-filling curves," *Computing Science and Statistics*, 30, 494-498.

Solka, J. L., Wegman, E. J., Rogers, G. W., and Poston, W. L. (1997), "Parallel coordinate plot analysis of polarimetric NASA/JPL AIRSAR imagery," *Automatic Target Recognition VII - Proceedings of SPIE*, 3069, 175-184.

Steinhaus, H. (1936), "La courbe de Peano et les fonctions independantes," C. R. Acad. Sci., Paris, 202, 1961-1963.

Wegman, E. J. (1990), "Hyperdimensional data analysis using parallel coordinates," *Journal of the American Statistical Association*, 85, 664-675.

Wegman, E. J. (1991), "The grand tour in k-dimensions," *Computing Science and Statistics: Proceedings of the 22nd Symposium on the Interface*, 127-136.

Wegman, E. J. (2000), *Geometric Methods in Statistics*, Lecture Notes available at http://www.galaxy.gmu.edu.

Wegman, E. J. and Bolorforoush, M. (1988), "On some graphical representations of multivariate data," *Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface*, 121-126.

Wegman, E. J. and Carr, D. B. (1993), "Statistical graphics and visualization," in *Handbook of Statistics 9: Computational Statistics*, (Rao, C. R., ed.), Amsterdam: North Holland, 857-958.

Wegman, E. J., Carr, D. B. and Luo, Q. (1993) "Visualizing multivariate data," in *Multivariate Analysis: Future Directions*, (Rao, C. R., ed.), Amsterdam: North Holland, 423-466.

Wegman, E. J. and Luo, Q. (1997), "High dimensional clustering using parallel coordinates and the grand tour," *Computing Science and Statistics*, 28, 352-360, republished in *Classification and Knowledge Organization*, (R. Klar and O. Opitz, eds.), Berlin: Springer-Verlag, 93-101, 1997.

Wegman, E. J., Poston, W. L., and Solka, J. L. (1998), "Image grand tour," *Automatic Target Recognition VIII - Proceedings of SPIE*, 3371, 286-294.

Wegman, E. J. and Shen, J. (1993), "Three-dimensional Andrews plots and the grand tour," *Computing Science and Statistics*, 25, 284-288.

Wilhelm, A. F. X., Wegman, E. J., and Symanzik, J. (1999), "Visual clustering and classification: The Oronsay particle size data set revisted," *Computational Statistics*, 14(1), 109-146.

Software References

Mason Hypergraphics, copyright (c) 1988, 1989 by Edward J. Wegman and Masood Bolorforoush, a MS-DOS package for high-dimensional data analysis. Originally programmed in Turbo-Pascal, the sofware contained parallel coordinate plots, scatterplot matrices, grand-tour linked to parallel coordinates, anaglyph stereo 3-D scatterplots, and parallel coordinate density plots. It is still available at ftp://www.galaxy.gmu.edu/pub/software/hypergra.zip.

ExplorN, copyright (c) 1992, by Daniel B. Carr, Qiang Luo, Edward J. Wegman, and Ji Shen, a UNIX package for Silicon Graphics workstations incorporating scatterplot matrices, stereo ray glyph plots, parallel coordinates, and the d-dimensional grand tour. Recent versions also include saturation brushing. The code was done using Silicon Graphics proprietary GL graphics subroutines and, hence, only runs on Silicon Graphics workstations. The package is available at ftp://www.galaxy.gmu.edu/pub/software/ExplorN_v1.tar.

CrystalVision, copyright (c) 2000 by Crystal Data Technologies, is a Windows 95/98/NT package for Wintel computers. The software incorporates scatterplot matrices, stereoscopic 3-D scatterplots using Crystal Eyes technology, parallel coordinate plots, d-dimensional grand tours and partial grand tours, saturation brushing, and density plots. The code was constructed using openGL and will run on any modern Wintel computer. A demonstration version of CrystalVision is available at ftp://www.galaxy.gmu.edu/pub/software/CrystalVisionDemo.exe