Statistical Graphics and Visualization

Edward J. Wegman
Daniel B. Carr

Center for Computational Statistics
George Mason University

Statistical Graphics and Visualization

Table of Contents

Part II: Graphics Techniques

7. Principles of Graphics Construction
7.1 Terminology
7.2 Perception
7.3 Statistical Graphics versus Realistic Imaging
8. One-Dimensional Variables
8.1 Density Estimation
8.2 Binning Methods in One Dimension
8.3 Quantile Plots
8.4 Box Plots
8.5 EDA Views and Transformations of Data
8.6 Labeled One-Dimensional Plots
9. Visualizing Multidimensional Data
9.1 Scatterplots, Rotating Scatterplots and Scatterplot Matrices
9.2 Glyphs and Icons
9.3 Parallel Coordinate Plots
9.4 Andrews Plots
9.5 Multivariate Density Estimation
9.6 Isopleths, Ridges and Skeletons
9.7 Generalized Rotations and the Grand Tour

Statistical Graphics and Visualization

1. Introduction

This paper focuses on graphics and visualization methods for statistical analysis. The traditional use of graphics in statistics had been principally relegated to simple two-dimensional graphics such as scatterplots and histograms. However, modern computer graphics methods introduce a host of new techniques which literally allow an explosion of new methods for a much more geometric understanding of statistical and scientific data. Computer graphics allows an easy leap to visualizing three-dimensional data and to using animation, lighting, rendering, transparency and a host of other techniques. This paper is intended to cover some of the basic elements of computer-based graphics and to give an overview of some of the resulting statistical graphics and visualization methods.

There are some excellent treatments of general computer graphics, the classic being Foley et al. (1990) which is encyclopedic in character. A helpful basic introduction to computer graphics is Plastock and Kalley (1986). The former text is some 1172 pages long and does not treat any of the elements of statistical graphics or visualization. It is clear that in the present article, there are a number of items which we simply will not cover. To be precise, among other topics, we will omit coverage of the mechanics of computer graphics, that is the hardware involved.

It is sufficient to say that there are two broad classes of hardware, vector graphics and raster graphics. In the former, points and lines are drawn with the cathode ray tube's (CRT) electron beam being deflected from point to point either drawing or not drawing lines between points. In the latter, which has become virtually the standard, the electron beam(s) scans the screen much like in a television set from top left to bottom right. Each of a finite number of locations on the screen is known as a *pixel* (picture element) and each pixel has a memory location associated with it. The memory complexity may be as simple as two bits (on and off) for simple monochrome systems to as many as 96 bits or more for complex memory systems that admit a broad range of colors, transparency and 3-D effects. Those readers familiar with personal computer standards will be aware of graphics standards such as EGA and VGA and others. The higher-end standard graphics for PCs is typically 640×400 pixels. These are horizontal by vertical. Workstations tend to have higher resolutions with 1024×768 and 1280×1024 typical. Standards change relatively rapidly and the interested reader is best advised to follow trade magazines to keep up with the changing hardware configurations. Publications such as ***PCWeek***, ***MacWeek***, ***Silicon Graphics World***, and ***NeXTworld*** are examples. For the most part, free subscriptions to these periodicals are available to professionals with some level of decision making authority in computer purchases.

We will also not attempt to cover software standards for computer graphics. There are a number of systems of subroutines available. These encode a number of graphics primitives sometimes in a hierarchical fashion so that the graphics programmer is freed from the chore of actually handling the mechanics of drawing points and lines on the screen. The earliest system to be standardized was the so-called GKS (graphics kernel system) and was later extended to three dimensions by GKS-3D. Another common system is PHIGS (programmer's hierarchical interactive graphics system). This was extended to three dimensions by PHIGS+. Another

system which is becoming popular is the Silicon Graphics proprietary system known as GL (graphics library). Also of considerable interest is the X-windows graphics standard originated at MIT. X-windows has attained a large amount of attention because of relatively inexpensive hardware support in the form of graphics terminals known as X-terminals. Hardware vendors usually implement one or more of these graphics standards in their graphics-oriented UNIX workstations. Historically, PC computers have been somewhat less standardized in graphics systems principally because they lacked the CPU power and memory to support complex graphics operations.

This paper is organized into two main parts. In the first we will cover the basics of computer graphics. In the second part, we will focus on a variety of the traditional statistical graphics techniques. We use “traditional" advisedly since the majority of these methods have only come into common usage during the last five to ten years.

2. Two-Dimensional Graphics Transforms, Viewing Transforms and Screen Coordinates

Remarkable effects have been achieved with static plots on paper. The excellent discussions by Tufte (1983, 1990) give many examples of this fact. Nonetheless, computer graphics has the ability to add a dynamic character to the graphic display which, indeed, is the very heart of recent developments in statistical graphics and visualization. One can conceive of motion in two ways: 1) move an object within a fixed coordinate system and 2) change the coordinate system relative to an object fixed in space. In the former case, we shall refer to this as a *geometric transformation* while in the latter we shall speak of a *coordinate transformation*. In a sense these are opposite transformations, that is moving an object in a coordinate system is equivalent to moving the coordinate system in the opposite sense. On motion picture sound stages for instance, an automobile can be made to appear to be moving either by having the automobile move forward (say to the left) against a stationary backdrop or by having the backdrop move backward (to the right) in back of a stationary automobile.

2.1 Geometric Transformations:

2.1.1 Translation: If $p = (x, y)$ is a point in a Cartesian coordinate system and that point is to be translated by an amount t_x in the x-direction and an amount t_y in the y direction, then the translated point p' equals (x', y') where $x' = x + t_x$ and $y' = y + t_y$. See Figure 2.1. We denote this transformation by T_v where $v = (t_x, t_y)$ so that $p' = T_v(p)$.

2.1.2 Rotations: While translations are straightforward, rotations take a bit more effort. Consider a point (x, y) which may be written in polar coordinates as (r, γ), so that $x = r\cos(\gamma)$ and $y = r\sin(\gamma)$. As illustated in Figure 2.2, if the point, (x, y), is rotated through an angle of θ, then $x' = r\cos(\gamma + \theta)$ and $y' = r\sin(\gamma + \theta)$. We may use the elementary trigonometric identities

$$\cos(\gamma + \theta) = \cos(\gamma)\cos(\theta) - \sin(\gamma)\sin(\theta)$$

and

$$\sin(\gamma + \theta) = \sin(\gamma)\cos(\theta) + \cos(\gamma)\sin(\theta)$$

to obtain

$$x' = r\cos(\gamma)\cos(\theta) - r\sin(\gamma)\sin(\theta)$$

and

$$y' = r\cos(\gamma)\sin(\theta) + r\sin(\gamma)\cos(\theta).$$

Thus a simple substitution yields $x' = x\cos(\theta) - y\sin(\theta)$ and $y' = x\sin(\theta) + y\cos(\theta)$. We denote this transformation by R_θ so that if $p = (x, y)$ and $p' = (x', y')$, then $p' = R_\theta(p)$.

2.1.3 Scaling: If s_x and s_y are positive scaling constants, then $x' = s_x \cdot x$ and $y' = s_y \cdot y$ are the rescaled version of x and y. We can construct a scaling transformation S_{s_x,s_y} where, in the obvious notation, $p' = S_{s_x,s_y}(p)$. If both $s_x = s_y = s$, then the scaling is *homogeneous*. If $s > 1$, then $S_{s,s}$ is a *magnification* and if $s < 1$, $S_{s,s}$ is a *reduction*. Figure 2.3 illustrates a scaling in which $s_x = s_y = 2$.

2.1.4 Inverse Transformations: For any of the above transformations we can find an inverse. Thus $T_v^{-1} = T_{-v}$, $R_\theta^{-1} = R_{-\theta}$ and $S_{s_x,s_y}^{-1} = S_{1/s_x,1/s_y}$.

2.2 Coordinate Transformations:

As indicated above the coordinate transformation is the inverse of the geometric transformation. In all of the descriptions below, we will let $p' = (x', y')$ represent the point $p = (x, y)$ in the new coordinate system. Then we have the following explicit results.

(2.1) Translation: $x' = x - t_x$ and $y' = y - t_y$.
Rotations: $x' = x\cos(\theta) + y\sin(\theta)$ and $y' = -x\sin(\theta) + y\cos(\theta)$.
Scaling: $x' = (1/s_x) \cdot x$ and $y' = (1/s_y) \cdot y$.

2.3 Matrix Expressions of the Basic Transformations:

The basic rotation and scaling transformations can easily be written in matrix form, in particular, for Geometric Transformations

(2.2)
$$R_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

and

(2.3)
$$S_{s_x,s_y} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}.$$

Translations cannot be expressed as a 2×2 matrix, but can be formulated as a 3×3 matrix with the help of natural homogeneous coordinates.

2.4 Natural Homogeneous Coordinates:

The natural homogeneous coordinate representation arises in the analytic expression of projective geometry. In projective geometry, the non-intersection of parallel lines axiom is replaced by an axiom stating that any two lines meet at a single point. To conceptualize this axiom, parallel lines intersect at the point at infinity called an *ideal point*. There is a distinct ideal point for every slope. The collection of ideal points is the *ideal line*. The traditional representation of a point in Euclidian geometry is an ordered pair. However, the usual coordinate pair, (x, y), is not sufficient to represent ideal points. We represent points in the projective plane by triples (x, y, z). Consider two distinct parallel lines having equations

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

Simultaneous solution yields $(c - c') \cdot z = 0$ so that $z = 0$ will describe an ideal point. 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 homogeneous coordinates representation of a point (x, y) in the usual Cartesian coordinates lying on line, $ax + by + c = 0$. Notice that if (px, py, p) is any multiple of (x, y, 1) on $ax + by + c = 0$, we have

$$a \cdot px + b \cdot py + cp = p \cdot (ax + by + c) = p \cdot 0 = 0.$$

Thus the triple (px, py, p) equally well represents the Cartesian point (x, y) lying on $ax + by + c = 0$ so that the representation in natural homogeneous coordinates is not unique. However, if p 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 which reiterates the fundamental duality between points and lines in the projective plane.

2.5 Matrix Expressions in Natural Homogeneous Coordinates:

We now represent any point, (x, y) in the Euclidian plane by the triple (x, y, 1). Then we may write T_v as

$$T_v = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

so that

$$\mathrm{p}' = \begin{bmatrix} \mathrm{x} + \mathrm{t}_x \\ \mathrm{y} + \mathrm{t}_y \\ 1 \end{bmatrix} = \mathrm{T}_v \times \mathrm{p} = \begin{bmatrix} 1 & 0 & \mathrm{t}_x \\ 0 & 1 & \mathrm{t}_y \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ 1 \end{bmatrix}. \tag{2.5}$$

Thus using homogeneous coordinates allows us to represent a translation as a matrix transformation. Similarly rotations can be represented in natural homogeneous coordinates. Consider the matrix now expressed in natural homogeneous coordinates

$$\mathrm{R}_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

so that

$$\mathrm{p}' = \begin{bmatrix} \mathrm{x}\cos(\theta) - \mathrm{y}\sin(\theta) \\ \mathrm{x}\sin(\theta) + \mathrm{y}\cos(\theta) \\ 1 \end{bmatrix} = \mathrm{R}_\theta \times \mathrm{p} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ 1 \end{bmatrix}. \tag{2.7}$$

Finally scaling transformations may also be written in natural homogeneous coordinates by considering

$$\mathrm{S}_{s_x,s_y} = \begin{bmatrix} \mathrm{s}_x & 0 & 0 \\ 0 & \mathrm{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

so that

$$\mathrm{p}' = \begin{bmatrix} \mathrm{s}_x \cdot \mathrm{x} \\ \mathrm{s}_y \cdot \mathrm{y} \\ 1 \end{bmatrix} = \mathrm{S}_{s_x,s_y} \times \mathrm{p} = \begin{bmatrix} \mathrm{s}_x & 0 & 0 \\ 0 & \mathrm{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ 1 \end{bmatrix}. \tag{2.9}$$

Finally it is worth noting that reflections about either the x or the y axis may be expressed in natural homogeneous coordinates where in the obvious notation we have

$$M_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } M_y = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.10}$$

Because all of these matrices are 3×3 matrices they may be concatenated in any order to obtain an affine transformation in the plane. Even though we have not used any properties of projective geometry, the natural homogeneous coordinate representation is a powerful device for rendering a simple computation for affine transformations.

2.6 Viewing Transformations and Screen Coordinates:

Displaying an image on a computer screen is accomplished by two additional transformations. The graphic we wish to display has been described in *world coordinate system* (WCS), a right-handed Cartesian coordinate system. However, we will be unable to display the whole graphic image so that we must look at it through a rectangular window. Let x^w_{min} and x^w_{max} represent respectively the minimum and maximum x values of the window we wish to represent and similarly let y^w_{min} and y^w_{max} represent the minimum and maximum y values of the window. We wish to map this into what is called the *normalized device coordinate system* (NDCS) which is a right-handed coordinate system on the unit square [0, 1] $\times$ [0, 1]. We may write the normalization transformation, N, mapping the window in the WCS into the NDCS by

$$N = \begin{bmatrix} s_x & 0 & -s_x \cdot x^w_{min} \\ 0 & s_y & -s_y \cdot y^w_{min} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

where $s_x = \frac{1}{x^w_{max} - x^w_{min}}$ and $s_y = \frac{1}{y^w_{max} - y^w_{min}}$. This normalization transformation is device independent and can be made with any computer graphics device. A second linear transformation takes the NDCS into the specific viewport for a particular computer graphics system. The screen coordinate system is normally a right-handed coordinate system with (0, 0) located at the bottom left-hand corner of the screen. In a high-resolution system with 1280×1024 there will be 1280 columns of pixels and 1024 rows of pixels. Thus the upper right-hand corner would be the point (1023, 1279). However the viewport may be some rectangular sub-area of the screen. If x^v_{min} and x^v_{max} represent the minimum and maximum x values of the screen coordinates for the viewport and similarly y^v_{min} and y^v_{max} the minimum and maximum y screen coordinate values, then the device transformation D is given by

$$D = \begin{bmatrix} s'_x & 0 & x^v_{min} \\ 0 & s'_y & y^v_{min} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

where $s'_x = x^v_{max} - x^v_{min}$ and $s'_y = y^v_{max} - y^v_{min}$. The composite transformation $V = D \times N$ is the viewing transformation moving the graphics image from the WCS to the screen coordinate system. The viewing transformation can be written as a single matrix

$$V = \begin{bmatrix} s_x & 0 & -s_x \cdot x^w_{min} + x^v_{min} \\ 0 & s_y & -s_y \cdot y^w_{min} + y^v_{min} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

where now $s_x = \frac{x^v_{max} - x^v_{min}}{x^w_{max} - x^w_{min}}$ and $s_y = \frac{y^v_{max} - y^v_{min}}{y^w_{max} - y^w_{min}}$. In general, the WCS is a

continuous coordinate system and will have an infinite number of positions available. In practice this is limited to the floating point precision of the computer. In any case, the number of pixel locations on the screen is finite, so that the location specified by the viewing transformation is rounded to the nearest integer location. This will give rise to distortions such as the *staircasing* effect. Sufficiently high resolution usually minimizes this although other approaches include shading of offending pixels either lighter or darker than they would normally be. This latter technique is known as *antialiasing* which, in many high performance workstations, is implemented in hardware. The individual and composite transformations are illustrated in Figure 2.4.

2.7 Aspect Ratio:

The viewing transformation, V, involves scaling in both the x and y directions. If $s_x \neq s_y$, then undesirable distortions can arise. For example a rotating figure will appear to flatten or elongate during different phases of its rotation. The *aspect ratio* of a window or viewport is defined as $a = \frac{x_{max} - x_{min}}{y_{max} - y_{min}}$. If the aspect ratio, a_w, of the window equals the aspect ratio, a_v, of the viewport, then $s_x = s_y$ and no distortion occurs. In general, we wish to design the window and the viewport so that the aspect ratios are the same.

3. Three-Dimensional Graphics Transformations and Projections

Statistical graphics and visualization gain much of their power by moving from the two-dimensional setting into the three-dimensional. Straightforward analogs of the two-dimensional transforms discussed in the previous section are available. In the discussion below we shall omit the development of the basic equations and the re-expression of those equations in ordinary 3×3 matrix form and proceed directly to the expression in natural homogeneous coordinates for 3-dimensions in which a point is represented by a four vector. As before we can distinguish between geometric transformations and coordinate transformations.

3.1 Geometric Transformations:

An object will be regarded as a collection of points $P = \{(x, y, z, 1)\}$. Note in general when we write a transformation as a matrix we must think of a point as a four element column vector so that the matrix and column vector are conformable. However, in most cases there will be no confusion if in the more casual narrative we represent a point in three-dimensional Euclidian space as a 4-tuple in natural homogeneous coordinates. The context should make it clear when the 4-tuple must be regarded as a column vector.

3.1.1 Translation: We now consider a point (x, y, z, 1) to be translated by a transformation T_v according to the following scheme:

$$x' = x + t_x$$
$$y' = y + t_y$$
$$z' = y + t_z.$$

Then the required homogeneous coordinate matrix transformation is given by

$$p' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = T_v \times p = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.1}$$

3.1.2 Scaling: Scaling, as with translation is very similar to the two-dimensional transformation. The scale factor may be taken to be different in each dimension as we had done earlier. Thus a scale transformation, call it S_{s_x,s_y,s_z} can be written as

$$p' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = S_{s_x,s_y,s_z} \times p = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.2}$$

3.1.3 Rotations: Rotations are only slightly more complicated in three dimensions. Rotations are given about the three axes, x, y and z, individually or, more precisely in the three planes, x-y, x-z and y-z, individually. We let θ_{xy}, θ_{xz} and θ_{yz} be the rotation angles in those planes respectively. The slightly more cumbersome notation will be used here because later in our discussion when dealing with hyperdimensional data we shall wish to talk about generalized rotations in hyperspace. Because in d dimensions, there are $d - 2$ orthogonal axes

to any two-dimensional plane, it will not make sense to talk about rotation about an axis in dimensions higher than three. Given that we are rotating individually in a two-dimensional plane we can use formulae analogous to those derived earlier.

Rotation in the x-y plane (about the z axis):

$$\text{(3.3)} \qquad \mathrm{p}' = \begin{bmatrix} \mathrm{x}' \\ \mathrm{y}' \\ \mathrm{z}' \\ 1 \end{bmatrix} = \mathrm{R}_{\theta_{xy}} \times \mathrm{p} = \begin{bmatrix} \cos(\theta_{xy}) & -\sin(\theta_{xy}) & 0 & 0 \\ \sin(\theta_{xy}) & \cos(\theta_{xy}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ 1 \end{bmatrix}.$$

Rotation in the x-z plane (about the y axis):

$$\text{(3.4)} \qquad \mathrm{p}' = \begin{bmatrix} \mathrm{x}' \\ \mathrm{y}' \\ \mathrm{z}' \\ 1 \end{bmatrix} = \mathrm{R}_{\theta_{xz}} \times \mathrm{p} = \begin{bmatrix} \cos(\theta_{xz}) & 0 & \sin(\theta_{xz}) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_{xz}) & 0 & \cos(\theta_{xz}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ 1 \end{bmatrix}.$$

Rotation in the y-z plane (about the x axis):

$$\text{(3.5)} \qquad \mathrm{p}' = \begin{bmatrix} \mathrm{x}' \\ \mathrm{y}' \\ \mathrm{z}' \\ 1 \end{bmatrix} = \mathrm{R}_{\theta_{yz}} \times \mathrm{p} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_{yz}) & -\sin(\theta_{yz}) & 0 \\ 0 & \sin(\theta_{yz}) & \cos(\theta_{yz}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \\ 1 \end{bmatrix}.$$

The direction of the positive angle of rotation is chosen according to the right-hand rule that we have adopted for the our default coordinate systems. This explains what appears at first glance to be a misplaced sign in the expression for rotation in the x-z plane. When discussing general d-dimensional rotations later on we shall forgo this convention in order to preserve

symmetry in the matrix expressions. General rotations can be constructed by sequential rotations in the appropriate canonical planes. Composite matrices for the general rotation in higher dimensions can be computed in closed form. However, the expressions are extremely messy, especially when computing general rotations beyond 3-dimensions. It is to be noted that for this reason, many statistical graphics packages allow rotation of three-dimensional scatterplots only in one plane at a time. Perhaps it is sufficiently obvious to be left unsaid, but for many applications such as rotating scatterplots it is desirable to have a constant rotation speed. Thus there is a fixed triple, (θ_{xy}, θ_{xz}, θ_{yz}), which will be used for every point and for every instance of rotation. The cosines and sines take on simple numerical values and the generalized rotation matrix is a simple 4×4 matrix of numerical values. This is to be computed outside of the loop for the data points and outside the loop for the time instances. Although obvious, the straightforward implementation of the rotation matrix formulae could lead (and has led some programmers) to embed the general matrix within both loops leading to recomputation of the sines and cosines for every data point and every time instance. It should also be noted that multiplication of rotation matrices is not commutative, so that the order of the rotations matters in the final result.

3.2 Coordinate Transformations:

As in the two-dimensional case, there is a fundamental duality between the geometric transformations and the coordinate transformation. We may compute the coordinate transformation by the following simple duality. We use the superscripts c and g to refer respectively to the coordinate and geometric transformation respectively. The table below gives the equivalences.

	Coordinate Transformation	**Geometric Transformation**
Translation	T^c_v	T^g_{-v}
Rotation	R^c_θ	$\mathrm{R}^g_{-\theta}$
Scaling	$\mathrm{S}^c_{s_x,s_y,s_z}$	$\mathrm{S}^g_{1/s_x,1/s_y,1/s_z}$

3.3 Alignments:

We have seen that elementary affine transformations can be constructed from simple primitive operations involving translations, rotations and scaling. For the most part, this will be all that we require for manipulation of three-dimensional computer graphics objects. However, the WCS is now a three-dimensional coordinate system, rather than the two-dimensional system we dealt with in the previous section. In order to translate the WCS to the computer screen, we will need to form a view plane. Typically we will want the view plane to be the x-y plane. However, in order to accomplish this, it is on occasion necessary to align an arbitrary view plane with the x-y plane. Alignment can be thought of as a simple process of two sequential rotations as illustrated in Figure 3.1.

In the description that follows let $\mathbf{e}_{\boldsymbol{x}} = (1,\ 0,\ 0,\ 1)$, $\mathbf{e}_{\boldsymbol{y}} = (0,\ 1,\ 0,\ 1)$ and $\mathbf{e}_{\boldsymbol{z}} = (0,\ 0,\ 1,\ 1)$ be the unit vectors in respectively the x-, y- and z-directions. A plane can be specified by a *reference point*, that is a point in the plane and the *normal vector*, $\mathbf{n} = \mathrm{n}_x\mathbf{e}_{\boldsymbol{x}} + \mathrm{n}_y\mathbf{e}_{\boldsymbol{y}} + \mathrm{n}_z\mathbf{e}_{\boldsymbol{z}}$. Consider a vector $\mathbf{v} = \mathrm{a} \cdot \mathbf{e}_{\boldsymbol{x}} + \mathrm{b} \cdot \mathbf{e}_{\boldsymbol{y}} + \mathrm{c} \cdot \mathbf{e}_{\boldsymbol{z}}$. We wish to align $\mathbf{v}$ with $\mathbf{e}_{\boldsymbol{z}}$. To do this we first rotate in the y-z plane (about the x-axis) until $\mathbf{v}$ is rotated into the

z-positive part of the x-z plane. We then rotate in the x-z plane (about the y-axis) until the resultant vector is aligned with $\mathbf{e}_z$. This is accomplished by the following matrix

$$A_v = \begin{bmatrix} \frac{\lambda}{|\mathbf{v}|} & \frac{-ab}{\lambda|\mathbf{v}|} & \frac{-ac}{\lambda|\mathbf{v}|} & 0 \\ 0 & \frac{c}{\lambda} & \frac{-b}{\lambda} & 0 \\ \frac{a}{|\mathbf{v}|} & \frac{b}{\lambda|\mathbf{v}|} & \frac{c}{\lambda|\mathbf{v}|} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.6}$$

Here $|\mathbf{v}| = \sqrt{a^2 + b^2 + c^2}$ and $\lambda = \sqrt{b^2 + c^2}$. Notice that if $b = c = 0$ so that $\lambda = 0$, then $\mathbf{v}$ is aligned with the $\mathbf{e}_x$ and a simple rotation of 90° will align $\mathbf{v}$ with $\mathbf{e}_z$. That matrix is then

$$A_z = \begin{bmatrix} 0 & 0 & \frac{-a}{|a|} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{a}{|a|} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.7}$$

We shall see momentarily how alignments are used to facilitate general projections.

3.4 Projections:

Let us now turn to the notion of a projection. As indicated above, the 3-D WCS must be transformed by some device to a 2-D view plane so that within the 2-D view plane we can describe a window and thence make the transformation to the NDCS. We do this by means of a projection. There are two basic methods of projection which we may use, *perspective projection* and *parallel projection*. The former method is used to render relatively “realistic" graphics images which provide *perspective foreshortening* and *vanishing points*. These provide the depth cues needed for visualizing complicated structures. These features, however, distort the true size and shape of an object and, hence, are not always useful for purposes of relative size comparison such as we might wish to do for more quantitative data analysis. We discuss the basic algorithms for both.

3.3.1 Perspective Projections:

The techniques of perspective projections arise from the methods historically used by artists. The description below is cast in terms of a projection of a 3-D object onto a 2-D projection plane although higher dimensional analogs are straight-forward. In most cases we would prefer a parallel projection for moving from high dimensions to two- or three-dimensions. The eye is placed at a viewing point called the *center of projection.* At some distance from the eye is a plane called the *view plane.* A ray emanating from the center of projection to a point, p, on the object to be projected will intersect the view plane. The ray is called a *projector* and the intersection of the projector with the view plane is called the *(perspective) projection*, p' of p on the view plane. A standard setup is to take the view plane as the x-y plane and locate the center of projection, $C = (0, 0, -d, 1)$. See Figure 3.2.

With this setup, it is a simple matter to determine $p' = (x', y', 0, 1)$ by using proportions of similar triangles. In particular,

$$x' = \frac{d \cdot x}{z+d}, \quad y' = \frac{d \cdot y}{z+d}, \quad z = 0.$$

This can be formulated in homogeneous coordinate matrix form as

$$p' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} d \cdot x \\ d \cdot y \\ 0 \\ z + d \end{bmatrix} = \text{Per}_{\mathbf{e}_z, C} \times p = \begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & d \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.8}$$

Here $\mathbf{e}_z$ is the normal to the view plane and $C = (0, 0, -d, 1)$ is the center of projection. The more general configuration is a projection plane $\mathbf{n} = n_x \mathbf{e}_x + n_y \mathbf{e}_y + n_z \mathbf{e}_z$ with reference point $p_0 = (x_0, y_0, z_0, 1)$ and center of projection $C = (a, b, c, 1)$. In that case the perspective projection of a general point, p, is given by

$$p' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \text{Per}_{\mathbf{n}, p_0, C} \times p = \begin{bmatrix} d + an_x & an_y & an_z & -ad_0 \\ bn_x & d + bn_y & bn_z & -bd_0 \\ cn_x & cn_y & d + cn_z & -cd_0 \\ n_x & n_y & n_z & -d_1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{3.9}$$

where $d_0 = n_x \cdot x_0 + n_y \cdot y_0 + n_z \cdot z_0$ and $d_1 = n_x \cdot a + n_y \cdot b + n_z \cdot c$.

The major features of perspective projections are 1) perspective foreshortening, 2) vanishing points, 3) view confusion and 4) topological distortion. As indicated above perspective foreshortening refers to the fact that objects more distant from the center of projection are projected as smaller than nearby objects thus providing depth cues. This may be a severe disadvantage when doing quantitative data analysis. Vanishing points are the apparent intersection points of parallel lines not parallel to the view plane. Notice the connection to earlier discussions of projective geometry in which parallel lines meet at ideal points at infinity. Indeed a perspective projection is also known as a *perspectivity* in projective geometry and is the subject of a rather elegant mathematical calculus within the theory of projective geometry. View confusion refers to the fact that the center of projection may actually lie between the view plane and the point, p; that is, in our original simple setup with $C = (0,\ 0,\ -d,\ 1)$, the point $p = (x,\ y,\ z,\ 1)$ may have $z < -d$. In this case, objects containing entirely such points are projected upside down and backward onto the view plane. To understand topological distortion consider the plane which is parallel to the view plane and passes through the center of projection. All points on this plane are mapped to infinity (actually ideal points in the view plane because $z + d = 0$ in this case) with the exception of the center of projection itself which has no projection point. Thus a line segment passing through this plane (that is from behind the center of projection to in front of the center of projection) will be disconnected in the projection. A perspective projection is not continuous and if not carefully chosen may lead to anomalous results.

3.3.2 Parallel Projections:

A parallel projection is, in effect, a perspective projection with the center of projection at infinity. In this case all projectors are parallel. In the simplest setup where the center of projection is at $C = (0, 0, 1, 0)$ and we desire to project into the x-y plane as before, then it is clear that

$$x' = x, \quad y' = y \quad \text{and } z' = 0.$$

This may be represented in homogeneous coordinate matrix form as

$$p' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \text{Par}_{\mathbf{e}_z} \times p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.10}$$

This equation is the obvious and simple case that is very useful for computer graphics. Basically we rotate the coordinate system and discard all but the x and y components. Because the projectors are parallel to the z-axis, hence, perpendicular to the x-y view plane, we refer to this type of parallel projection as an *orthographic projection*. This is not the most general setting, however.

To consider the general case, we first consider a view plane described by surface normal $\mathbf{n} = n_x\mathbf{e}_x + n_y\mathbf{e}_y + n_z\mathbf{e}_z$ and reference point $r_0 = (x_0,\ y_0,\ z_0,\ 1)$. In a parallel projection, we no longer have a meaningful center of projection, so we replace this by a vector **v** representing the direction of projection. We'll denote $\mathbf{v} = a \cdot \mathbf{e}_x + b \cdot \mathbf{e}_y + c \cdot \mathbf{e}_z$. Basically the steps involved are to translate the reference point, r_0, to the origin, align the surface normal **n** to $\mathbf{e}_z$, perform the projection in standard form, and then carry our the inverse operations to restore the view plane to its original location and orientation. These steps are accomplished as follows.

$$p' = Par_{v,n,r_0} \times p = T_{-r_0}^{-1} \times A_n^{-1} \times Par_v \times A_n \times T_{-r_0} \times p, \tag{3.11}$$

where

$$T_{-r_0} = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 0 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.12}$$

and

$$A_n = \begin{bmatrix} \frac{\lambda}{|\mathbf{n}|} & \frac{-n_x n_y}{\lambda|\mathbf{n}|} & \frac{-n_x n_z}{\lambda|\mathbf{n}|} & 0 \\ 0 & \frac{n_z}{\lambda} & \frac{-n_y}{\lambda} & 0 \\ \frac{n_x}{|\mathbf{n}|} & \frac{n_y}{\lambda|\mathbf{n}|} & \frac{n_z}{\lambda|\mathbf{n}|} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.13}$$

where $\lambda = \sqrt{n_y^2 + n_z^2}$ and

$$Par_v = \begin{bmatrix} 1 & 0 & -a/c & 0 \\ 0 & 1 & -b/c & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.14}$$

Of course, if $\mathbf{v}$ is aligned with $\mathbf{e}_z$, the $a = b = 0$ and the last matrix reduces to the standard parallel projection matrix given above. We have found that these equations are sufficient for the basic computer graphics needed in statistical graphics and visualization. Of course, much more complicated projections and visualization problems have been studied and are discussed in the computer graphics literature. We refer to Foley et al. (1990) for a more detailed treatment of special situations.

4. Geometric Forms and Models, Hidden Surfaces

Modeling of objects is a major application of computer graphics. For purposes of statistical data analysis, we are rarely interested in modeling realistic objects, but in our work, we have often been interested in treating surfaces such as regression and density surfaces and contours as if they are real objects. The use of lighting and shading has the advantage of showing fine structure on surfaces and contours. This allows us to judge the degree of smoothing and look for outlier structures. Moreover, it is sometimes appropriate with spatial statistics to model geographic or other forms and shapes as the basis for reporting statistical measurements. There are three basic forms commonly treated: 1) wireframe models, 2) smooth nonparametric shapes and 3) regular parametric geometric solids and surfaces.

4.1 Wireframe Models:

Points and lines are fundamental to all graphics modeling. We specify a point by its coordinates. As we have done before, we continue to use the natural homogeneous coordinate representation for computational purposes. However, for purposes of discussion here, this will be unnecessary and we shall simply use the traditional Cartesian coordinate system. A *line segment* is specified by giving its endpoints, say $p_1 = (x_1, y_1, z_1)$ and $p_2 = (x_2, y_2, z_2)$. A *polyline* is a chain of connected line segments and is specified by giving the vertices or *nodes*, say, $p_0, p_1, \ldots, p_n$ defining the lines segments. The first vertex, in this case p_0, is called the *starting node* and the last, p_n, is the *terminal node*. A *polygon* is a closed polyline in which the starting and terminal nodes coincide. The line segments, $p_0p_1, p_1p_2, \ldots, p_np_0$, are called the *edges* of the polygon. A *planar polygon* is one in which all the edges line in a single two-dimensional plane.

A *wireframe model* is a collection of vertices, edges and polygons. The vertices are connected by edges (in some sequence) and polygons share common edges. The edges are typically straight line segments in which case the wireframe model is called a *polygonal net* or *polygonal mesh*. Typically, the polygons are rectangles or triangles or a combination of both. Often the vertices of a polygonal net are used as the points at which the numerical approximation to the solution of a partial differential equation are computed. Thus wireframe models are frequently used as the basis for numerical modeling of properties a shape. For example, the Navier-Stokes equations describe the fluid dynamic properties so that a polygonal mesh model of a ship hull might be particularly appropriate in determining the flow and pressure properties associated with fluid flow around that hull shape. Wireframe models of density, regression and other mathematical shapes are often used because they illustrate three-dimensional shapes very well in a monochrome environment, for example, as with laser

printer output. Wireframes are particularly useful when the polygons are planar. In this case, the surface normal is easy to compute since the tangent plane is the plane of the polygon. The surface normal is needed when computing lighting and rendering effects. An example of a wireframe model of a bivariate probability density is given in Figure 9.9 later in this work.

While we shall not explicitly deal with data structures, wireframe models are principally stored in two ways. The first method, called *polygon listing*, consists of a list a vertices, each vertex being listed exactly once with each polygon defined by pointers to the vertex list. This data structure is more economical in storage space, but results in some edges being drawn multiple times. The second method, known as *explicit edge listing*, consists of a list of vertices and a list of edges again with each vertex and each edge being listed exactly once. Each edge in the edge list points to its two vertices. A polygon is represented as a list of pointers to the vertex list and to the edge list. This method takes up more memory space, but is faster in execution since each edge is drawn only once.

4.2 Smooth Nonparametric Shapes

It is obvious that if we have a highly curved surface to deal with, the wireframe model will require many edges and many vertices to smoothly model the surface. In many situations, we are interested in replacing straight line segments with smooth curves, and rectangular polygons with curved surface patches. This is generally the domain of differential geometry with which we shall not deal here. However, we are interested in *interpolating* or *approximating* curved lines and surfaces. We can do this with piecewise polynomials with suitable continuity conditions, that is, by splines. Polynomial splines and, particularly, cubic splines are widely used in computer graphics as well as in statistics. Of course, we highly recommend Wegman and Wright (1983) for a general discussion of spline methods in statistics.

Let us consider now $N + 1$ points, p_0, p_1, ..., p_N. We wish to find a curve which either interpolates of approximates the shape outlined by these points. We model the curve, say f(x), by the weighted sum of *basis* or *blending* functions

$$\mathrm{f(x)} = \sum_{i=0}^{m} \mathrm{a}_i \phi_i(\mathrm{x}). \tag{4.1}$$

A *continuous piecewise polynomial* (spline), g(x), of degree n is a set of polynomials, g_i(x), each of degree n and k + 1 *knots* $\mathrm{t}_0, \mathrm{t}_1, \ldots, \mathrm{t}_k$ so that

$$\mathrm{g(x)} = \mathrm{g}_i(\mathrm{x}) \text{ for } \mathrm{t}_i \leq \mathrm{x} \leq \mathrm{t}_{i+1} \text{ and } i = 0, \ldots, \mathrm{k} - 1. \tag{4.2}$$

Typically we require $\mathrm{g}_{i-1}(\mathrm{t}_i) = \mathrm{g}_i(\mathrm{t}_i)$ and depending on degree, matches of derivatives of higher order as well. Polynomials of high degree are not very useful for computer graphics purposes because they may have substantial variation. Cubic splines (n = 3) are most often used.

4.2.1 Polynomial Basis Functions: There are several standard sets of polynomial basis functions.

Lagrange Polynomials of Degree n: For a given set of roots, $\mathrm{x}_1, \ldots, \mathrm{x}_n$

(4.3) $$g_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}, \ i = 0, 1, \ldots, n.$$

B-splines:

For knot set $t_0, t_1, t_2, \ldots$, the n^{th} degree *B-spline* are defined recursively by

(4.4) $$B_{i,0}(x) = 1 \text{ for } t_i \leq x \leq t_{i+1} \text{ and } B_{i,0}(x) = 0 \text{ otherwise,}$$

(4.5) $$B_{i,n}(x) = \frac{x - t_i}{t_{i+n} - t_i} B_{i,n-1}(x) + \frac{t_{i+n+1} - x}{t_{i+n+1} - t_{i+1}} B_{i+1,n-1}(x).$$

Here we take the knots to be distinct. If knots are not distinct and we arrive at indeterminant forms of the sort $\frac{0}{0}$, we take $\frac{0}{0} = 0$.

Bernstein Polynomials:

(4.6) $$\phi_{k,n}(x) = \frac{n!}{k!(n-k)!} x^k (1 - x)^{n-k}, \ 0 \leq x \leq 1.$$

4.2.2 Interpolation:

Interpolation to fit a smooth curve through points $p_i = (x_i, y_i)$ is a fundamental problem for computer graphics. The basis sets given above are the commonly used tools for smooth curve fitting.

Lagrange Polynomial Interpolation:

(4.7) $$f(x) = \sum_{i=0}^{n} y_i g_i(x).$$

Here $g_i(x)$ are the Lagrange polynomials and $f(x)$ is the *n^{th} degree Lagrange interpolator*.

Spline Interpolation:

The m^{th} degree interpolating spline is written in terms of the B-spline basis functions as

(4.8) $$s_m(x) = \sum_{i=0}^{m+n-1} a_i B_{i,m}(x).$$

We take the knots to be $t_0 = t_1 = \cdots = t_m < x_0$, $t_{n+1} = \cdots = t_{m+n+1} > x_n$ and

$$t_{i+m+1} = \frac{x_{i+1} + \cdots + x_{i+m}}{m}, \ i = 0, \ldots, n - m - 1.$$

For m = 3, the interpolating conditions

$$y_j = s_3(x_j) = \sum_{i=0}^{m+2} a_i B_{i,3}(x_j) \tag{4.9}$$

yield n equations in n + 2 unknowns. The remaining two equations are typically derived from boundary conditions.

1) Natural Spline Condition:

$$s_3''(x_0) = s_3''(x_n) = 0.$$

2) Clamped Spline Condition:

$$s_3'(x_0) = y_0', \ s_3'(x_n) = y_n'.$$

Here y_0' and y_n' are some desired endpoint derivative constraints.

3) Periodic Spline Condition:

$$s_3'(x_0) = s_3'(x_n), \ s_3''(x_0) = s_3''(x_n).$$

A wide variety of boundary conditions could be constructed depending on the appropriate circumstances.

4.2.3 Approximation:

Smooth approximation to a set of points is, of course, a stock-in-trade of statisticians doing nonparametric function estimation. Somewhat different methods are employed in computer graphics since the goal is not to find a smooth curve approximating the points, but a smooth curve approximating the polyline. The points, p_0, p_1, …, p_n, $p_i = (x_i, y_i, z_i)$, are called the *control points*. The *Bézier-Bernstein approximation* is the path defined by $p(t) = \Big(x(t), y(t), z(t)\Big)$ with

$$x(t) = \sum_{i=0}^{n} x_i \phi_{i,n}(t), \ y(t) = \sum_{i=0}^{n} y_i \phi_{i,n}(t), \text{ and } z(t) = \sum_{i=0}^{n} z_i \phi_{i,n}(t). \tag{4.10}$$

p(t) is called the *Bézier curve.* Note that $p_0 = p(0)$ and $p_n = p(1)$. The Bézier curve is tangent to the the polyline at p_0 and p_n. The Bézier curve lies entirely within the convex hull of the control points. The Bernstein polynomials can be replaced by the spline B-spline basis functions to obtain the *Bézier–B-spline approximation.* We will not pursue the details of this approximation here.

4.2.3 Surfaces:

A standard construction of a surface in computer graphics is the *Bézier-Bernstein surface* which is computed as follows p(s, t) $=$ $\Big($x(s, t), y(s, t), z(s, t)$\Big)$ with

$$\text{x(s, t)} = \sum_{i=0}^{m}\sum_{j=0}^{n} \text{x}_{ij}\phi_{i,m}(\text{s})\phi_{j,n}(\text{t})$$

and

$$\text{y(s, t)} = \sum_{i=0}^{m}\sum_{j=0}^{n} \text{y}_{ij}\phi_{i,m}(\text{s})\phi_{j,n}(\text{t})$$

and finally

$$\text{z(s, t)} = \sum_{i=0}^{m}\sum_{j=0}^{n} \text{z}_{ij}\phi_{i,m}(\text{s})\phi_{j,n}(\text{t})$$

where $\text{p}_{ij} = (\text{x}_{ij}, \text{y}_{ij}, \text{z}_{ij})$, $i = 0, \ldots, \text{m}$ and $j = 0, \ldots, \text{n}$ form the *guiding net*. The variables s and t satisfy $0 \leq \text{s}, \text{t} \leq 1$. Moreover, $\text{p}_{00} = \text{p}(0, 0)$, $\text{p}_{0n} = \text{p}(0, 1)$, $\text{p}_{m0} = \text{p}(1, 0)$ and $\text{p}_{mn} = \text{p}(1, 1)$. The analogous derivative properties hold at the corners and the convex hull property likewise holds. The Bernstein polynomials may be replaced by B-spline basis functions to obtain the *Bézier–B-spline surface*.

Notice that all of the curve and surface fits we have discussed are linear combinations of basis functions. All of the transformations we have discussed such as rotation, translation, scaling, alignment and so on have been cast in the form of homogeneous coordinate linear transformations. Thus, for example, if $\text{x} = \sum \text{x}_i\phi_i$, $\text{y} = \sum \text{y}_i\phi_i$, and $\text{z} = \sum \text{z}_i\phi_i$, and a 4×4 matrix $\mathbf{V} = [\text{v}_{ij}]$, then

$$\text{p}' = \text{V} \times \text{p} = \begin{bmatrix} \sum_i(\text{v}_{11}\text{x}_i + \text{v}_{12}\text{y}_i + \text{v}_{13}\text{z}_i)\phi_i \\ \sum_i(\text{v}_{21}\text{x}_i + \text{v}_{22}\text{y}_i + \text{v}_{23}\text{z}_i)\phi_i \\ \sum_i(\text{v}_{31}\text{x}_i + \text{v}_{32}\text{y}_i + \text{v}_{33}\text{z}_i)\phi_i \\ 1 \end{bmatrix} \tag{4.11}$$

Thus, transformations of geometric forms which are described as a linear combination of basis functions are easily computed by our standard homogeneous coordinate methods.

4.3 Regular Parametric Geometric Solids and Surfaces

Curve and surface fitting by smooth curves and wireframe models are frequent techniques for making two- and three-dimensional structures of general shape. These techniques are particularly useful in the statistical graphics and visualization setting since the densities and regression curves and other statistical structures are unlikely to have regular

geometric shapes. Much of computer graphics is, however, used in computer aided design (CAD) and computer aided manufacturing (CAM). For manufacturing purposes, it is highly desirable to have regular geometric shapes with a parametric character, that is, cubes, parallelepipeds, cylinders and so on since these are the shapes which can be manufactured easily. Hence in much of computer graphics, objects are built up from relatively simple primitives. We list just a few for completeness:

Sphere: $x^2 + y^2 + z^2 = r^2$

Ellipsoid: $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$

One-sheeted Hyperboloid:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$$

Two-sheeted Hyperboloid:

$$\frac{z^2}{c^2} - \frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

Elliptic Cylinder:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

If $a = b = r$, then the cylinder is a circular cylinder of radius r.

Elliptic Paraboloid:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = cz$$

Elliptic Cone:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = z^2$$

Hyperbolic Paraboloid:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = cz.$$

4.4 Hidden Surfaces

In drawing complex three-dimensional geometric shapes, there is a potential problem in computing which pieces of an image is obscured. We mention this in passing because it can be a problem for certain statistical graphics applications. A simple approach when for example doing rotating scatterplots is to ignore depth considerations. In this application, unless the number of points is extremely massive, the likelihood of two points being superimposed is relatively small and in a dynamic graphic one will obscure the other only

momentarily. Indeed, if the presentation is in monochrome, it will not matter anyway. However, if points are colored for example with brushing or for red/green stereoscopic displays (more on these later), the overlap may matter. The pixel color that is drawn is the color of the last item to be listed which is not necessarily the one with the largest z-value. Indeed, the problem is more acute if one is trying to do realistic rendering, complicated wireframe models or rotating stereoscopic displays.

There are several solutions available, some of which are accomplished in hardware. A standard solution is to have a so-called *z-buffer*. Normally with each pixel location, as we have indicated earlier, there are a number of bits of memory, ranging up to 24 or 32 bits. The more bits, the more distinct colors can be rendered. However, this is ultimately limited by the perceptual capabilities of the human observer and the phosphor technology. Adding more bits does not necessarily increase the number of distinct colors that can be observed since the number of just distinguishable color differences is limited by the combination of the human visual system and the response of the phosphors on the cathode ray tube to increased electron intensity. For this reason 32 bits seems to be a useful upper limit. However, more bits can be added for depth so that if two objects in the image contribute distinct colors to a given pixel, then the one with the smallest z-value is drawn. This hardware solution is contained in most high-end graphics-oriented workstations.

A second hardware oriented solution is to draw the image starting with the largest z values and finishing with the smallest z. This way, items with large z values are overwritten later by items with smaller z-values. This method is simple but not particularly efficient in implementation and is useless for dynamic graphics. There are a large number of algorithms with evocative names such as area-subdivision algorithm, depth-sort algorithm, list-priority algorithm as well as the scan line algorithm and the z-buffer algorithm just mention. One algorithm of interest is the ray tracing algorithm which we shall treat separately below.

5. Realistic Rendering and Transparency

Traditional statistical graphics has historically consisted of line drawings and graphics and it would seem at first glance that there is little reason to consider so-called photo-realistic techniques in the analysis of statistical data. Scientific visualization on the other hand has emphasized imaging (and imagining) the illustration of scientific concepts (touring a black hole, visiting a cluster of atoms, watching the flow inside of a storm cloud) that would be impossible for humans to see under normal experimental circumstances. Statistical graphics has emphasized quantitative behavior while scientific visualization has normally emphasized more qualitative aspects of behavior. For this reason there has been more emphasis on realistic rendering in what has begun to be called scientific visualization.

In our own work, however, we have found that often we are interested in the qualitative behavior of densities and regression surfaces, particularly in non-trivial, non-linear and typically higher dimensional settings when the density or regression structure is convoluted and complicated. In these circumstances we have found it to great advantage to introduce some of the realistic rendering concepts including transparency. We have, for example, given density surfaces or density contours metallic characteristics. When combined with lighting and specular reflection models, we are able to visualize fine structure on the surface of the density that we would simply be unable to see with ordinary wire frame models. With the

addition of transparency (and stereoscopic displays) we are able to see the behavior of extremely complicated density structures. Of course, these images also have a certain level of visual appeal whose value should not be discounted. In this section we should like to cover basic lighting and shading models, continue with a brief discussion of transparency and complete the section with a short discussion of ray tracing.

5.1 Lighting and Shading:

In most real visual environments, there is a considerable amount of ambient light, light with equal intensity from all directions. Ambient light is the easiest type of light to model because it is assumed to produce equal amount of light on all surfaces independent of an object's location and orientation. Using ambient light alone produces flat, unrealistic objects. For this reason we often wish to add one or more *point sources* of illumination which may either be located locally to the object or located at infinity. In the latter case the light rays all come from the same direction and the source is known as a *directional source*. Modeling these sources is more difficult since their effect depends on the surface's orientation. If the surface is normal, that is perpendicular, to the light source, the surface is brightly illuminated whereas if the incident light rays are oblique to the surface, the object will be less brightly illuminated.

Realism is enhanced if the surface properties of the modeled material are taken into account when shading is determined. Some materials are dull and disperse light approximately equally in all directions whereas others are shiny and reflect light in a very limited number of directions. Our discussion so far has been on the lighting of opaque surfaces only. Transparent or semi-transparent surfaces can be quite useful as well. It is not always necessary to account fully for refraction effects, since we often use transparency for revealing hidden structure and not for completely photorealistic imaging. In the next several sections, we use the standardized notation of Foley et al. (1990).

5.1.1 Ambient Light:

As indicated above, the simplest form of a lighting model is ambient light. We express a lighting model in terms of an *illumination equation*. The simplest such model is $I = k_i$ where I is the calculated intensity and the coefficient k_i is the i^{th} object's intrinsic intensity. This is a model for self-illuminated objects such as incandescent bulbs or stars. If instead of self luminosity, we consider a non-directional, diffuse source of ambient light as would result from general reflection of light from all objects in the environment, we may model the illumination equation as

$$I = k_a I_a \tag{5.1}$$

where I_a is the intensity of ambient light which is assumed to be constant for all objects and k_a is the *ambient-reflection coefficient.* The ambient reflection coefficient is the fraction of ambient light reflected by a surface under consideration and ranges between 0 and 1. Of course, different surfaces in the image may have different ambient-reflection coefficients. The ambient reflection coefficient is a property of the material under consideration and not of the light sources.

5.1.2 Diffuse Reflection:

Now let us consider an object lit by a point source. Objects illuminated by ambient light are uniformly lit in proportion to the intensity of the ambient light. For objects lit by a point source, the illumination will vary from one part of the object to another with obviously no illumination on the backside of the object relative to the point source. See Figure 5.1. Dull or matte surfaces exhibit *diffuse reflection* or *Lambertian reflection.* These surfaces appear equally bright from all viewing angles because they reflect light equally in all directions. For a given surface the brightness depends only on the angle θ between the surface normal $\vec{N}$ and the direction to the point light source $\vec{L}$. To see this consider an infinitesimal area ∂A of the point source light beam. For a given surface with angle θ, the light in this beam will intersect an area on the surface which is ∂A/cos(θ). See Figure 5.2. Thus the amount a surface area intercepted by a fixed-width beam will grow as θ approaches 90° in magnitude so that the amount of energy reflected by a fixed patch on the surface will decrease to 0 as θ approaches 90°. Thus for Lambertian surfaces, the amount of reflected light seen by the viewer depends on the viewpoint and is proportional to cos(θ). The diffuse illumination equation becomes

$$I = k_d I_p \cos(\theta) = k_d I_p(\vec{N} \cdot \vec{L}). \tag{5.2}$$

Here k_d is the material's diffuse reflection coefficient. As with k_a, k_d ranges between 0 and 1 and is the fraction of light reflected. In many circumstances $k_d = k_a$. I_p is the illumination intensity of the point light source. $\vec{N}$ and $\vec{L}$ are the unit vectors respectively for the surface normal and the direction of light source. Of course cos(θ) is the dot product of these two unit vectors. Also, (5.1) and (5.2) can be combined into the following illumination equation

$$I = k_a I_a + f_{att} k_d I_p(\vec{N} \cdot \vec{L}) \tag{5.3}$$

where f_{att} is the attenuation factor for light, $f_{att} = d_L^{-2}$. That is, light intensity from a point source falls off as the inverse square of the distance, d_L, between the light source and the object. To avoid the singularity when $d_L = 0$, a useful approximation can be made by letting

$$f_{att} = \max\left(\frac{1}{c_1 + c_2 d_L + c_3 d_L^2}, 1\right).$$

Equation (5.3) can be generalized in the obvious way to account for variable surface reflectivity for different colors by introducing an additional multiplicative attenuation factor for each color, RGB, or, more generally, for each frequency, λ, of light. Thus the equation becomes

$$I_\lambda = k_a I_{a\lambda} O_{d\lambda} + f_{att} k_d I_{p\lambda} O_{d\lambda} (\vec{N} \cdot \vec{L}). \tag{5.4}$$

Here $O_{d\lambda}$ is the diffuse attenuation associated with a color whose frequency is λ.

5.1.3 Specular Reflection and the Phong Illumination Model:

Specular reflection occurs with objects with a shiny surface such as a mirror or a metallic surface. If one were to illuminate a metallic sphere with a point light source, one would have a highlight from the specular reflection as well as more general illumination of the sphere arising from diffuse reflection. Light specularly reflected from a colorless surface has the same color as the light source. Motion of the viewpoint also causes the highlight to move on the surface of the object. For a perfect specular reflection, the angle of incidence is equal to the angle of reflection. See Figure 5.3. If $\vec{R}$ is the unit vector in the direction of reflection then the angle between $\vec{N}$ and $\vec{R}$ is θ. If $\vec{V}$ is the unit vector to the direction of the viewpoint, the viewer would only see a perfect specular reflection if $\vec{V}$ and $\vec{R}$ coincide. However, for less perfect specular reflection, there is some level of visibility of the highlight which is proportional to the angle between $\vec{V}$ and $\vec{R}$. Call this angle α. Maximum specular reflectance obviously should occur when $\alpha = 0°$ and should drop to 0 when $\alpha = 90°$. The Phong lighting model introduced by Bui-Tuong (1975) models this dropoff as proportional to $\cos^n(\alpha)$. For $n = 1$, there is a gentle dropoff and as $n \to \infty$, we approximate perfect specular reflection. The Phong model is then written

$$I_\lambda = k_a I_{a\lambda} O_{d\lambda} + f_{att} I_{p\lambda}[k_d O_{d\lambda} \cos(\theta) + W(\theta) \cos^n(\alpha)] \tag{5.5}$$

where $\cos(\alpha) = \vec{R} \cdot \vec{V}$ and $W(\theta)$ is the material's specular reflection coefficient at angle θ which as before ranges between 0 and 1. $W(\theta)$ is typically set to a constant k_s which is determined experimentally. Using this constant approximation and the vector notation, the illumination equation becomes

$$I_\lambda = k_a I_{a\lambda} O_{d\lambda} + f_{att} I_{p\lambda}[k_d O_{d\lambda}(\vec{N} \cdot \vec{L}) + k_s O_{s\lambda}(\vec{R} \cdot \vec{V})^n]. \tag{5.6}$$

Here we have added one other subtlety, $O_{s\lambda}$, which is the attenuation factor for specular reflection at a color of frequency λ. Since it is known that the angle of reflection and the angle of incidence are the same, $\vec{R}$ can be computed in terms of $\vec{L}$ and $\vec{N}$ using simple geometry by the formula, $\vec{R} = 2\vec{N}(\vec{N} \cdot \vec{L}) - \vec{L}$. Thus $\vec{R} \cdot \vec{V}$ can be computed by $(2\vec{N}(\vec{N} \cdot \vec{L}) - \vec{L}) \cdot \vec{V}$. Thus it is not necessary to compute $\vec{R}$ explicitly.

If there are multiple point light sources, say, m, the illumination equation becomes

$$I_\lambda = k_a I_{a\lambda} O_{d\lambda} + \sum_{1 \le i \le m} f_{att,i} I_{p\lambda i}[k_d O_{d\lambda}(\vec{N} \cdot \vec{L}_i) + k_s O_{s\lambda}(\vec{R}_i \cdot \vec{V})^n] \tag{5.7}$$

where the quantities indexed by *i* have their obvious interpretation.

What is perhaps not readily apparent to the reader is that the computations in formula (5.7) and its predecessors are pointwise computations, i.e. computations for a single point on the surface of the object. Obviously $\vec{N}$ changes as does θ, $\vec{R}$ and α when the surface point changes. We have seen earlier in our discussions that solid structures are often modeled using

planar polygons. Obviously the surface normal, $\vec{N}$, is constant for the entire polygon. If the light source is relatively close to the polygon, it may not be a realistic assumption that the vector pointing to the light source, $\vec{L}$, is fixed for every point in the polygon. However, if the point light source is at infinity, $\vec{L}$ is fixed for every point in the polygon and so the dot product, $\vec{N} \cdot \vec{L}$, is fixed for all points in the polygon. In some ways this simplifies the computation since there are only a finite number of polygons to deal with rather than an infinite number of surface points. Of course, $\vec{R} \cdot \vec{V}$ would not in general be constant for a given polygon since the view vector would vary from point to point. Indeed, in general we would not wish the light intensity to be constant across the surface of the polygon since, unless the polygons were extraordinarily tiny, this would lead to discernibility of individual polygons and a resulting faceting in the image. Indeed, even with very small polygons, the smoothness of the reflected intensity is problematic if adjacent polygons have perceptibly different intensities. This holds because of apparent intensity differences are perceptually magnified by the human visual system. Thus, particularly for polygonal-based three-dimensional displays, it is desirable to have a continuous approximating shading algorithm. The most well known of these is called *Gouraud shading.*

5.1.4 Gouraud Shading and Phong Shading:

Gouraud shading or *intensity interpolated shading* is designed to eliminate intensity discontinuities. The idea is to begin by calculating a lighting model at each vertex of the polygonal mesh model. To do this, of course, we would need to know the surface normal, $\vec{N}_v$, at each vertex v. This may be known by knowing the actual surface normal from an analytical description of the surface or, alternatively, may be approximated by calculating an approximating normal by averaging the surface normals for the polygons contiguous to the vertex in question. Thus, we would use the approximation

$$\vec{N}_v = \frac{\sum_i \vec{N}_i}{|\sum_i \vec{N}_i|}.$$

The second stage is then to find the vertex intensities by using an appropriate lighting model such as (5.7) with the vertex surface normal. The intensities along the edges of the polygon are then linearly interpolated so that the boundaries of all polygons have an intensity assigned. Finally, the intensities along each scan line (Gouraud shading assumes a raster scanning display) between edges are again linearly interpolated between the values of the intensities on intersection of the edges with the scan line. This type of interpolation can be computed with a relatively computationally efficient algorithm.

While Gouraud shading is an effective technique, one can easily see the source of potential difficulties. Suppose, for example, that a specular highlight should have occurred somewhere in the middle of a polygonal region, i.e., that $\vec{R} \doteq \vec{V}$. Because Gouraud shading is basically a linear intensity interpolation, if none of the vertices picks up the specular reflection, it will not be reflected on the interior of the polygon either. *Phong shading* is based on the same sort of interpolating idea, but in place of interpolating intensities, one linearly interpolates surface normals, first along the edges of the polygon and then along scan lines

from edge to edge. At each pixel the appropriate lighting model is applied based on the interpolated normal at that pixel. Since the lighting model must be computed at each pixel to arrive at the intensity at that pixel, Phong shading is much more computationally intensive. However, the resulting improvement in realism often justifies the added computational effort.

5.2 Transparency:

Transparency can be implemented in either non-refractive or refractive form. The latter involves Snell's law calculations which are necessary for transparency calculations for photorealistic visualization. An extensive discussion of refractive transparency may be found in Foley et al. (1990). However, refractive transparency has relatively limited application in the statistical context and so we shall concentrate only on non-refractive transparency. This form of transparency is primarily used for showing internal structure. Two distinct methods are used in creating non-refractive transparency: 1) *interpolated transparency* and 2) *filtered transparency.*

Consider two polygons, say polygon 1 and polygon 2, one positioned in front of two. Interpolated transparency determines the shade of a pixel on the intersection of the two polygon's projection through linear interpolation,

$$I_\lambda = (1 - k_{t1})\, I_{\lambda 1} + k_{t1}\, I_{\lambda 2} \tag{5.8}$$

where k_{t1} is the *transmission coefficient* and measures the transparency of polygon 1. Here, of course, the t index reminds that this coefficient is associated with the transparency factor. Obviously if $k_{t1} = 0$, polygon 1 is opaque and transmits no light from polygon 2 while if $k_{t1} = 1$, polygon 1 is completely transparent. This form of transparency is called interpolated transparency. It is often useful to separate the diffuse or Lambertian reflection component and the specular reflection component and to apply the interpolated transparency to the diffuse component but leave the specular reflection component of polygon 1 untouched. This is obviously not completely satisfactory since a specular reflection on polygon 2 would have to be handled in some exceptional way.

A second method is to treat polygon 1 as a transparent filter that selectively passes different wavelengths, i.e. a colored filter. The resulting illumination equation is given by

$$I_\lambda = I_{\lambda 1} + k_{t1} O_{t\lambda} I_{\lambda 2} \tag{5.9}$$

where $O_{t\lambda}$ is the *transparency color* of polygon 1. In either transparency model, if there are multiple layers, the model is applied recursively from the back towards the front.

Of course, computer graphic images are not dealt with frequency by frequency, but as a composition of red, green and blue. Thus each pixel will be represented by a three vector, (R, G, B). Transparency as well as other forms of image composition are often dealt with in real hardware by the so-called α*-channel*. In essence a fourth component is added to the RGB vector, (R, G, B, α) where α represents a blending parameter. Thus equation (5.8) could be handled by the hardware composition of two pixel values (R_1, G_1, B_1, $1 - k_{t1}$) and (R_2, G_2, B_2, k_{t1}). Many graphics oriented computers have the α-channel implemented in hardware so that transparency is easy to implement and rapid to execute.

5.3 Ray Tracing:

Phong lighting, Gouraud and Phong shading and even Snell's law transparency calculations are image-based approximating models to reality. In a sense the viewer is an uninvolved observer of the scene. The effect of mirrors, multiple refractive transparent objects and multiple specular reflections is almost impossible to capture with such models. An alternative approach to computing photorealistic images is the so-called *ray tracing* method.

The fundamental problem for any graphics system is to decide on the color, that the viewer's eye will perceive, associated with each pixel in the viewport. In our previous discussions, we have tried to do this by calculating the color and related visual properties of the object based on an illumination model more or less independently of the interposition of the viewport and the viewer's eye. One significant problem with this perspective is that reflected images from mirrors, glass windows, and other specular surfaces for example could not be accommodated well by previous models.

One could imagine a method of modeling in which a photon of a given color emerges from a light source in a given direction. One could at least in principle trace the path of that photon through the scene one had described, accounting for passage through transparent objects according to Snell's law and accounting for a certain level diminution of intensity, accounting for specular reflection off of mirrored surfaces, accounting for diffuse reflection off of Lambertian surfaces, and, of course, accounting for multiple optical interactions according to standard laws of optics, following the path to discover whether or not the photon actually passed through any pixels of the viewport to the eye of the viewer. If such a photon did pass through the viewport to the eye of the viewer, then its resulting color after accounting for all of its optical interactions would be a contributor to the final color associated with that pixel. One would then have to integrate over the distribution of colors emanating from the given light source and over all directions in which that light source could radiate to account for the contribution of that light source to the scene as seen through the viewport by the viewer's eye. Obviously this would have to be done for each light source. Once the color-intensity properties are known for each photon passing through a given pixel, a resultant average color for the pixel could be calculated. The process just described is called *forward ray tracing*. Forward ray tracing is a computationally inefficient method since most of the photons would never pass through the viewport to the viewer's eye.

It is easy to imagine the more efficient method, diagrammed in Figure 5.4, of *backward ray tracing* or simply *ray tracing* in which one begins by following a hypothetical light ray emanating from the eye through a given pixel (assuming the pixel is infinitesimal). The ray may not intercept any objects in which case the pixel would be black. The ray may intercept a transparent, a specular or a diffuse object whose surface properties are defined by the model. Based on the physical properties of the object, one may trace backwards the path of light rays which contribute to the ray from the object to the eye through the viewport. This is obviously a recursive process that is normally quite computationally intensive, but still far more efficient that a forward ray tracing would be. Because the ray traced path is based on known physical properties of light and the physical surface properties modeled for the objects, the resulting images can be extraordinarily photorealistic. Ray tracing techniques are a relatively specialized methodology which, to this point, have not been extensively used, or perhaps not used at all, in the statistical graphics and visualization framework. Because of the

computationally intensive nature of the process, however, statistical sampling methods have proven to be useful in ray tracing. These methods are known as *stochastic ray tracing* and *statistical supersampling.* Foley et al. (1990) discuss ray tracing briefly. A more extensive reference is Glassner (1989).

6. Stereoscopic Displays

Stereoscopic displays help the analyst escape from the world of 2-D visualization into the natural domain of 3-D visualization. In the following discussion the typical goal will be the production of a 3-D scatterplot. The goal has strong implications in terms of selecting a stereo projection. In the ordinary world, familiarity with objects and many depth cues facilitate the fusion of left and right retinal images into a stereo image. Monocular depth cues include *linear perspective* (objects of equal size transect areas inversely proportional to their distance), *interposition* or *occlusion* (when one object is in front of another it obscures the more distant object), *shadows* (we generally assume light comes from above), *detail perspective* (no fine detail appears in distant objects due to limited visual acuity), and *aerial perspective* (greater optical depth through the air leads to a blue shift). For most elementary 3-D scatterplots prior knowledge about the form to be perceived is limited and monocular cues are restricted, so care must be taken in the selection of a stereo projection.

6.1 An Infinite Family of Reasonable Stereo Projections:

Most people are able to fuse separate left and right eye images into a stereo image. In fact the visual processing system is so powerful that fusion is possible under a variety of conditions that might not be anticipated. Julesz (1971) has studied the limits of stereo fusion. He found, for example, that image fusion is still possible when one eye image is expanded by as much as 15% or rotated by as much as 6°. Such perceptual flexibility leads to several classes of viable stereo projections. For statistical visualization purposes, classes of stereo projections that included unnecessary distortions, such as enlarging one image, are of little interest. However, this still leaves an infinite class of stereo projections.

Different geometric models lead to different projections. (Geometric models are idealized in that they ignore basic facts about the eye such as the blind spot, the region of high resolution, and various ocular imperfections.) A simple model (Newman and Sproull, 1973) presumes that the eyes converge on a single focal point. The model constructs left and right images by projecting onto left and right projection planes that contain the focal point, and are orthogonal to the respective lines of sight (projectors). This *fixed-focal-point model* is appropriate for a advanced dynamic systems that update immediately as the eyes change their focal point. However, the fixed-focal-point projection does not correspond to typical stereo viewing scenarios. Valyus (1962) states, “it has been shown experimentally that eye movements performed when stereoscopic picture are viewed are similar to those performed in observing a real object. As the gaze is transferred from one object to another the eyes perform conjugate movements directed to the subjectively most important regions, and at the same time coordinated convergence movements take place." While the fixed-focal-point projection has proved passable for showing images of familiar scenes, this projection is inadequate for 3-D scatterplots unless the projected image is updated as the eyes move (Carr and Littlefield 1983).

When the eyes have multiple focal points within a fixed stereo image, a reasonable compromise uses a single common projection plane that is parallel to a frontal view of the face. In the multiple-focal-point model, a data point projected into the projection plane has the same y coordinate for both left and right images. This is a fundamental requirement for 3-D scatterplot projections.

Two classes of projections satisfy the *multiple-focal-point* (single projective plane) constraint. The first uses standard projective methods with separate centers for projection for the left and right eyes, denoted LCOP and RCOP respectively. See Figures 6.1a and 6.1b. The LCOP and RCOP coordinates relative to the center of the workstation screen are $(\frac{e}{2}, 0, -d, 1)$ and $(-\frac{e}{2}, 0, -d, 1)$ where e is the eye separation and, as before, d is the distance from the viewscreen. The coordinates of a display-scaled data-point (x, y, z, 1) projected onto the screen are

$$x_l = \frac{xd-\frac{ez}{2}}{d+z}$$

$$x_r = \frac{xd+\frac{ez}{2}}{d+z} \tag{6.1}$$

$$y' = \frac{yd}{d+z}$$

The second class of stereo projection may be called a *depth-cued orthogonal projection*. In this projection, the LCOP and RCOP are shifted for each data point so that the midpoint between the eyes has the same x and y coordinates as the display point to be projected. The depth-cued points are then

$$x_l = x - \frac{ez}{2d}$$

$$x_r = x + \frac{ez}{2d} \tag{6.2}$$

with y invariant.

The fact the both left and right images have the same y coordinate is evident from geometric considerations. The two eyes are centers of projection for projecting a data point onto the projection plane and the three points define a plane that intersects projection plane. The projection plane is parallel to a frontal view of the face. Since eyes are level, the twin projected images must also be level.

The two projection classes have projection parameters and viewing parameters that can vary. Projection parameters include eye separation, e, projection distance, d, and implicit scaling that controls the extrema for z, z_{min} and z_{max}. Viewing conditions may differ from the projection model. For example, a stereo slide might be projected from different distances yielding a magnification parameter, m, and viewed from different distances yielding a view distance parameter, d'. These two parameters can be varied over an interval and still lead to

comfortable image fusion of a 3-D point cloud. Consequently both stereo projection classes are infinite.

6.2 Projection Parameter Bounds and Parallax:

While an infinite number of projections are effective, some parameters bounds should not be violated and some projections are more desirable than others. The following provides some background concerning parameter constraints and the curious reader is referred to Valyus (1962) for additional detail.

For most people, eye separation falls in the interval between 52 and 74 millimeters (Valyus, 1962). Fortunately, using the exact eye separation for each individual is not crucial as evidenced by stereo publications that are enjoyed by diverse audiences. Specific examples that follow use $e = 60$ mm for convenience. Parallax is a key concept for understanding the projection parameter bounds. The *horizontal parallax*, p, of a point refers to the distance between the projected coordinates on the screen, x_l and x_r. Then for the perspective stereo projection

$$p = x_r - x_l = \frac{ez}{d+z} \tag{6.3}$$

A point in front the screen will have a negative z and the parallax, p, will be negative. Similarly, a point behind the screen will have positive parallax. Parallax must be limited to provide acceptable stereo fusion. Image focus constrains the amount of acceptable parallax. Suppose the eyes converge on a twin images of a point as if the point were real. The convergence and accommodation coupling for the eyes (see Section 7.2) create a region in front and in back of the point that is in focus. If this region includes the projection plane (workstation screen) the image of the point will be clear. While those experienced in stereo viewing often learn to decouple the convergence and accommodation of their eyes, a mismatch can lead to either fusion or focus problems. Constraining the parallax to keep the perceived depth of image close to the screen avoids such problems.

Visual constraints in the literature are typically studied in terms of angular measure on the retinae. As a simple approximation, define the *angular parallax*, p', of a point to be the angle between the left eye image of a point, the midpoint between the eyes, and the right eye image of a point. Figure 6.2 provides a view from above. Thus

$$p = 2d \tan\left(\frac{p'}{2}\right). \tag{6.4}$$

Valyus recommends a limit of $p' = 1.6°$ for positive parallax and Yeh and Silverstein (1990) suggest a bound of 4.93° for negative parallax. The time it takes to fuse left and right images depends on parallax and Yeh and Silverstein suggest .46° and .48° as limits for fusion of 200 millisecond displays.

Consider parallax in an example. Suppose the data has been scaled to fall in a display cube with 20 cm on a side. Suppose the display cube has the front face against the screen and is otherwise centered. Also suppose that the viewing distance is 60 centimeters. Then the

lower left corner of the display cube has coordinates (− 10, − 10, 20, 1) and the angular parallax is

$$p' = 2 \arctan\left(\frac{ez}{2\,d\,(d+z)}\right) = 1.43° \tag{6.5}$$

Making the display cube much beyond 23 cm on a side will exceed the Valyus bound. Compared to the size of a large workstation screen, 23 cm can seem confining. However, Lipton (1982) notes that parallax can be substantially reduced by centering the image depthwise on the display screen. Thus setting $z_{min} = -23$ and $z_{max} = 23$ make the display cube 46 cm on a side and keeps the parallax within bounds. However, the display cube must be much smaller to meet the Yeh and Silverstein rapid fusion criterion.

6.3 Magnification and Viewing Distance:

View related parameters include the viewing distance, d′, and the image magnification, m. These are described in turn. The equation of parallax can be solved for the apparent distance from a display point to the screen, z′, in terms of the actual viewing distance, d′.

$$z' = \frac{d'}{(e/p-1)}. \tag{6.6}$$

If the parallax is fixed then doubling the viewing distance doubles the apparent distance of the point to the screen. Thus if a 3-D plot frame were constructed to be a cube, the cube can be made to appear squashed or elongated by selecting a different viewing distance, d′, than actual projection distance, d.

Often a stereo image is shown to a large audience on a big screen. If the image was not designed for the room, magnification may cause problems. Magnifying the image magnifies the parallax. A Taylor series expansion of the depth equation (6.6) yields

$$z' = d'\,(p/e - (p/e)^2 + (p/e)^3 \cdots) \tag{6.7}$$

When the parallax is small relative to eye separation magnifying the image increases the apparent depth almost linearly. However, as magnification of positive parallax approaches eye separation, the depth of image theoretically approaches infinity. This causes both distortion and fusion problems. (Perceived depth does not reach infinity for complex reasons.) At first consideration it might be thought that increasing the viewing distance will reduce angular parallax and, assuming the image is still bright enough, solve the fusion problem for large parallax images. However, at any viewing distance, parallax beyond eye separation prevents optimal focus of the eyes on the twin image points, so large parallax is undesirable.

Image production problems will be avoided if d, z_{min} and z_{max} are chosen based on room size. The image will be reduced, for example, if the production media is slides, but then will be magnified appropriately by the slide projector. Slides have sufficient resolution that the reduction process does not usually cause a problem. However, pixel rounding problems might be exacerbated by magnifying an image from a low resolution CRT screen.

6.4 Stereoscopic Resolution:

Stereoscopic resolution can be thought of as the number of discriminable depth planes per unit depth. The number of discriminable depth planes per meter is a function of visual acuity and depth of focus and can be approximated by

$$N = \frac{e}{ad^2} \tag{6.8}$$

where visual acuity, a, is expressed in radians, and d is in meters.

Stereoscopic (*binocular*) *acuity*, a, is usually defined as the minimum discriminable parallax. Valyus (1962) indicates that subjects vary considerably with respect to stereoscopic acuity. The population mode for stereoscopic acuity is around 5″ of arc but the mean is considerably closer to 30″ of arc.

As an example, assume that the eyes are focused on the screen at a distance of .6 meters and that the visual acuity is 5″ of arc. The number of discriminable depth planes per meter is 6875. If the depth of the display cube is .2 meters, 1375 depth planes will be discriminable within the display cube. Unfortunately the pixel resolution of typical workstations does not allow this degree of discrimination. The parallax for a depth of .2 meters is 1.5 cm. If the screen has 40 pixels per centimeter then only 60 depth planes can be shown. While depth resolution is much less than horizontal resolution, even the horizontal pixel resolution is not that good. If the projected display cube is .2 meters width, only 800 pixels are available for representing different x values. Higher resolution screens are desirable for increasing resolution in all directions. It might be thought the 300 line laser resolution is adequate for most visual purposes. However, equation (6.8) shows that stereoscopic resolution increases with decreasing in viewing distance. When viewing side-by-side laser-printed stereo plots with a magnifying stereopticon, resolution limitations are evident for points that are supposed to be on flat surface.

Stereoscopy and display resolution limitations influence statistical graphics design. For example Carr and Nicholson (1989) use a 1.5 cm long stereo ray to represent a 4-D point. The ray angle represents the fourth coordinate. Since the parallax difference for the two ray endpoints determines the depth angle of the ray and this parallax difference translates into a few pixels as best, depth angle provides little information. Further, if the ray points toward or away from the analyst, it also becomes impossible to assess the angle in the plane of the screen. Consequently, Carr and Nicholson use ray angle in the plane of the screen to encode the fourth coordinate and always draw the ray parallel to the screen. Hence, the ray angle is always visible (see Section 6.6) for the grand tour and 4-D rotation. Similar to the production of stereo images, obtaining the desired display while exploiting the power of standard geometry engines requires a bit of trickery.

6.5 Eye Separation and Hyperstereoscopy:

Increasing the stereoscopic effect by increasing the effective eye separation is called *hyperstereoscopy*. This provides greater relief (more distinguishable depth planes) for objects at a distance. For example two pictures of a mountain taken while riding a boat may be used to create a stereo image. The distance between the two camera positions might be kilometers rather than the 6 centimeters between the eyes. For familiar objects, human perception scales

everything to normal eye separation. Hyperstereoscopic pictures with increased viewpoint separation appear as if the whole scene, including the viewing distance, were reduced. Thus increasing viewpoint separation decreases the apparent depth of the image even though it ostensibly increases the parallax.

6.6 Control of Perspective Differences:

Under natural conditions the left and right eyes have different views of the world. The field of view for a single human eye is about 150° horizontally and 135° vertically. In binocular vision the field of view covered by both eyes is 120° so that 30° or 1/5 of each eye's field of view is unique to the eye. For 3-D scatterplots all points must be seen unless hidden by occlusion, so images must be restricted to the shared field of view. If the display cube is sufficiently smaller than available display space on the workstation screen this is not a problem. How large the display cube can be, relative to the screen, may depend on how the projection is implemented.

The hardware-based geometry engines on advanced workstations typically have just one center of projection and no direct means of controlling this center. Lipton (1982) and others have noted that simple translations allow the left and right eye images to be computed by making the single center of projection the midpoint between the eyes. Then the left eye image is generated by shift the display points to the right by e/2, projecting them and then shifting the projected points to the left by e/2. Translating points in the opposite directions produces the right eye image. While this off-center projection approach produces the correct coordinates, Hodges (1992) notes that the fields of view are different than those for the standard LCOP and RCOP images. This creates discrepancies for visualization of images that use the whole screen. For viewing 3-D scatterplots, the display cube will have to be constrained so that all projected points fall in the shared field of view.

Perspective differences within the shared field of view can cause fusion and interpretations problems. Perspective induced fusion problems are most likely to arise when showing the display cube frame in small side-by-side plots designed for viewing through a stereopticon. If, for example, the left cube face has an x coordinate $-e/2$, the left face will project as a line in the left eye image, but as a trapezoid in the right eye image. This radical perspective difference in shape complicates image fusion. Since the display cube frame is simple, avoiding radical differences in perspective is straightforward. For example data cube rotation is often sufficient. In fact some analysts prefer perspective views of the cube that have two or three infinity points (ideal points) rather than the common head-on view that has only one ideal point.

Increasing the projection distance, d, reduces the perspective difference in the shape of objects and the parallax. If the viewing distance, d', stays fixed and the image is magnified so the scatterplot retains the same depth (the largest parallax is the same), then the 3-D scatterplot will have reduced perspective. Thus an infinite family of projections provides continuous control over the amount perspective in an stereo image. All the projections are compromises but then no static stereo image is “correct" under in the multiple-focal-point scenario.

A stereo projection with no perspective may be called the *depth-cued orthogonal projection*. This projection results from moving the head in front of each data point before

projecting the point. Thus the lines from the midpoint between the eyes to the projective plane are all perpendicular to the plane. Carr and Littlefield (1983) describe this projection and Carr et. al (1986) provide color anaglyph illustrations. Coordinate computations for the depth-cued orthogonal projection is straightforward using conventional plotting tools. The left eye and right eye coordinates of a point are given as ($x - x_p$, y) and ($x + x_p$, y) in data units. The half parallax, x_p, in x data units is computed as

$$x_p = k \frac{\text{range(x)}\ (z - z_{min})}{\text{range(z)}}. \tag{6.8}$$

With this representation conventional graphics scaling produces the desired coordinates for the plot. When plotting the points, care must be taken to prevent the points plotted last from occluding previously plotted points. This problem is readily addressed on devices that support color mixture tools such as alpha blending and bit-plane masks.

The value k determines the fraction of the plot width used as parallax. Carr et al. (1986) suggest the value $k = .026$, but this should be modified for extreme viewing conditions. Trial and error selection of k works well. The parallax in the depth-cued orthogonal projection can be viewed as a first order Taylor series approximation to the parallax of the perspective transform (6.3) since

$$p = \frac{e\,d}{d+z} = e\left(\frac{z}{d} + \left(\frac{z}{d}\right)^2 + \left(\frac{z}{d}\right)^3 \cdots\right) \tag{6.9}$$

Thus, the two projections are similar when d is much greater than the extreme value for z.

6.7 Interpretation Issues:

Motion parallax often facilitates investigation of 3-D data structure. However the very motion that provides the perception of 3-D structure inhibits the detailed study of such displays. Consequently depth-cued 3-D scatterplots have an interpretational advantage over motion parallax 3-D displays.

The construction of 3-D scatterplot should reflect the interpretation goals. If the dominant goal is perception of interpoint relationships such a clusters, holes, strings and surfaces, then axes are superfluous. However, if the units of measurement help to relate the data to other information, then axes and the ability judge approximate values from the plot become important. While using a mouse to locate stereo points provides one approach to determining values in data units, having the axes available provides immediate input concerning magnitudes. The depth-cued orthogonal projection allows the x and y coordinates can be read as in ordinary scatterplots and the distance between points can be assessed without mental adjustment for perspective. The primary drawback of this projection is the difficulty in comparing the depth of points to a scale. Of course this can be addressed by rotating the display cube or by providing multiple display cube views.

6.8 Stereo Production Methods and Common Problems:

The drive to produce stereo images as led to an amazing list of alternatives. The methods can be grouped into three classes, methods that provide continuous images to both eyes (*continuous-source methods*), methods that provide alternating images to the left and right eyes (*left/right multiplexed methods*), and methods the flash images on a screen or mirror that changes in depth (*depth multiplexed methods*).

Image brightness is a problem for most stereo production methods. Brightness is lost when color filters or polarized filters separate the images and when the images are not continuously present. Since the visual system can adapt to the general illumination level, reducing the room lighting ameliorates the problem for stereo workstations.

Interocular crosstalk is a second common problem. *Crosstalk* refers the situation in which the right eye observes the left eye image or vice versa. Crosstalk occurs when filters or opaquing devices do not completely block out the opposite eye images. Many factors, such as phosphor persistence in left/right multiplexed methods, affect crosstalk. Lipton (1982) notes that crosstalk produced ghosting can be mitigated by a number of factors, such as image brightness, contrast, textural complexity and horizontal parallax. As a contrast reducing example in color anaglyph stereo, Carr and Littlefield (1983) describe adding as small amount of the filter color to the background to match the light intensity of the ghost for the particular eye. The ghosts still have a different hue but small hue differences are relatively easy to ignore.

Left and right image alignment can be a problem for some stereo production methods. For example, two slide projector systems can suffer from slide alignment and keystoning problems. Keystoning can be ameliorated by making the two projectors as close together as possible, for example, one above the other. Stereo enthusiasts often use special projectors that allow both images to be precisely mounted in the same slide frame.

Most stereo production problems can be ameliorated. However some production systems are better than others. In today's world the continuous, separate-eye views provided by virtual reality headsets seems to the production method of choice for individual viewing.

6.9 Virtual Reality:

The development of virtual reality as a graphics construct began in the mid-seventies at Wright-Patterson AFB with attempts to develop more realistic flight simulation. The standard paradigm was to have training pilots look at a visual display screen. The effect, as might be expected, was that these pilots were aware that they were looking at a visual display screen and, consequently, had a comparatively strong sense of unreality about the whole exercise. An approach to overcoming this problem has become known as *virtual reality.* The intent of the virtual reality construct is to create a more intimate sense of involvement (originally in the flight simulation). Virtual reality constructs can be implemented in several ways. The goal is to recreate via computer the items that the subject's senses might sense in real reality. Visually, this would mean replacing a flat display screen with binocular stereo views which would change along with the motion of the subject's head. That is, if the subject looked down, he or she would see the view which is below. If the subject looked backwards, he or she would see the the view in back. This is accomplished with a headgear containing miniature binocular video displays. There is an audio component which can be achieved in the same

headgear by incorporating binaural headphones. Finally, one would want to create a tactile component which can be achieved by incorporating instrumented gloves. The technology is only now becoming commercially available, but has been demonstrated for the flight simulation application as well as for an interesting application for giving a sense of liberation to handicapped individuals.

The major new thrust in statistical data analysis and visualization is to combine scientific visualization, exploratory data analysis and virtual reality into a new technology for exploring data. There is a history of several years of experimentation with 3-D visualization and many aspects of this tool are in hand. Data analysis has seen movement of two-dimensional paper-based graphics in the seventies to the computer based three-dimensional color graphics of the eighties. Three-dimensional graphics have been achieved by kinematics displays where motion gives the depth cues and by stereoscopic displays. We have used both red-green stereo and polarized-light stereo to good effect. However, as satisfying as these displays are, they clearly represent the paradigm of a scientific investigator looking at a screen. What we are doing now is to leave this scenario and immerse the scientific data analyst in a data world. Allowing the analyst to literally move about in the data world, to fly around the data, to look at different portrayals of the data, to make a turn and literally turn into another dimension seamlessly. We would imagine a *direct manipulation* setting where an analyst can actually grab hold of a data point in this virtual world and see the effect of moving it around. We would imagine a *progressive disclosure* setting in which, as the analyst approached a data point in this virtual data world, the numerical value or other attributes of the data point would come into focus. What we are proposing is to create a data analysis environment where the scientist could quite literally explore the data much the same way that an oceanographer with scuba gear or a submersible could explore the sea.

Indeed, this undersea exploration analogy is a good one. To understand the potential impact, one can imagine an oceanographer who wishes to explore the ocean. On the one hand, the oceanographer could create a remotely piloted vehicle with a television and light system and watch the exploration process on a video monitor within a darkened room in his ship. This is to a large extent what computer-based visualization does now. A second model might be called the *scuba divers model*. In this model, the oceanographer straps on a scuba tank and other associated gear, dives in the water and explores the ocean directly. This is a qualitatively different experience and has historically yielded a different type of result. A third model we might call the *aquarium model.* The scuba divers model obviously has the handicap of being encumbered with clumsy equipment and such distracting concerns as running out of air. The data analytic analog has the same kind of concerns with wearing a tethering helmet apparatus with limited visual resolution. The aquarium model is an intermediate stage and is analogous to visiting an aquarium with a 40 foot glass wall. We are currently developing a 20 foot stereoscopic projection system which will allow the data analyst to view the data through a very large window unencumbered by requirements to wear physical hardware.

Conceptually what we are developing is easy to grasp. We are proposing a human-computer interface system which will allow the human operator control of his or her motion through the virtual data world simply by movement of an instrumented hand or controller. The binocular visual and binaural audio headgear will also be equipped to sense position and motion. The computations to update views are, in principle, straightforward three-dimensional

Euclidian geometry computations. The basic limitations tend to be the accuracy and resolution of the input devices and the throughput of the rendered images. Systems that use a high-powered graphics workstation for each eye still run below 30 frames a second for complex images.

7. Principles of Graphics Construction

7.1 Terminology and Graphical Goals:

The terminology for describing graphical displays is not consistent. Different disciplines tend to develop their own vocabulary and statistical graphics has borrowed from many disciplines such as cartography, computer science, geometry, cognitive science, graphic design and art. The limited terminology presented here was suggested by Cleveland (1985) and provides a basic page description. Cartographic alternatives and extensions can be found in Bertin (1983), Robinson, Morrison and Sale (1978), and Dent (1991). Dondis (1973) provides one reference from the world of art. Suffice it to say that terminology abounds and that there is room for establishing consensus and making new distinctions.

Figures 7.1 and 7.2 introduce terminology that describe basic elements of common two-dimensional plots. The figures define the terms by example. The plotting symbols represent observations or summary statistics. A key shows the plotting symbols used and data labels indicate data classes denoted by different plotting symbols. Markers and reference lines call out particular values to help in the understanding and evaluating the data. The data region is the portion of the plot in which symbol locations represent values. Vertical and horizontal scales provide the basis for interpreting the symbol locations. A scale label provides the name of the variable that the scale represents and often includes the units of measurement. Tick marks and tick mark labels prove specific values for comparison and indicated the scale of the representation. A title and legend typically provide an overview description of the plot and may call attention to particular details of interest.

Two key concepts for representing data are *superposition* and *juxtaposition*. Figure 7.1 shows two superimposed curves. Figure 7.2 shows juxtaposes curves by plotting separate panels on the same page. Superposition, juxtaposition, and locations of symbols relative to scales all involve visual comparison. Visual comparison is fundamental in statistical graphics.

The primary objectives of statistical graphics are the interpretation of data and data summaries. The interpretation of data involves relating the measurements to knowledge of how the measurements were obtained, to knowledge about the subject matter including related variables, to statistical models and to the variation of the data elements relative to each other. The objectives for a particular plot motivate the plot construction. When the goal is to assess the measurements in view of direct empirical knowledge about the phenomena and the measurement process, showing the units of measurement is important. Other objectives motivate re-expressing the original data.

Often the analysis goal is to assess the data relative to a statistical model. Typical statistical models include deterministic and stochastic components. Assessment of the data relative to a statistic model involves the decomposition of the data into fit (a deterministic

component) and residual (a stochastic component). Study of residuals helps in assessing the adequacy of the decomposition. Characterizing the distribution of the residuals provides a basis for indicating how well the fit is known. A major contribution of statistics is to provide a basis for assessing the quality of estimates by considering observations in relationship to each other.

Simplistically, statistical graphics has two focal points, presentation and discovery. In terms of presentation, showing the information in layers is often helpful. Unfortunately the first layer, the fit or basic estimate, is often all that is shown. Confidence intervals, upper and lower confidence surfaces and other indications of knowledge and data quality limitations form a second layer that should be included for the first layer to be evaluated. A “beautiful" rendering of the ozone layer above the earth can be generated from simulated data, ten observations, or a billion observations using defective equipment or inappropriate algorithms. Unless additional information is presented concerning data quality and statistical variability, a plot cannot be trusted to have scientific meaning. When statistical graphics is focused on presentation, a driving goal should be to indicate how good the summaries are.

This paper primarily describes the tools of discovery. We want to characterize distributions of observation or residuals. We want to find apparent deterministic structure that provides a parsimonious description of the data. We use cross validation and other techniques to help decide if a description is happenstance or if it will stand up when additional data is collected. Comparison underlies these processes.

Graphical methods allow us to exploit the power of our eye-brain systems to make comparisons. The two techniques juxtaposition and superposition are fundamental comparison techniques. However as our knowledge of the eye-brain system evolves, so will our choice of graphical methods. We are not good at all graphic tasks. Juxtaposition is a limited tool since our short term visual memory is limited. Comparing a plot in focus to a plot in memory is not one of our strengths. We are better at comparing two things in close visual proximity. Thus superimposed views are often advantageous relative to juxtaposed views.

Unfortunately superposition does not necessarily provide an optimal representation even in those cases in which overplotting and clutter are not problems. Our eye-brain systems evolved in a 3-D world and the compromises of our various visual processing subsystems lead to a variety of optical illusions and incorrect interpretations. Figure 7.3 provides a particularly instructive example that has been adapted from Cleveland and McGill (1984a). This example shows that we do not assess the difference between curves in the technically correct vertical direction. Our visual “hardware" computes distances by looking for closest points between curves in any direction. Thus we can be easily fooled by superimposed curves, and this phenomenon surely applies to more complex situation such as the comparison of superimposed surfaces. As a consequence, many discovery-oriented statistical displays represent differences and relative differences directly. The modifications to facilitate our visual processing often lead us away from the scales of the original data.

An recurrent theme in statistical graphics concerns transformations that simplify the statistical model or visual comparison standards. As one example from modeling, we often use a log transformation to make a right-skewed positive valued distribution more symmetric so that the our more familiar symmetric models will be appropriate. As a example of striving

for simple visual standards consider comparing the distribution of observed data to a theoretical model. We might try overlaying an empirical density estimate and the theoretical density function, but comparing such can be visually complex. Consequently we start our visual comparison using a theoretical quantile-data quantile plot. (See Section 8.3). This transformation of the data and reference model is particularly convenient visually. When the matched quantiles fall close to a straight line, the data can be described as coming from the same family as the reference distribution up to location and scale parameters. Because we are not particularly good at assessing the slope and intercept of a straight line fit to the data, we often add a robust regression line to the plot. (The regression uses the theoretical quantiles as the independent variable because they are measured without error.) When the straight line fits reasonably well, the estimated intercept and slope have an immediate interpretation relative to the theoretical density. To assess discrepancies from the theoretical density we subtract the estimated line values from the data quantiles and look at the residuals. This simplifies the visual reference line by making it an implicit horizontal line. Because we are quite good at judging positions along a common scale (Cleveland, 1985), this visual representation is a reasonable stopping place.

7.2 Perception:

Human visual perception is complex. This brief description provides background light and eye function. The background motivates discussion of several facets of perception, color perception, depth perception, texture perception, and accuracy of feature extraction.

7.2.1 Light and Focus:

Light consists of photons that travel at c = 299,694 kilometers per second in a vacuum. Photons are characterized by their wavelength, frequency = c/wavelength or, equivalently, their energy. Photon energy is proportional to frequency so the three descriptions have a one-to-one correspondence. The description here is in terms of wavelength. Visible light includes all photons of wavelength between 300 and 700 nanometers (nm), or billionths of a meter. The color of photons in the visible spectrum is related to their wavelengths with 460, 530, and 650 nm nominally corresponding to blue, green and red respectively. In reality, perceived color is usually a function of the dominate wavelength, so that there are many spectral mixes which lead to the same perceived color.

Feynman's book, QED (Quantum Electrodynamics, Feynman, 1985), describes in layman's terms the curious interaction between light and objects. The book provides a deeper, unified insight into familiar simplified accounts of reflection, absorption, refraction, diffraction and polarization. Feynman's account describes interaction of light with matter on its path to retinae. The current brief description picks up with refraction by components of the eye.

The refraction of the lens-cornea components of the eye focuses images on the retinae (see Figure 7.4). A simple optical lens (see Figure 7.5) provides a model that helps explain the focusing of images. Rays of light from an object at optical infinite arrive at simple lens and converge on a plane behind the lens. The distance from the lens center to the plane of convergence is called the *focal length of the lens*. The reciprocal of the focal length is call the *power of the lens*. When the focal length is measure in meters, the units of power are called

diopters. The relationship is power = 1/object_distance + 1/focal_plane_distance. When objects get closer to an optical lens, the focal plane moves further away. For the eye, the distance to the retina is fixed and the image is to be focused on the retina. Consequently the power of the eye's lens must change to focus objects at different distances on the retina. The power of the eye's lens is around 59 diopters of which the front cornea surface provides 49 diopters. Ciliary muscles changes the thickness of the lens to provide a variation in power of about 15 diopters.

The feedback process that allows the eye to focus on images at different depths on the retinae is called *accommodation*. Accommodation usually works in conjunction with eye convergence which is described below. As people age, their ability to change lens thickness diminishes and, as some of us know, many people obtain bifocal lenses to facilitate focusing on images at different distances.

The retinal cells have thickness and images that focus within this thickness are considered in focus. The phrase *depth of field* refers to the range of distances for which objects are in focus. When the human eye is accommodated at a distance of 14 meters, objects from 7 meters to infinity are in focus. The focus of objects depends on many factors including the lens system and retinal shape. The distance from the lens to the retinae will vary from person to person. If a person cannot change the power of the lens sufficiently to focus the image of an object on the retinae, the image will appear out of focus. Short- and far-sighted people have a mismatch between the range of power provided by the accommodation of the lens and the distance to the retinae. People wear glasses to correct this problem and in some cases to ameliorate problems associated with imperfections of retinal shape.

The focus of objects also depends on the wavelength. Short wavelength photons will “change course" more that long wavelength photons as they pass through a lens. The viewing multicolored objects involves some compromise of focus. A depth effect can be obtained by plotting blue and red lines on a surface. The effect is too small to be used to much advantage. However, some people enjoy the sharper focus provide by wearing glasses that filter out blue.

Object focus is not usually considered as a statistical graphics design factor for construction of static 2-D plots. Most people take responsibility for optical correction of their vision to the 20/20 standard. Object focus is a consideration for dynamic stereo plots because accommodation and eye convergence are often coupled as described in Section 6.1.

7.2.2 The Retinae:

The retinae is a curved mat of photoreceptor cells, call rods and cones. Photons must strike the cells in sufficient quantities trigger electrical response. Visual acuity is a function of the packing of cells and processing of cell responses to photons. The 120 million rods that surround the center of the eye are associated with low light intensity, achromatic vision. The output of neighboring rods is pooled, thus visual acuity is low. The cone-rich central area of the retina is call the macula lutea. The macula lutea contains the fovea (about 1 degree of arc) that is almost exclusively composed of cones. The six million cones cells provide high acuity color vision.

7.2.3 Eye Movement:

The eyes are never at rest. Beyond head movement, the eyes move in four distinct ways. First the eyes are held in dynamic balance by three pairs of antagonistic muscles. Instability in this balance causes a continuous small-amplitude tremor (Bruce and Green, 1990). The tremor shifts a fixed image between adjacent fovea cones on the order of .1 seconds. When an image contains regularly spaced high contrast lines, physiological tremor will make the image appear to vibrate. In art, this falls under the label *moire effects*. Tufte (1983) presents common examples of statistical graphics images that appear to vibrate. He suggests that vibrating images have no place in statistical graphics.

When a person reads a book or looks at a image the eyes make rapid, intermittent jumps called *saccades* to fixate objects within foveal vision. This is the second type of eye movement. The third and four types of movement, pursuit and convergence, are smooth and continuous. When the eyes focus on a moving object they must move to keep it in foveal vision. *Pursuit* refers to eye movements that occur when the moving object remains a fixed distance from the viewer. The eyes move in the same direction. *Convergence* refers to movements that occur as eyes follows an object that changes distance. When an object is at infinity the lines of sight for the eyes are parallel. As the object moves closer, the lines of sight cross at progressively larger angles. Convergence is a factor in the design of stereo images in two ways. First, divergence beyond 1° of parallel is physically difficult. Ideally the eyes should not diverge. Thus if the eyes are to focus on corresponding points in left/right images so that the points appear behind the display surface, the points should not be further apart the eye separation.

7.2.4 Color Perception:

The cones, which are responsible for color perception, fall into three types, those most sensitive to blue, green and red portions of the spectrum. The absorption peaks for these three types of cones are approximately 419, 531 and 558 nm respectively. Perceived color is a function of the relative response of retinal cells to photon stimulation. A consequence of the relative response is that light mixtures can produce the same perceived color as monochromatic light. A further factor in color perception is that cells decrease response under conditions of continuous stimulation. For example looking at a blue saturated image will decrease the response to blue. If the image changes to a white, the viewer will see the complementary color because the relative response to the blue in the white mixture has been reduced. This after-image phenomena quickly disappears as the blue sensitive cells adapt to the new mixture of light. The concepts of relative response and adaptation provide a useful heuristic for understanding perceived color phenomena. For example an annulus of one color about a disk of another color modifies the perceived color of the disk. Land (1977) describes perception of color in terms of the reflectance of objects and provides a good introduction into the complexities of color perception.

In statistical graphics, many have tried encoding information in color. Experience has shown that a few groups can be effectively distinguished using color (see Morse, 1979), but that color is a relatively poor choice for representing one (or more) continuous variables. While some of the difficulty in representing a continuous variable may be associated with unintended results that come from the relativistic nature of color perception, but much of the difficult comes from the fact that humans do not order colors along a linear scale. Shepard and

Carrol (1966) use multidimensional scaling to obtain a two-dimensional order of colors. Such studies suggest that wavelength (rainbow) and other well-intended orderings are effective only to the extent that the ordering is learned through repeated use. Restricting colors often facilitates ordering colors. In cartography, value (or gray scale) provides a consistent and effective ordering. Color saturation also provides a reasonable ordering. However, the number of distinctions that can be made with only these scales is limited. Consequently, color is one of the last resorts for representing a continuous variable.

7.2.5 Visual Subsenses and Preattentive Visions:

Knowledge about human visual processing helps in the design of effective statistical graphics. Friedhoff and Benzon (1989) report that there appears to be at least three major distinguishable visual processing channels (see also Zeki, 1992). One channel carries color information but not detailed shape information. A second channel channel carries high-resolution information about shapes. The third principally handles movement and stereoscopic depth information. Good graphical designs will typically avoid conflicts between these channels. For example, Friedhoff and Benzon show an example in which the color selection competes against other depth information.

An important question in the design of statistical graphics concerns what draws visual attention. Julesz (1986) addresses the question of foveal attention in his theory of textons. The theory distinguishes between preattentive vision and attentive vision. Preattentive vision works in parallel to find almost instantaneously the location of abrupt changes in texton density. Textons include color, angular orientation, object width and length, binocular and movement disparity and flicker rate. As an example, consider color as a texton. A blue point in a field of red points provides an abrupt change in texton density. Such abrupt change locations become foveal attention locations. Surprisingly, scrutiny in attentive vision is required to reveal the nature of the textons that have drawn attention to particular locations. Scrutiny in attentive vision is characterized as serial search by a small aperture of focal attention in steps lasting 15-50 ms each. This is still much faster that eye-movement scanning, but slower than preattentive, parallel search.

Theories, such as texton theory, may be eventually be replaced by more inclusive or parsimonious theories, but the visual phenomena they attempt to characterize are real. The theories suggest effective design principles for statistical graphics. Texton theory suggests methods for “highlighting" specific points in images.

7.3 Statistical Graphics versus Realistic Imaging:

To a certain extent, as we have pointed out above, there is a dual goal associated with statistical graphics and visualization: one, a quantitative assessment of data; the other, a qualitative assessment of structure. To a very large extent, traditional presentation graphics involving pie charts and bar graphs and the like have emphasized the qualitative aspects. Combined with icons, these devices have been used and often misused in a popular format to convey the qualitative aspects of statistical data. As a response to a perception of the low quality of much of presentation graphics, there has been a substantial reaction in the statistical literature emphasizing quantitative aspects of statistical graphics. See for example Tufte (1983, 1990), Cleveland and McGill (1984a) and more recently, Becker and Cleveland

(1991a,b). While we agree that extraction of quantitative information when possible is a desirable goal, it is worth recognizing that much of the use of statistical graphics and visualization is targeted at exploratory qualitative assessment preparatory for building more analytical and quantitative models of the data. Phrased slightly differently, much of statistical practice is focused on assessing nominal behavior and variability from that nominal behavior, means, variances, distributions or their robust analogs. On the other hand, much of the general scientific enterprise is focused on assessing the relationship among measured variables and seeking to assign some structural model to the relationship among these variables. Often, this is a qualitative process rather than a strictly quantitative process. For example, a density function with three modes might suggest that we should try to model the generating process by a mixture model with three mixing variables. The precise location of the modes and the exact characterization of the tail behavior of the density may be more or less irrelevant to the supposition that we need three mixing variables.

Thus we have been to a large extent advocates of a perspective that sees considerable value in the qualitative aspects of statistical visualization. Of course, we believe that this should be combined with a rigorous quantitative modeling process. The visualization of complex structure, we have found, requires techniques which move beyond simple two-dimensional plots, static wire-frame models and even dynamic rotating scatterplots. To this end we have focused in our previous discussions within this paper on relatively sophisticated computer graphics techniques which go beyond the standard techniques needed for traditional statistical graphics methods. We have, indeed, adopted a philosophy which treats relatively abstract mathematically defined statistical structures, such as density surfaces, density contours and regression surfaces, as real physical surfaces which may be assigned physical properties and lit and rendered accordingly. As indicated in our earlier discussions, one can use these techniques to good effect in two ways: 1) the resulting images are often extremely useful in qualitative if not quantitative assessment of structure and 2) the resulting images are often visually dramatic, even exciting and consequently serve to stimulate interest in the graphical analysis among layman and expert alike. The value of this latter effect is not to be diminished.

In the next several sections, we review some of the more traditional statistical graphics techniques many of which require little or no computer power. Many of the graphical methods used in statistical graphics are well known and relatively elementary from an algorithmic point of view. That is not to say that they are simple to construct because there are difficult, even profound, design issues associated with statistical graphics. The process of inventing an effective graphic is not a casual, trivial process. Nor is it a fully algorithmic, scientific process. We shall not attempt to treat further the elements of graphic design except as discussed in Section 7.1. There are several excellent books available on the topic. These include the previously mentioned books by Tufte (1983, 1990) and also the book by Cleveland (1985). We refer the reader to these works in connection with issues of graphic design and style. In our discussion following we shall allude to a number of well-known techniques with relatively little discussion except to note the existence of these methods. Our purpose, as with the earlier parts of the paper, is to record computational algorithms, particularly where these are subtle and less well known.

8. One-Dimensional Variables

For one-dimensional variables, a primary goal is to discover information about the distribution or often more intuitive density function associated with the observations. Perhaps the earliest graphical forms of the distribution is the histogram. The one-dimensional histogram has a long history and, indeed, is the staple method for conveying distributional information among the majority of non-statistical scientists. It is computed by partitioning a segment of the real line, normally a segment just slightly larger than the range of the data, into a number of equal intervals called *class intervals*, by counting the number of observations in each class interval and by plotting a rectangle whose base is the class interval and whose height is the count of observations in the class interval. A minor variation is to re-scale the histogram so that the total area of the rectangles is one. The scaled histogram is, then, interpretable as a probability density and empirical probabilities can be read from the scaled histogram. There have, of course, been many refinements of the histogram.

8.1 Density Estimation:

Since the mid-1950s, density estimation has been an extremely popular subject with an explosion of interest in the early 1970s. A large number of books have been written on the topic and include Tapia and Thompson (1978), Prakasa Rao (1983), Silverman (1986), and Scott (1992). An early review article was Wegman (1972a, b) and a more recent one is Izenman (1991). Various parametric approaches to density estimation have existed since as early as 1890. All these early schemes involve systems of distributions or frequency curves which are intended to represent as wide a variety of observed density as is possible. Parameters are estimated and the resulting density is taken as the estimated density. Perhaps the best known system is the Pearson system originated by Pearson between 1890 and 1900. A Pearson density satisfies a differential equation of the form

$$\frac{df(x)}{dx} = -\frac{(x-a)f(x)}{b_0+b_1x+b_2x^2}. \tag{8.1}$$

If f is unimodal, the mode is a and f has smooth contact with the x-axis as f tends to 0. A thorough treatment of this system is given in Elderton and Johnson (1969) or in Johnson and Kotz (1969-1972). Other approaches include systems based on expansions of densities in orthogonal series (Gram-Charlier), based on translations (Johnson), and based on representations of the distribution (Burr). See Johnson and Kotz for a treatment of all of these.

Receiving far more attention since 1956 have been the nonparametric approaches. One may distinguish four main traditions of nonparametric density estimation: 1) kernel methods, 2) orthogonal series methods, 3) maximum likelihood methods, and 4) spline methods.

The smoothing kernel approach has been the most thoroughly developed theoretically and has an extensive literature. Much of the early and contemporary literature is written in the tradition of mathematical statistics where emphasis is placed on obtaining asymptotic results. From the perspective of a graphical data analysis and visualization, smoothing kernel methods are just that: convolution methods for obtaining smooth functions which may or may not be true underlying densities. If we consider $x_1, x_2, \ldots, x_n$ be a sequence of observations thought to be distributed according to an unknown density, f, the general kernel estimate has the form

$$\hat{f}_n(x) = \int_{-\infty}^{\infty} K_n(x, y)dF_n(x) = \frac{1}{n}\sum_{i=1}^{n} K_n(x, x_i) \tag{8.2}$$

Here F_n is the empirical distribution function based on the first n observations. The idea of these estimates is the following. The empirical distribution function is a discrete distribution with mass l/n placed at each of the observations. (8.2) smears this probability out continuously, smoothing according to the choice of $K_n(x, y)$. Thus the choice of $K_n(x, y)$ is significant and to a large extent determines the properties of $f_n(x)$. The first published work on estimates of this type was that of Rosenblatt (1956). Rosenblatt considers a naive estimator,

$$\hat{f}_n(x) = \frac{F_n(x+h) - F_n(x-h)}{2h} \tag{8.3}$$

This estimate is the special case of (8.2) when $K_n(x, y)$ is 1/2h for $| x - y | \leq h$ and 0 elsewhere. Of course, no assumptions are necessary on f simply to form this estimate. Asymptotic results do require assumptions, however. In general, if f is the true unknown density with appropriate continuity conditions, the problem is to choose the sequence of $h = h_n$ converging to zero at an appropriate rate so as to minimize mean square error (MSE). If $h_n = kn^{-\alpha} > 0$, the choice of α minimizing MSE is 1/5 and the optimum value of k is

$$\frac{9}{2} \frac{f(x)}{|f''(x)|^2}.$$

Since we are attempting to estimate f, it is unlikely that we will know enough to choose optimum k. Nonetheless, we have found that a satisfactory choice of the constant $k > 0$ is relatively easy to determine experimentally for purposes of visualization. A considerable amount of attention has been paid to developing methods such as cross validation for choosing h_n and can be further studied, for example, in Silverman (1986). Refined estimates of the convolution kernel type are usually expressed as

$$\hat{f}_n(x) = \frac{1}{nh_n} \sum_{i=1}^{n} K(x - x_i) \tag{8.4}$$

where $h_n = h(n)$, $\int_{-\infty}^{\infty} K(u)\, du = 1$, $\int_{-\infty}^{\infty} [K(u)]^2\, du < \infty$, $\int_{-\infty}^{\infty} [K(u)] \mid u \mid^3 du < \infty$ and K(u) is symmetric about 0. Under these conditions, the condition that the observations are independent and identically distributed and the condition that f have derivatives of the first three orders, the optimum choice of h_n leads to mean square error $E(\hat{f}_n(x) - f(x))^2$ no smaller than $O(n^{-4/5})$. An example of a kernel density estimator is given in Figure 8.1 along with several other exploratory data summaries.

An alternative approach is to represent the density by means of an orthogonal series in the following formulation. Let us further consider a subset E of the real line spanned by the orthonormal basis $\{g_k(u)\}_{k \in I}$ where I is some index set. An orthogonal series estimate can be formed by letting

(8.5) $$\widehat{f}_n(x) = \sum_{k \in I} < g_k, f > g_k(x)$$

where $< g_k, f > = a_k = \int_{\mathbb{R}} g_k(x)\, f(x)\, dx$. The coefficient a_k can be estimated in the obvious way by

(8.6) $$\widehat{a}_k = \frac{1}{n} \sum_{i=1}^{n} g_k(x_i).$$

Note that if the g_k are chosen as indicators of disjoint intervals, this estimator is essentially a histogram. More interesting cases exist when the $g_k(x)$ are chosen as an infinite orthonormal series. One such sequence is the normalized Hermite series

(8.7) $$g_k(x) = (2^k k \pi^{1/2})\, e^{-x^2/2}\, h_k(x),\ k = 1, 2, \ldots$$

where $h_k(x) = (-1)^k e^{x^2} (d^k/dx^k)(e^{-x^2})$. This form was originally suggested by Schwartz (1967). Kronmal and Tarter (1968) consider an estimate based on trigonometric functions rather than the Hermite functions. The form of their estimate is (8.5) where $\{g_k(x)\}$ are chosen as one of the orthogonal systems $\{\sin(\pi kx)\}$, $\{\cos(\pi kx)\}$ or $\{e^{i\pi kx}\}$. More recently, Wegman (1992) has suggested orthonormal wavelet bases.

A third rather different approach is the maximum likelihood approach and the related approach based on splines. These methods have received some substantial attention in the statistical literature but are computationally intensive and, thus, are somewhat less valuable as tools in statistical graphics and visualization.

8.2 Binning Methods in One Dimension:

As a precursor to our discussion of two- and multi-dimensional methods, it is worth mentioning briefly other variants of the histogram. Histogram-type binning methods have the following computational advantage over kernel smoothers and orthogonal series methods of density estimation. Suppose the histogram range is, for purposes of discussion, the interval, (0, u), and we wish to create k class intervals. Then the class intervals would be (0, u/k), (u/k, 2u/k), ..., ((k − 1)u/k, u) which we can label with indices 1, ..., k. The integer part of x_i/k, $[x_i/k]$, is just the index of the class interval to which observation x_i belongs. Thus constructing a binned (histogram) estimator is a simple one-pass through the data involving a division, a floating point to integer conversion and an accumulation operation. This algorithm involves no conditional branching or floating point calculation of a transcendental or polynomial function. Thus it is extremely fast when compared with traditional density estimation methods. The negative aspect of histogram estimators is that they may smooth too much where the data is most concentrated (and hence fail to reveal fine detail) and may smooth too little in the tails. A simple historgram is illustrated in Figure 8.1.

Wegman (1975) suggested a data-adaptive, variable bin-width histogram. The simplest version involved partitioning the data set rather than the range of the data into equal size groups; i.e. if there are n observations and k class intervals, take the [n/k] smallest

observations to form the first class interval, the next [n/k] smallest observations to form the second class interval and so on. Here [x] is the greatest integer function as above. This formulation allows for long class intervals where the data is sparse and short class intervals where the data is dense. Actually Wegman (1975) proposes a wide class of histogram-type which anticipated sieve methods.

Scott (1985) suggested another histogram-based estimator known as the *average shifted histogram*. The idea is to compute a series of conventional histograms, say k of them, each displaced a small amount, say δ, so that kδ is equal to the binwidth. The resulting k histograms are averaged together. The average shifted histogram has the property that it is asymptotically equivalent to a kernel smoother, but has the computational advantage of the binning estimators.

It is worth noting that Tukey (1977) proposed a manual version of a histogram-like estimator called the *stem-and-leaf* which is widely discussed in the exploratory data analysis literature.

8.3 Quantile Plots:

Another method for investigating distribution properties is the *quantile* (or *percentile*) *plot.* The idea here is to arrange a data set, $x_1, x_2, \ldots, x_n$, in ascending order so that it is represented by the ordered observations, $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$. The quantile corresponding to $x_{(k)}$ is $(k - c)/(n + 1 - c)$, $c \in [0, 1/2]$ and the percentile is 100k/n. The *quantile-quantile plot* or *q-q plot* is used to investigate the distribution associated with the x's. It is formed by calculating the quantiles associated with a known distribution, often a normal distribution, and plotting them against the sample quantiles. If F(x) is the theoretical distribution, then x_α is the α^{th} quantile if $F(x_\alpha) = \alpha$. Thus a quantile-quantile plot is simply a plot of the pairs $\left(x_{k/n}, x_{(k)}\right)$. If F is the correct distribution for $x_1, \ldots, x_n$ up to scale and location, then the q-q plot should result in a straight line. A normal quantile-quantile plot is also illustrated in Figure 8.1 which is a so-called EDA View Plot. This plot is a macro in the Splus statistical computer software.

A two-dimensional variation which we mention here for completeness replaces the theoretical quantiles associated with F with the sample quantiles from a second data set, say $y_1, \ldots, y_m$. Note that if $m \neq n$, some approximations and/or interpolations have to be made to obtain quantiles from each distribution corresponding to the same probability points. Often quantile plots are combined with transformations on the data variable in order to more usefully approximate a known distribution.

8.4 Box Plots:

The box plot is a simple summary that facilitates making distributional comparisons. Figure 8.2 provides an example that compares the distribution of albumin levels for five age groups. Another example of the box plot is given in Figure 8.1. The single box plot typically provides a five-number summary of a distribution and shows conjectured outliers. The box plot has many variations. Three summary numbers the 1st, 2nd and 3rd quartiles are common to all variations. The quartiles defined a box and the median is shown within the box. The variation shown in Figure 8.2 makes the width of the box proportional to the square root of the

sample size. The choices for the other two summary numbers vary across software and applications. Perhaps the most common choices are the two most extreme data values that fall inside the boundaries defined by the first quartile minus 1.5 times the interquartile range and the third quartile plus 1.5 times the interquartile range. The pairs of braces for each box plot in the figure represents these data values. Data values outside this interval are shown as line segments here and are conjectured outliers. The figure also has notches about the medians that represent approximate confidence intervals. When the notches do not overlap, the medians are different at the 95% significance level. In this example, only age group 35-44 and 45-54 are similar and otherwise, the albumin level decreases with age.

8.5 EDA Views and Transformations of Data:

A convenient first view of a mysterious continuous univariate data set includes a histogram and density plot to show the general distributional shape. The view also includes a normal quantile plot and box plot to focus attention on possible assymetry, tail thickness and outliers. Such plots as Figure 8.1 begin to address the applicability of conventional normal statistics methods. (A cautionary second view includes times series related subplots that may suggest serial correlation. Serial correlations when present seriously compromise most of the elementary inference procedures used in statistics.) If the data appears to have an appropriate normal distribution, description and inference (if appropriate under a sampling model) methods developed for the normal distribution become reasonable. If not, three standard choices are 1) to use nonparametric/robust methods, 2) to identify an more appropriate distributional family and use maximum likelihood methods, or 3) to transform the data so that a more convenient distribution family becomes appropriate. Prior knowledge about the data may suggest a distribution family or reasonable transformation.

A common scenario is that the data consist of nonnegative values and have a long tail. For such data the Box-Cox transformation given by (8.8) helps to achieve symmetry:

$$x_\lambda = \frac{x^\lambda - 1}{\lambda}. \tag{8.8}$$

The Box-Cox transformation is a power transformation which is indexed by a single parameter. Note that (8.8) is an indeterminant form when $\lambda = 0$. But by continuity $x_0 = \log_e(x)$. We have used this transformation to good effect in a high interaction computer graphics setting by creating a slider bar to set the value of the parameter. In this context, it is easy to see an appropriate value of λ to transform the data to some appropriate known distribution. If the tail thickness is not too extreme, normal theory methods may be applicable. The *symmetry plot* provides visual evaluation of symmetry. The coordinates for a symmetry plot are $(m - x_{[k]}, x_{[n+1-k]} - m)$ where m is the median and $k = 1$ to $[(n+1)/2]$. In a square plot with identical ranges for each axis, departures from a overplotted 45° line indicate departures from symmetry. Interactively altering λ using a slider and viewing the plots provides rapid, intuitive selection of a symmetrizing power transformation.

Analysts often transform data to achieve approximate symmetry, to stabilize variance across multiple distributions, to promote a straight line relationship between variables and to simplify the structure of a two-way or higher-dimensional table (Emerson and Soto, 1983).

Transformations as described in Daniel and Wood (1980), and Hoaglin, Mosteller and Tukey (1983, 1985) provide a tremendously powerful tools.

Primary cautionary notes about transformations concern: 1) the dangers of correlation, 2) questions of appropriateness upon further collection of data, and 3) difficulty in explaining the analysis to clients. The third item is particularly important. Clients are often reluctant to discuss the analysis if anything more complex than a log transformation is applied. When clients deal with familiar units of measure, they more readily bring related information to the exploration. Thus compromising a bit on the transformation and showing the units in the plots represent important statistical graphics strategies.

8.6 Labeled One-Dimensional Plots:

Many plots are available for showing labeled univariate data or statistics, with common variants being bar charts, pie charts and dot charts. While pie charts appear commonly in the business graphics, the statistical graphics community has responded to perceptual experiments that generally give the edge to bar charts. Bar charts have survived the test of time and are likely to remain popular. However, the dot charts as described by Cleveland (1985) consistently use readable horizontal labels and convey the same information with less ink (and, for some, with more elegance). Thus dot charts should see increasing use in the future.

9. Visualizing Multidimensional Data

The fundamental objective of the scientific enterprise is to discover the so-called “laws of nature." Suspending for the moment any consideration of stochastic aspects of the problem at hand, the scientist endeavors to take two or more variables thought to play a critical role in describing the phenomenon at hand and attempts to discover the mathematical relationship among these variables which describes the phenomenon and allows for predictions of new phenomena. Boyle's Law, Kepler's Laws of Planetary Motion, Newton's Laws of Motion and Gravitation, the Laws of Electromagnetism, Quantum Mechanics, Einstein's Special and General Relativity Laws are all examples of such laws of nature in the realm of physical science. In this sense, physics is an easy science because its traditional focus is on comparatively few variables which are contaminated with relatively little noise. Social science, biological and medical science, and economics, by comparison, tend to deal with many more variables which tend to be far more stochastic in character. When dealing with a relatively few variables, say 3 or 4, with little or no noise, traditional plotting tools such as line graphs or three-dimensional wire-frame surface plots are relatively satisfactory. The discovery of the “laws of nature" by these graphical devices is comparatively easy. However, as the number of variables and the amount of noise increases, many graphical methods originating prior to 1980 are inadequate. For the most part, the problem of discovering functional relationships among random variables has largely been approached through a combination of experimental design and linear models.

Experimental design and linear models are closely linked although it is clear that a carefully thought-out pattern of experimental sampling points (the objective of experimental design) plays a crucial role in nonlinear models and graphical design as well. Linear models are a powerful tool in multivariate data analysis, but far too often they are a default tool because no visualization techniques were available to gain insight into the structural functional

relationships (presumably non-linear relationships) among many variables. However, with the introduction of high performance workstations beginning around 1980, a good deal of thought has gone into creating graphics and visualization tools for gaining insight into the structural relationship among three or more variables. It is in this discovery process that, in our opinion, computer graphics makes its most valuable contribution to the scientific discovery process.

9.1 Scatterplots, Rotating Scatterplots and Scatterplot Matrices:

The scatterplot, perhaps the most obvious device for plotting data, is obtained for bivariate data, (x_i, y_i), $i = 1, \ldots, n$ simply by plotting the y_i against the corresponding x_i as a series of n points in a two-dimensional Cartesian plot. The scatter plot allows the scientist to inspect simultaneously for linear or non-linear relationships, clustering, and degree of correlation. The scatterplot is a simple tool, but very powerful in its ability to allow for inspection a key functional properties.

An easy generalization is the three-dimensional scatterplot for trivariate data, (x_i, y_i, z_i) which is obtained by plotting the triple as points in a three-dimensional Cartesian coordinate system. Unfortunately, the without some animation or other depth-cuing device, a three-dimensional scatterplot rendered in a planar graph loses any meaningful interpretability. Typical devices for depth cuing include kinematic displays and stereoscopic displays. Two principal forms of stereoscopic displays have been a) *side-by-side left-eye right-eye (stereo pair) plots* which some individuals have been able to fuse without assistance of an optical viewer, but which can more readily be fused by means of a stereopticon of the type common in the late Victorian era, or b) *red-green* (or red-blue) *stereo plots* viewed through filtering lenses common in comic books in the 1950s. An example of a 3-D scatterplot using the stereo pair plot is given in Figure 9.1. Both of these stereoscopic display tools were illustrated in Wegman and DePriest (1986) and are to the best of our knowledge the first use in the print media of stereoscopic displays for statistical graphics purposes.

A more common depth cuing device is the use of animation, typically *dynamic rotation of a three-dimensional scatterplot* by means of computer graphics. Rotation is relatively straightforward to achieve using the equations described in Section 2. The rotation animation gives a differential rotation rate to point depending on how close or how distant the points are from the axis of rotation (in three dimensions). However, the depth cue disappears immediately with the cessation of the animation. Moreover, this method depends upon availability of appropriate computer graphics hardware and so does not lend itself easily to extended study of a static plot as stereoscopic displays do. One technique used has been to replace full rotation with a *rocking* rotation of limited angular displacement. Rocking animation allows for a more extended study from a chosen perspective. The techniques of stereoscopic displays and rotation animation have been combined in several of the present authors' software development efforts, notably Mason Hypergraphics^{TM}, ExplorN^{TM} and Mason Ridge^{TM}. Animation has not been extensively used in print media for technical purposes, but at least one example of *flip animation* can be found in Wegman (1991).

Rotating scatterplots up the ante from two dimensions to three dimensions, but still do not offer much help for more general d-dimensional data. A scatterplot matrix for d variables is obtained by arranging the scatterplots for all $d(d-1)$ ordered pairs of variables in a matrix so that each marginal scale applies to $d-1$ plots. Figure 9.2 illustrates such a plot for four

variables. The originator of the scatterplot matrix does not seem to be known. Tukey and Tukey (1981) described an organized collection of two-variable scatterplots and called it a *draughtman's display*. Further descriptions are found in Chambers et al. (1983), Carr and Nicholson (1984) and Becker and Cleveland (1987). The display of multiple microplots facilitates the rapid scanning of many dimensions. In some cases extreme points can be followed from plot to plot. However, the disconnected representation of multiple aspects of the same multidimensional point complicates the discovery of higher-dimensional patterns. Friedman, McDonald and Stuetzle (1982) and many others have demonstrated the use of color as a linkage to identify selected points in a matrix of plots.

Because individual plots in the matrix are small and space is at a premium (owing to limited resolution in even the best of display terminals) the practical limitation on the scatterplot matrix is approximately 20 dimensional data and often less. Techniques involving color are particularly advantageous. One of the most useful techniques is interactive subset selection and representation using color. Comparing subsets provides a powerful approach to extracting more information for scatterplot matrices or multiple subplots. The comparison of subsets has two aspects, definition and representation.

Approaches to subset definition include algorithmic methods such as stratifying based on a variable (original or computed) and interactive graphical methods. Littlefield (1984) presents graphical subset definition by drawing a polygon around the selected subset in one subplot. Becker and Cleveland (1987) describe a power variant of this approach called *brushing* or *painting*. This technique has its roots in the work of Friedman et al. cited above. The generalization separates the tasks of fixing the polygon shape and positioning the polygon. For algorithmic simplicity, they choose a rectangle as the polygon and the user selects the size and the aspect ratio. Points inside the rectangle are selected as the user moves the rectangle. An option determined whether or not the selection was cumulative. This provides the basis for both point identification and live animation of evolving subsets. Typically, interactive graphical subset selection and representation is applied to relatively small data sets. The algorithmic and polygonal subset definition approaches are more often used with larger data sets. For large data sets, animation is made possible by separating the computations of the individual images from the display of the image sequence (Carr, Nicholson, Littlefield and Littlefield, 1987).

Representations of the selected and deselected subsets have evolved with the available technology. In the dynamic setting, selected points originally were typically distinguished by a filled dot versus an open dot and by a second color. The filled and open dot plotting of Becker and Cleveland was based on *bit-blit* technology for a monochrome workstation. This did not completely address the issue of overplotted points because some could be associated with the selected set and some with the deselected subsets. As color workstations became available, the common approach used overplot mode which determined pixel color by the point last plotted. Typically this point was from a selected subset.

To help assess overplotting, Wegman (1990) developed the *scintillation technique*. This technique cyclicly assigned a sequence of colors to data points. The data points are replotted at a controlled rate. Pixels on the screen that changed color represent more than one point and the rate of color change generally indicates the amount of overplotting. Carr et al. (1986) use bit plane masks and color table definition to represent the overplotting of (two or

three) subsets. Different colors represent the overplotting of the various subset combinations. Again the representation was enabled by the technology available. Carr et al (1987) addresses overplotting by representing the density difference between the two subsets. This is not as fast as previous methods, but considers the amount of overplotting and not just the subset membership. Now, α-blending provides new opportunities and challenges for overplotted points.

9.2 Glyphs and Icons:

Our definition of “glyph" is rather broad. We consider a *glyph* to be any plotting symbol that displays data by changing the appearance of the symbol. This concept encompasses a wide variety of graphical forms, including the original Anderson (1957) glyphs and metroglyphs and Chernoff (1973) faces, plus weathervanes, d-sided polygons (star plots), and castles and trees developed by other investigators. Fienberg (1979) gives an overview and references many of the specific forms. We will discuss general aspects of the glyph technique.

As with many other graphical techniques, glyphs have two main uses: exploration and presentation. The motivations and corresponding glyph designs are different. For presentation use, the display is intended to convey a specific impression. Some glyph designs, particularly Chernoff faces, seem well suited for this task. An example of the Chernoff face plot is given in Figure 9.3. One can carefully assign data dimensions to glyph features. There is little need to maintain neutrality or independence among dimensions. In addition, one can select the number of data points and dimensions as necessary to best display the important aspects. For exploratory use, the purpose of the glyph is to display as many dimensions and points as possible, in a neutral fashion. An ideal exploratory glyph would enable the viewer to consider any combination of dimensions simultaneously, from singly to all combined. It is important that the glyph display the data dimensions independently, and that no dimensions be allowed to overwhelm others.

Another important consideration is the tradeoff between glyph complexity and number of data points displayed. A glyph which works acceptably with 50 data points can become useless with 500 data points. There are two problems. The first is that the human visual system can handle only a few features (at a glance). If the glyph is so complicated that study is required to understand it, then it is possibly too complicated for use with large numbers of points. For interactive exploratory analysis we are dubious about glyphs that represent data in dimensions much above five. The second problem is that glyphs overlap as the number of data points increases, and the overlapping glyphs can be difficult to interpret. When many data points are viewed in the aggregate, global patterns are seen.

When viewing several variables, it is common to restrict the range of some variables and focus attention on the relationships among two or three other variables. For example, Cleveland (***The Elements of Graphing Data***, 2nd Edition to appear) describes multiple plots obtained by systematically conditioning on the range of other variables. Such plots are called coplots in Chambers and Hastie (1992). A similar, mental conditioning process facilitates the interpretation of multivariate glyphs. For a multivariate glyph plot, all the information is present in a single plot and the objective is to make comparisons among those glyphs that have restricted ranges for selected variables. This is difficult unless position or distinct colors represent the range restricted variables. For example, if the curvature of the eyebrow in a

Chernoff face represents a variable, it is difficult to focus attention on all the faces with eyebrow curvatures in a particular interval. It is much easier to focus attention on the left of the plot or on all red glyphs. With respect to choice of glyphs, we prefer to use symbol characteristics that facilitate mental conditioning and have relatively high perceptual accuracy of extraction (see Cleveland and McGill, 1984a). Our preferences for glyph characteristics are point position, stereo depth (if stereo is used), orientation, linear size, area size, and color (if desperate). Stereo provides point position and avoids a shifting of visual/mental gears even though relative depth may be assessed less accurately than relative orientation. For some people stereo is not an option. As an example of ray glyphs, Figure 9.4 shows nitrate levels plotted on a geographic map using ray glyph techniques. A ray glyph is typically a line segment attached to a point located in two or three dimensions by other variables. The angle, length and color of the ray glyph may be used to convey additional data. Indeed, several ray glyphs can be attached to the same point in two or three dimensions. In Figure 9.4, the two-dimensional data is geographic coordinate data.

Easily discernible horizontal and vertical orientations can convey the distributional aspects, central tendency and extremes, respectively. Each ray orientation represents the value of an observation in a sample. Think of the ray as a pointer on a measuring instrument, such as a speedometer, increasing from left to right. A linear transformation of the sample codes the median as vertical (90°) and the minimum as left (0°) for a skewed left distribution or the maximum as right (180°) for a skewed right distribution. Thus, all the values fall between 0° and 180°. Values near the median are discernible as near vertical. Skewness in the distribution is also discernible as a single orientation for near-horizontal rays. Preventing the rays from dropping below horizontal avoids ambiguity.

Ray length is less quantitative than orientation but can be used for non-spatial representation of an additional dimension. Length and color together do better. We have tried two approaches with varying degrees of success. Both use a linear ray length scale. For the first approach, length and color are used redundantly. Since a positive length is necessary for the ray to have an orientation, the minimum length for easy visualization of orientation corresponds to the minimum of the data. If rays are too long then the scatterplot becomes cluttered. Trial and error selects minimum and maximum lengths to correspond to the minimum and maximum of the data. Color can separate the ray length distribution into quantiles. For example, red and green can split lengths at the median with light and dark hues splitting at the quartiles. Variations using color ordering are reasonably successful at reinforcing a finer length discrimination. The second approach is a variation on the thermometer. Now all the rays are the same length. With color, say yellow, the “mercury" portion of the ray represents the quantity, with the rest of the thermometer closer to the background color. The thermometer approach has the advantage of a common reference scale for cross comparison of distinct rays (Cleveland and McGill 1984a). Without color, the mercury portion can be drawn thicker. However, the additional thickness consumes space and accentuates saturation problems. Stereo methods can also apply to the display of glyphs. Figure 9.5 is an example of stereo pairs plot using ray glyphs to represent 4-dimensional data.

Aggregation is one approach to overplotting and glyphs can be used to represent properties of the aggregate. For example, a histogram is an aggregation across one variable, with one summary statistic (count) displayed by a particularly simple glyph (box with variable

height). Similarly, the box-and-whisker plot is an aggregation across one variable, with five summary statistics displayed with a somewhat more complicated glyph. For saturated scatterplots the data are aggregated or grouped using two variables. Bachi (1978) and Tukey and Tukey (1981) illustrate the technique using a regular grid to group the data. In each cell formed by the regular grid a distributional summary of a third variable is plotted using a glyph. The aggregation approach is not entirely satisfactory.

9.3 Parallel Coordinate Plots:

The classic scatter diagram is a fundamental tool in the construction of a model for data. It allows the eye to detect such structures in data as linear or nonlinear features, clustering, outliers and the like. Unfortunately, scatter diagrams do not generalize readily beyond three dimensions. The principal difficulty, of course, is the fact that while a data vector may be arbitrarily high dimensional, say d, Cartesian scatter plots may only easily be done in two dimensions and, with computer graphics and more effort, in three dimensions. In place of a scheme trying to preserve orthogonality of the d-dimensional coordinate axes, draw the axes as parallel. A vector (x_1, x_2, ..., x_d) is plotted by plotting x_1 on axis 1, x_2 on axis 2 and so on through x_d on axis d. The points plotted in this manner are joined by a broken line. Figure 9.6 illustrates two points (one solid, one dashed) plotted in parallel coordinate representation. In this illustration, the two points agree in the fourth coordinate. The principal advantage of this plotting device is clear. Each vector (x_1, x_2, ..., x_d) is represented in a planar diagram so that each vector component has essentially the same representation. This scheme is due to Inselberg (1985) and to Wegman (1990). The fundamental idea of parallel coordinates is that the transformation from Cartesian coordinates to parallel coordinates is a highly structured mathematical transformation, hence, maps mathematical objects into mathematical objects. Of course, Cartesian coordinate representations have had a long history and consequently much development of intuition about the appearance of structures represented in Cartesian coordinates. Similar intuition for parallel coordinate representations must be developed.

9.3.1 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 9.7. We superimpose a Cartesian coordinate axes t,u on the xy parallel axes so that the y parallel coordinate 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.

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. The ideal points may be thought of as extra point added to the ordinary plane and may be thought of intuitively as the points where parallel lines intersect. There are as many ideal points as there are slopes. Thus, these parallel lines intersect at the ideal point in direction b^{-1}. One model for the projective plane is a hemisphere with diametrically opposed equatorial points identified.

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.

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

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

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

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

describes the projective transformation of points in Cartesian coordinates to lines in parallel coordinates. Because these are nonsingular linear transformations, 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' where x' denotes x transpose, represents the general conic. Clearly then since $t = xB$, B nonsingular, we have $x = tB^{-1}$, so that $tB^{-1}C(B^{-1})'t'$ 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 $ct^2 - b(u + v)^2 = av^2$, a general hyperbolic form.

There is a 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.

9.3.2 Statistical Interpretations:

Since ellipses map into hyperbolas, we can have an easy template for diagnosing uncorrelated data pairs. With a completely uncorrelated data set, we would expect the 2-dimensional scatter diagram to have circular contours. The parallel coordinate plot would approximate a figure with a hyperbolic envelope. As the correlation approaches negative one, the hyperbolic envelope would deepen so that in the limit we would have a pencil of lines, what we like to call the *cross-over effect*. As the correlation approaches positive one, the hyperbolic envelope would widen with fewer and fewer cross-overs so that in the limit we would have parallel lines. Thus correlation structure can easily be diagnosed from the parallel coordinate plot. Griffen (1958) used this as a graphical device for computing the Kendall τ. Griffen, in fact, attributes the graphical device to Holmes (1928) which predates Kendall's discussion. Griffen demonstrates that the computational formula for computing Kendall τ by means of the Holmes' graphical method is

(9.1) $$r = 1 - \frac{4X}{n(n-1)}$$

where X is the number of intersections resulting by connecting the two rankings of each member by lines, one ranking having been put in natural order. While the original formulation was framed in terms of ranks for both x and y axes, it is clear that the number of crossings is invariant to any nondegenerate monotone increasing transformation of either x or y, the ranks being one such transformation. Because of this scale invariance, one would expect rank-based statistics to have an intimate relationship to parallel coordinates.

Linear relationships are comparatively easy to diagnose using parallel coordinates particularly negative linear relationships since the eye seems to quickly note the crossover effect. Moreover, linear relationships exhibited by several sets of adjacent pairs of parallel coordinate axes may be interpreted as several sets of collinearities. Two sets of collinearities, in turn, may be interpreted as points lying in a 2-dimensional plane with d sets of collinearities being interpreted as points lying in a d-dimensional hyperplane. Thus detecting linear structure is important in understanding data structure, particularly if we are interested in fitting multiple linear regression models. A linear rescaling of one or more of the axes is sometimes helpful because it guides the eye to looking for approximately parallel line segments. Of course, nonlinear relationships will not respond to simple linear rescaling. However, by suitable nonlinear transformations, it should be possible to transform to linearity. Knowing the nonlinear transformation which yields linearity in the data, gives us a fundamental model building tool. It should be further noted that clustering is easily diagnosed using the parallel coordinate representation. Separation on any one axis represents a view of the data which allows for detection of clustering. Because of the connectedness of the multidimensional parallel coordinate diagram, it is usually easy to see whether or not this clustering propagates through other dimensions. Our experience indicates that clustering may occur not in any single dimension but in combinations. So far we have developed intuition for pairwise parallel coordinate relationships. The idea however is that we can, so to speak, stack these diagrams and represent all d dimensions simultaneously.

Consider also the appearance of a mode in parallel coordinates. The mode is, intuitively speaking, the location of the most intense concentration of probability. Hence, in a sampling situation it will be the location of the most intense concentration of observations. Since observations are represented by broken line segments, the mode in parallel coordinates will be represented by the most intense bundle of broken line paths in the parallel coordinate diagram. Roughly speaking, we should look for the most intense flow through the diagram. Figure 9.8 represents a parallel coordinate plot.

9.4 Andrews and Related Plots:

An Andrews plot is a multidimensional plotting device that is somewhat related to the parallel coordinate methodology. See Andrews (1972) for a description of Andrews plots as a data analysis tool. There are several conceptual viewpoints that can be described in connection with Andrews plots. First of all think of a data vector ($x_1, \ldots, x_d$) as represented by pairs of the form (1, x_1), $\ldots$, (d, x_d). One way of think of the parallel coordinate plot is as a linear interpolation between these points. The reason for using a linear interpolation is that the transformation from Cartesian space to parallel coordinate space is a projective transformation and, thus, leads to an elegant geometric interpretation of mathematical structure. In particular, we can map Cartesian geometric structures into parallel coordinate geometric structures. However, other general sets of interpolations may be suggested. The earliest one is essentially a Fourier interpolation. That is, plot a multidimensional vector as a trigonometric polynomial expansion with coefficients determined by the weights x_i. Specifically Andrews suggests plotting

$$f_x(\theta) = x_1/\sqrt{2} + x_2 \sin(\theta) + x_3\cos(\theta) + x_4\sin(2\theta) + x_5\cos(2\theta) + \cdots \tag{9.2}$$

Each unique d-dimensional point, (x_1, ..., x_d), gets mapped into a unique trigonometric polynomial. These are then plotted in a way similar to parallel coordinate plots. Two properties of Andrews plots are interesting. First, because of the Fourier series interpretation, the classic Parseval's Theorem holds. Parseval's Theorem basically has to do with L_2-norms and asserts that mean square error in the Fourier domain and mean square error in the untransformed domain are the same. Thus while the untransformed domain is d-dimensional Euclidian space, the Fourier domain is 2-dimensional space so that by looking at an Andrews plot we can visually get an idea of the mean square error structure. The second property relates to the fact that we are talking about orthonormal trigonometric series. Because of this (thinking of the x-axis variable as an angle, say θ), for every θ we get a different linear weighting of the x_is. We can think of a slice at θ as a 1-dimensional projection of the multivariate vector onto an axis whose orientation is determined by θ. This in effect gives us a one-dimensional grand tour as discussed below. As with any grand tour this offers us the possibility of looking for orientations that show up interesting or unusual properties.

There is nothing inherently sacred about either the piecewise linear (parallel coordinate plot) or the trigonometric (Andrews plot) interpolation. The former is useful because in preserves geometric properties, the latter because of the mean square interpretation. The 1-dimensional grand tour would work with any orthonormal series so there may be some other interesting orthonormal series to think about. It may be that we can invent series which highlight different properties so that we can have a family of plots designed to explore different aspects of the structure. That is to say, if we are interested in highlighting clustering or outliers, an appropriate orthogonal series that would exaggerate those aspects of the data in the plot. Thus, we generalize the parallel coordinate and Andrews plots.

One related interesting proposal that we are currently exploring is to do an expansion in two dimensions instead of just one. What we have described before is an expansion $f(\theta; x)$ where $x=(x_1, \ldots, x_d)$ where f is either a piecewise linear interpolant or a trigonometric series. We use a bivariate expansion say $\vec{f}(\theta; x) = (f_1(\theta), f_2(\theta))$ as a 2-dimensional Fourier transform with irrational phase ratio (or, in fact, any orthonormal series). In this situation we should essentially preserve the Parseval-type property and create the two-dimensional grand tour. We can then think of a 3-dimensional plot, plotting $\vec{f}(\theta)$ against θ. Presumably, having a three-dimensional plot helps uncover more structure in the data than a simple two-dimensional plot would. Moreover, by rotating the plot so that the y-z axis is the screen axis. Then slicing this graph along the x-axis would correspond to doing a two-dimensional grand tour. This provides a unified treatment of Andrews/parallel-coordinate-type plots with the grand tour idea.

9.5 Multivariate Density Estimation:

Nonparametric multivariate density estimation is probably the only significant challenge (perhaps unsolvable challenge) left to those who work in the area of probability density estimation. The recent book by Scott (1992) is a superb treatment of and a rather definitive statement of the state of the art with respect to the multivariate density estimation problem. For massive data sets, the examination of data sets by means of scatterplots,

scatterplot matrices, parallel coordinate plots and similar devices are basically flawed by their inability to overcome problems with serious overplotting. Techniques such as scintillation described above can mitigate overplotting for modest sample sizes, but ultimately the only recourse for serious overplotting is the replacement of points (or more precisely their symbolic representation) with their densities.

Multivariate density estimation has historically had two major approaches: a) multivariate kernel estimation and b) binned-type estimates. Both of these approaches have had their detractors. For example Tukey and Tukey (1981, p. 223) argue that "... it is difficult to do well with bins in so few as two dimensions. Clearly, bins are for the birds!" Silverman (1986, p. 80) echoes this perspective and adds further that "... it is virtually impossible to draw a meaningful contour diagram of a bivariate histogram." Kernel-based methods tend to be dismissed with arguments based on the so-called "curse of dimensionality" and calculations of "equivalent sample sizes." For example, Silverman (1986, p. 94) argues that some 842,000 10-dimensional observations are needed to equal in terms of mean square error a mere 4 1-dimensional observations. Scott (1992, p. 83) calculates that 10,634,200 5-dimensional observations are required to yield mean integrated squared errors equivalent to those obtained with 1000 1-dimensional observations. Moreover, kernel estimators are required on a denser grid in higher dimensions. That is, a linear spacing of .1 requires only 11 evaluations of the density estimator (and hence $11 \times$ n evaluations of the kernel, where n is the sample size, i.e. 11,000 evaluations in the Scott example) whereas in the same linear spacing in 5 dimensions would require some 11^5 density evaluations and using Scott's calculations, some $(11 \times 10{,}634{,}200)^5 = 2.19 \times 10^{40}$ evaluations of the kernel. While these reasons a priori tend to discourage the investigator, there is much merit to forging ahead anyway. In our experience, the story for either of these density estimation approaches is not so gloomy as this discussion would seem to indicate.

9.5.1 Multivariate Kernel Density Estimation:

Probably the first principle to remember with the graphical use of multivariate kernel estimation is that we are attempting to understand the structure of the data and not necessarily trying to minimize the mean square error estimate of the true underlying density. Indeed, the data may be sufficiently nonhomogeneous that there is no "true" density and even if there were, the nature of the data may not justify traditional asymptotic theory. Thus while we do not disregard density estimation theory, it is perhaps irrelevant to kernel smoothing for the purposes eliciting of clustering, functional structure, correlations and so on. Relieved of the baggage of optimality considerations, it is clear that the mean square error computations as well as the linear spacing concerns can be set aside. Moreover from the point of view of graphical tools, density estimators are most valuable in two, three, four and five dimensions, since it is exceptionally difficult to meaningfully visualize densities of dimension higher than five.

The multivariate analog to equation (8.4) is

$$\widehat{\mathrm{f}}_n(\mathbf{x}) = \frac{1}{\mathrm{nh}_n^d} \sum_{i=1}^{n} \mathrm{K}(\mathbf{x} - \mathbf{x}_i) \tag{9.3}$$

where $\mathbf{x} = (x_1, \ldots, x_d)^T$ is the evaluation point and the $\mathbf{x_i} = (x_1^i, \ldots, x_d^i)^T$, $i = 1, \ldots, n$ are the d-dimensional observations. Usually the kernel K is a radially symmetric, unimodal probability density function. One choice of kernel is the standard multivariate normal given by $K(\mathbf{x}) = (2\pi)^{-d/2} \exp(-\frac{1}{2}\mathbf{x}^T\mathbf{x})$. Another choice of some interest is the Epanechnikov kernel given by $K(\mathbf{x}) = \frac{1}{2} c_d^{-1}(d+2)(1-\mathbf{x}^T\mathbf{x})$. Here c_d is the (hyper-) volume of a unit d-dimensional hypersphere so that, for example, $c_1 = 2$, $c_2 = \pi$ and $c_3 = 4\pi/3$. Kernels with bounded, finite support are to be preferred provided they have sufficiently high-order continuity properties so that at least second order and preferably third order derivatives exist.

The use of a single bandwidth parameter, h_n, implies that all of the x_j^i for each *j* are scaled equally. If not, the h_n could be made directional dependent or alternatively, the data could be pre-whitened by making a covariance dependent transform. As suggested initially by Fukunaga (1972) and repeated by Silverman (1986),

$$\hat{f}_n(\mathbf{x}) = \frac{(\det S)^{-1/2}}{nh_n^d} \sum_{i=1}^{n} k\left(h_n^{-2}(\mathbf{x}-\mathbf{x_i})^T S^{-1}(\mathbf{x}-\mathbf{x_i})\right) \tag{9.4}$$

where k is given by $k(\mathbf{x}^T\mathbf{x}) = K(\mathbf{x})$. Here S is a (possibly robust) estimator of the covariance matrix.

The choice of bandwidth parameter can be pursued empirically by trial and error or through such techniques as cross validation. By asymptotic considerations of mean square error or mean integrated square error similar to those in the univariate case, an optimal choice of h_n is $c \cdot n^{-1/(d+4)}$. As in the univariate case, the choice of c depends on the unknown density function and so the above expressions serves only to establish the asymptotically optimal rate of decrease for h_n. A recommended strategy for selecting bandwidth is the use of cross validation. Explicit formulae for cross validation for multivariate kernel density estimators are given in both Silverman (1986) and Scott (1992). It is worth pointing out that Monte Carlo experiments using bandwidths known to be optimal demonstrate that the resulting estimators tend to look visually too rough (Wegman, 1972b). Thus there is merit to somewhat oversmoothing particularly for visualization purposes. Figure 9.9 represents a wireframe representation of a bivariate kernel density estimator. A fully rendered color version of the same bivariate density estimator is given in Figure 9.13.

A second observation worth making is that for purposes of graphical rendering of density surfaces (and contours), it is necessary to know surface normals, gradients and tangent planes. Density estimators based on kernel estimators are well suited to this purpose. Three- or four-dimensional kernel density estimators are well within the capability of a relatively modest workstation. Most of the associated graphics computations can proceed without intensive numerical computation once the density is computed. This is particularly the case in graphics-oriented computers having rotation, perspective and lighting computations built into the hardware. The kernel estimators are particularly desirable in these settings since a differential operator passes through the summation sign in (9.3) or (9.4) and so that derivatives and gradients can be computed analytically, i.e. as new kernel smoothers with kernels derived analytically. We shall see more of this in section 9.6.

9.5.2 Multivariate Binned Methods

In spite of the declaration of Tukey and Tukey (1981) cited above, binning methods have proven to be extraordinarily effective in the multivariate setting. Scott (1992) provides a rather extensive discussion of the asymptotic L_2-theory of histograms. In particular he discusses choice of fixed binwidths (which, in common with kernel density estimation, he labels as h), the choice of the number of bins, k and choice of the initial bin boundary, say t_0. A historical choice of number of bins is given by the so-called Sturges Rule, $k = 1 + \log_2 n$. Scott shows that the optimal asymptotic choice is $k = c\ n^{-1/3}$ in which case the MSE is $O(n^{-2/3})$. As a practical alternative to the Sturges Rule, Scott proposes the Normal Bin Width Reference Rule, $k = 3.5\,\widehat{\sigma}\ n^{-1/3}$. He also shows that $\text{var}\{\widehat{f}(x)\} \doteq f(x)/nh$ so that histograms are heteroscedastic and, consequently, can benefit from a variance stabilizing transform such as square root. The resulting function, $\sqrt{\widehat{f}(x)}$, is known as the rootgram. Interestingly enough, because the square root is a monotone transformation, both the multivariate histogram and the multivariate rootgram have the same contours. Scott also discusses asymptotic L_1, L_p and L_∞ theory of histograms.

Our interest here is in multivariate binned density estimators. In this case, we consider observations from $\mathbb{R}^d$ and consider a series of hyper-rectangular bins whose sides are h_1, h_2, $\ldots$, h_d. If bin, B_k, contains ν_k points with $\sum_k \nu_k = n$, then the histogram evaluated at $\mathbf{x}$ is

$$\widehat{f}(\mathbf{x}) = \frac{\nu_k}{n h_1 h_2 \cdots h_d} \text{ for } \mathbf{x} \in B_k. \tag{9.5}$$

In analogy with the univariate case, Scott's Normal Reference Rule becomes $h_k = 3.5\,\sigma_k\ n^{-1/(2+d)}$ where σ_k is the marginal standard deviation of the *k*-th variable. The value of a multivariate histogram is, however, still somewhat limited as the asymptotic convergence rate of mean integrated squared error is $O(n^{-2/(2+d)})$ and the binning structure determined by the h_j is still fairly coarse. The positive side of binned estimators is the computational efficiency due to the floating to integer conversion mentioned earlier.

The favorable asymptotic properties of kernel estimators can be combined with computational properties of binned estimators in the average shifted histogram (ASH). Here the idea is to begin with a a histogram with origin, $\mathbf{t_0} = (t_1^0, t_2^0, \ldots, t_d^0)$ and compute a series of shifted histograms based on origins $\mathbf{t_0} + (\mathbf{h}_{i_1 \ldots i_d}/m)$ where $\mathbf{h}_{i_1 \ldots i_d} = (i_1 h_1, i_2 h_2, \ldots, i_d h_d)$ and $0 \leq i_j \leq m - 1$. The resulting m^d histograms are then averaged to form the average shifted histogram. Scott shows that as $m \to \infty$, the ASH is asymptotically equivalent to a kernel estimator with a triangular kernel. Indeed, he shows that the multivariate kernel is, in fact, the product of d one-dimensional triangular kernels. Of course, as m becomes large the computational advantage of the binned estimator disappears. However, a relatively small amount of shifting is needed to substantially improve the smoothness properties. Scott (1985, 1986) are the original references on the average shifted histogram but Scott (1992) includes the details in one convenient discussion. In any case, it is clear that ASHes based on (hyper-) rectangular or (hyper-) cubic bins can be an effective tool in dimensions two through five at least.

While Scott has advocated (hyper-) rectangular bin, Carr et al. (1987) have advocated a hexagonal bin in the two-dimensional setting. While the asymptotics are essentially equivalent, the hexagonal bins have the advantage that they are “rounder" than squares or rectangles and, hence, have better visualization properties. That is, it is considerably easier and psychologically more appealing to have contours based on hexagons than based on squares. Since two dimensional data is often generated by spatial data, the hexagonal binning is extremely useful for data visualization on maps. Carr, Olsen and White (1992) discuss map applications more extensively. The regular tiling of two-dimensional space can be done by squares, equilateral triangles and hexagons. In a similar way, three-dimensional space can be regularly tiled by octahedrons and, in fact, four-dimensional space can be regularly tiled by a four-dimensional regular figure known as a 24-cell. Thus Carr's hexagonal tiling idea can be extended in a natural way to three and four dimensions. An example of a scatterplot matrix of bivariate densities using hexagonal binning is given in Figure 9.10.

A third type of tessellation of (hyper-) space can be done on a data-adaptive basis not unlike the results of Wegman (1975) for the one-dimensional case. This idea has been explored by Hearne and Wegman (1991, 1992). The idea here is to use an $\alpha \times 100\%$ subsample of the data set. Using the observations so chosen as vertices, one constructs the Delauney triangularization of the selected points. Finally, by constructing the bisecting $(d-1)$-planes of the sides of the triangularized points, one can construct a Voronoi tessellation of d-space which is data dependent having “small" tiles where there is much data and “large" tiles where there is little data. By resampling with additional $\alpha \times 100\%$ subamples, one can construct a bootstrap-type estimator which, like Scott's ASH, has the property of smoothing by averaging density values on flat tiles.

9.5.3 Parallel Coordinate Density Estimates

As we have seen, parallel coordinate plots draw their basic utility from the fact that the mapping from Cartesian space to parallel coordinate space is a projective transformation. The implication of this is that geometric structures in Cartesian space are mapped into geometric structures in parallel coordinate space, in particular, structures which carry statistical interpretations. The drawback of both ordinary scatter diagrams and their multidimensional parallel coordinate analogs is that they begin to lose their effectiveness as the size of the data set becomes large. Replacing the scatterplot by a density estimate allows for visualization of structure where only heavy overplotting would have occurred. Miller and Wegman (1991) propose using a density plot for parallel coordinates in much the same way that we have earlier suggested for ordinary scatterplots. However, for parallel coordinate plots a somewhat different form of density is required. This is true because the limiting asymptotics appropriate for points in an infinitesimal box are inappropriate for lines crossing through an infinitesimal box. The limiting form is called a *line density.*

Although the primary use of parallel coordinate plots is in the display of sample data, it is important to understand the properties of sample density plots in terms of the theoretical form of the line density. By this is meant the density arising when the variables being plotted have a joint probability distribution. Consider the case of a line density plot for the parallel coordinate plot of two variables, x_1 and x_2. The parallel axes will be set at a distance u_0 apart. For ease of exposition, we will superimpose a Cartesian (t, u) coordinate system on the parallel

plot, where the x_1 axis is given by the line $u = 0$ and the x_2 axis is given by the line $u = u_0$. See Figure 9.11. Then the point (x_1, x_2) in the original Cartesian coordinates is represented by the line connecting the points $(x_1, 0)$ and (x_2, u_0) in the parallel coordinate plot. The equation of this line in (t, u) coordinates is easily seen to be $(t - x_1)/(x_2 - x_1) = (u/u_0)$ or $t = (1 - p)x_1 + px_2$, where $p = u/u_0$. This equation shows that the u coordinate is most conveniently described by the proportion $p = u/u_0$; hence, a generic point (t, u) is best represented by recoding $u = pu_0$. This will mean that densities may be described in terms of t and p rather than t and u. The actual value of u_0 used in a parallel coordinate plot is unimportant as far as these densities are concerned. Hereafter we will refer to points in their (t, u) coordinates by (t, p).

By a theoretical form is meant the theoretical form of the density in parallel coordinates appropriate when X_1 and X_2 are random variables with a bivariate distribution. We will restrict our attention to the case where X_1 and X_2 have a continuous bivariate distribution, since discrete X_1 and X_2 yield density plots which simply mimic ordinary parallel coordinate plots. A density for such a distribution is typically defined by some sort of limiting argument. For points in two dimensions (u_1, u_2), the natural method for defining the density at a point (u_1, u_2) is to set up a rectangle with u_1 extending to $u_1 + \delta u_1$ and u_2 to $u_2 + \delta u_2$. The probability mass within this rectangle is then computed and divided by the area of the rectangle, $\delta u_1 \delta u_2$, with the limit taken as δu_1 and δu_2 approach zero. When the random variables, U_1 and U_2 have a continuous bivariate distribution, this argument will yield precisely the probability density function in Cartesian coordinates. One might presume that the proper method to define a line density is to set up the same sort of rectangle, establish the probability for the points in (x_1, x_2) whose lines pass through it, divide by the area of the rectangle, and take limits.

Using rectangles will, in fact, not work because, as detailed in Miller and Wegman (1991) the attempt to define a density using rectangles will always result in a limit of $+\infty$. Although an attempt to define a line density by considering rectangles will not be successful, the correct method can be seen by examining lines passing through the *line segment* from (t, p) to $(t + \delta t, p)$. The limit of the probability contained in this region divided by δt will be the density $g_p(t)$ of $V_p = (1 - p)X_1 + pX_2$ where V_p has a continuous distribution. Then defining the line density in the following manner as $g_p(t) = \lim_{\delta x \to 0} P\{t \leq V_p \leq t + \delta t\}/\delta t$ yields the following result:

Proposition: Let X_1 and X_2 have a bivariate distribution such that $V_p = (1 - p)X_1 + pX_2$ has a continuous distribution for all $0 \leq p \leq 1$. The line density at the point (t, p) of a parallel coordinate plot for X_1 and X_2 is given by $g_p(t)$, where $g_p(\cdot)$ is the density of the random variable V_p. The density so defined is a density over $0 \leq p \leq 1$, $-\infty < t < +\infty$ in the sense that $\int\int f(t, p)dt\,dp = 1$. □

There is a practical motivation for defining a density in terms of lines passing through a line segment rather than lines through a rectangle which derives from counting considerations. Since the line segment from (t, p) to $(t + \delta t, p)$ is parallel to the parallel axes, the line representing a point (x_1, x_2) will cross it in exactly one position. However, a line representing (x_1, x_2) could, in principle, cross several rectangles with the same p but with different values

of t, thus resulting in a form of double counting. Using the definition of density we have chosen avoids any possibility of this form of double counting.

The parallel coordinate density can be conceived as as an infinite collection of one-dimensional densities indexed by *p*. However, as seen by the proposition, the parallel coordinate line density is in fact a true two-dimensional density. The practical calculation of a parallel coordinate line density plot, however, conveniently reduces to a series of one-dimensional densities. From a graphics point of view, we can regard each row of pixels as a line and simply form the univariate sample determine by parallel coordinate line segments passing through that line. Then it is a simple matter to use any reasonable univariate density estimate to form the parallel coordinate line density plot. We have used both kernel estimators and ASH-type estimators to this end. An example of a theoretical parallel coordinate density plot is given in Figures 9.12a and 9.12b. The former is a 3-D perspective plot of a bivariate uniform density and the latter is a contour plot of the same.

9.5.4 Display of Multi-Dimensional Densities

A two-dimensional data set $\mathbf{x_1}, \ldots, \mathbf{x_n}$ where $\mathbf{x_k} = (x_1^k, x_2^k)$ is represented by a three-dimensional density function $f(\mathbf{x})$. The function may either be represented by a two-dimensional surface in three-dimensional space or by a series of contours in the two-dimensional space where the data live. Contours at level α are, in this case, essentially slices of the density surface defined as $S_\alpha = \{\mathbf{x} \in \mathbb{R}^2 \colon f(\mathbf{x}) = \alpha f_{max}\}$ where $f_{max} = \sup_{\mathbf{x}\in\mathbb{R}^2} f(\mathbf{x})$. (These density contours are also known as *isopleths* and were studied in a formal sense by Sager (1979) who develops estimators of isopleths and properties of these statistical estimators.) In a similar way, three-dimensional data is represented by a three-dimensional hypersurface in four dimensional space and, of course, four-dimensional data by a four-dimensional hypersurface in five-dimensional space. The contours or isopleths for three-dimensional data live in three-dimensional space so that realistically there is some hope of graphically understanding densities for three-dimensional data from there contours. Indeed, isopleths corresponding to four-dimensional data would live in $\mathbb{R}^4$ which would not be directly visible since they would be true four-dimensional structures. However, by treating one of the four variables as a time index, we can view the four-dimensional contours as a time indexed series of three-dimensional contours which we could view as an animation using the graphics tools about to be described. Thus there is at least some reasonable expectation of directly visually understanding five-dimensional density estimates based on four-dimensional data.

Let us for the moment focus on two-dimensional data or more precisely, the rendering of mathematical functions of two variables. It is by now reasonable conventional to render these as wireframe models typically viewed from an oblique position. An example is shown in Figure 9.9. Wireframe surfaces are visually appealing and simple to produce with monochrome hardcopy devices. However, fine structure located in interstitial areas is impossible to detect. Luo and Wegman (1992) have experimented extensively with rendering, three-dimensional stereoscopic and transparency effects for visualization of such two dimensional functions. By rendering, they mean attributing surface characteristics and lighting effects to the mathematical function. Typically they have use metallic specular surfaces for the obverse side of a surface and purely diffuse surfaces for the reverse. These different surface characteristics allow them to distinguish the sides of a surface when dealing with complex

surfaces. Scott (1992) also has explored such approaches extensively and suggests color coding the surfaces, red towards the mode, for example, and blue away from the mode. Luo and Wegman have found that the surface texture coding allows them to reserve color coding more readily for other uses such as coding of contour levels or red-green stereoscopic coding. The combination of metallic surface effects and lighting (using Phong lighting and Gouraud shading effects) creates surfaces which readily show fine structure as one might expect to see by viewing an imperfect mirrored surface. Of course, these images are dramatic and attractive in their own right. See Figure 9.13 for an illustration of a two-dimensional density surface using the rendered metallic surface and lighting effects discussed here. Figure 9.14 is an illustration of three-dimensional contours of a four-dimensional density surface using similar lighting, rendering and surface representation techniques. The data represented in Figure 9.14 is distribution of matter in the universe.

In addition to the surface texture and lighting effects for mathematical surfaces, Luo and Wegman introduced stereoscopic effects. Initial efforts involved using red-green stereo and, surprisingly, they have reported the remarkable ability to use continuous-tone rendered images with red-green stereo. The human visual system is able to converge and fuse red-green images even when no sharp visual cues are available, such as might be found in red-green stereoscopic versions of wireframe models. This seems to be the case even when the surfaces are lit with non-red or non-green highlighting. Indeed, in some ways rendered red-green stereo seems to be even more effective since the absence of sharp visual cues as might be found with line drawings seems to minimize the annoying effects of optical crosstalk due to imperfect filters.

A third component to the visualization of mathematical functions has been the introduction of transparency. By allowing carefully adjusted layers of transparency coupled with stereoscopic views, Luo and Wegman have indicated the ability to visualize exceedingly complex surfaces. Indeed, they have examples of convoluted functions in which as many as eight surface layers may be distinguished. Coupling transparency, stereoscopic displays and rendering effects appears to allow the human system to effectively visualize exceptionally complex mathematical structures. Of course, while they have applied these tools to visualization of density estimates, there is no a priori reason that these tools in conjunction could not be applied to visualization of other complex scientific structures.

It is worth noting that red-green stereoscopy is readily and cheaply available on even PC platforms. A pair of colored stereoscopic glasses can be purchased for well under one dollar. More expensive stereoscopic systems are available. In particular, liquid crystal polarizing systems are available from a number of vendors, systems which either fit filters over a high resolution monitor or which are built into active shutter glasses. Luo and Wegman have used the former system to create full color, transparent, rendered, stereoscopic images of mathematical functions.

9.6 Isopleths, Ridges, and Skeletons:

The previous section began with a discussion of visualizing density surfaces of two-dimensional densities, i.e. two-dimensional surfaces embedded in three space. Under mouse control to allow dynamic rotations, scaling and translations, with all of the visualization tools described, it is possible to thoroughly understand such two-dimensional densities. Density

contours associated with three-dimensional data may be examined in much the same way since isopleths of densities of three-dimensional data are also two-dimensional surfaces embedded in three space. The computations are similar except for the fact that somewhat more computation must be done in order to compute the kernel density on a grid of equally fine linear dimensions. Also for two dimensional density surfaces, the surface normal is orthogonal to the tangent surfaces (hence, to the gradient) while for two-dimensional density isopleths, the gradient and the surface normal coincide. However, except for these minor details, the visualization is essentially the same. One additional feature can be accommodated, namely *dynamic thresholding*. By dynamic thresholding is meant the adjustment of the slicing level, α, associated with the isopleths. This is done by Luo and Wegman under mouse control and allows for examination of the density isopleths from near the mode to the tails of then density. Scott (1992) also reports using nested contour levels, e.g. $\alpha = .25$, .50, and .75, together with transparency as a tool for understanding density structure. As indicated above, density contours for true four-dimensional data can be accommodated by animating on one of the variables and viewing a sequence of density contours for the remaining three variables.

Contouring is a subtle problem, particularly in three dimensions. The grid upon which the density is computed must be sufficiently fine so that the density surface can be well approximated and that the fine structure is reliably reproduced. For two dimensions, assuming a square grid, one possible algorithm is to examine the sign of the value of $f(\mathbf{x}) - \alpha f_{max}$ on each of the four corners of the grid elements (or, indeed, if there are other polygons used a similar technique may be employed). If all the vertices have the same sign, then the contour does not pass through that element. If one or two vertices have different signs from the remainder, then the contour passes through this square grid and using linear interpolation, an approximating line segment can be calculated. If the grid is sufficiently fine, a close approximation to the contour can be made.

For three-dimensional surface contours, Scott suggests a technique that involves fixing one of the three variables and computing a series of two-dimensional contours at several equally spaced values of the fixed variable. This method gives the appearance of a series of slices through the contour surface and allows one to see interior structure easily. It is well suited to viewing nested contours.

A three-dimensional variant of the two-dimensional algorithm can be constructed. Consider a cubic grid and evaluate $g(\mathbf{x}) = f(\mathbf{x}) - \alpha f_{max}$ at each of the eight vertices. If $g(\mathbf{x})$ for one of the vertices has a different sign from the remaining vertices, the linear interpolant will be a triangle. Similarly if either two or four of the signs are different from the remainder, then the linear interpolant is a rectangle. If three of the signs are different from the remainder, the linear interpolant is a pentagon. Finally, if exactly three of the vertices have $g(\mathbf{x}) = 0$, then the interpolant is a regular hexagon. In any case, the resultant interpolants can be combine to approximate the surface contour. This latter algorithm is called the *marching cubes* and was discussed by Lorensen and Cline (1987).

A thoughtful examination of the density surface allows us to construct a generalization of regression. Consider the intuitive notion of a ridge as we might think of a mountain ridge. The *ridge* is in some sense to be made more precise a maximal one-dimensional feature on the two-dimensional surface of a density. Figure 9.15 illustrates the “ridge" and the contours of density for a collection of two-dimensional data. One simple fact to observe is that if one

considers a point on the ridge and calculates directional derivatives at that point on the ridge, the directions of derivatives with greatest magnitude will be orthogonal to the ridge. The ridge as conceived by Luo and Wegman (1992) is a feature lying on the surface of the density. Its support, i.e. the closure of $\{\mathbf{x}: f(\mathbf{x}) > 0\}$, is defined to be the *skeleton*.

The utility of the skeleton is as a geometrically derived summary statistic. In a classical bivariate normal density function with $\rho \neq 0$, the ridge will lie exactly over the major axis of the elliptical contours and skeleton will coincide with the major axis. However, because of its geometric inspiration there is no particular need for the skeleton to be linear and, in fact, there is no particular need for the skeleton to be a proper non-intersecting Euclidian manifold. Thus it is a highly flexible non-linear, nonparametric generalization of the classic regression curve. Indeed, Luo and Wegman (1992) show that it is a generalization of a conditional mode just as classical regression is a generalization of conditional expectation.

The ridge and skeleton notions can be generalized to arbitrary dimensions. Consider a vector $\mathbf{x}$ in d-dimensional space with probability density function $f(\mathbf{x})$. Let R be a compact k-dimensional, $k < d$, smooth manifold on the density surface with supporting manifold S. The *likelihood function of* R *on* S is defined as the k-dimensional integral of $f(\mathbf{x})$ on S which is the $(k+1)$-dimensional hypervolume under R. S and R are said to be *consistent with respect to the mode* if each point $\mathbf{x}$ on S is the conditional mode of the density conditioned on the $(d-k)$-normal plane of S at $\mathbf{x}$. The *k-ridge of a d-dimensional density* is the k-dimensional smooth consistent manifold R that maximizes the likelihood defined as the hypervolume under the manifold. The *k-skeleton* is the support of the k-ridge. Two simple observations are of interest. First, the 0-skeleton coincides with the mode. The 0-ridge is just the value of the density at the mode. Second, the ridges and skeletons are nested. That is, the k-ridge is contained in the $(k+1)$-ridge and, similarly, the k-skeleton is contained in the $(k+1)$-skeleton. This hierarchical arrangement is informative in determining the higher dimensional structure of the data.

There are two simple algorithms for finding the k-ridge known respectively as *orthogonal slicing* and *gradient tracing.* The former algorithm, primarily for 1-ridges, begins by choosing the mode. A step size, say h, is chosen and the the density in the neighborhood of the mode, say m_1, is searched at a radius of h for a local maximum, a point approximately on the ridge. These two points are joined by a chord which approximates the ridge, the orthogonal bisector found and the mode, say m_2, of the local density estimate formed by using points in a say δ-neighborhood of the bisector is found. This algorithm is then repeated with m_2 replacing m_1. Of course, the neighborhood of m_1 is rejected as a candidate for the new point m_3. This process is repeated until there is a saddle point or no local maximum. Saddle points may exist in the density and the process may have to be repeated with other local modes. Parameters h and δ may be adjusted in order to assure some reasonable approximation to the ridge.

A perhaps more satisfactory algorithm involves gradient tracing. As Luo and Wegman have implemented it, the density gradient is calculated at each observation. The trace at a particular observation, say $\mathbf{x}_j = \mathbf{x}_j^0$, is computed by taking a step length proportional to the magnitude of the gradient in the direction of the gradient vector. Call the endpoint of that step $\mathbf{x}_j^1$. This is a steepest ascent type algorithm. See Figure 9.15. The gradient is recomputed at

$\mathbf{x}_j^1$ and a second step is taken with the step length again proportional to the gradient at $\mathbf{x}_j^1$ in the direction of the gradient vector to compute $\mathbf{x}_j^2$. This process is repeated for general i to obtain $\mathbf{x}_j^i$. As i $\rightarrow \infty$, $\mathbf{x}_j^i$ approaches the local mode, but, in fact, through the nested sequence of k-ridges. This view of the density is essentially the density as a potential surface with trajectories, basins of attractors and separatrix. See Figure 9.16. The separatrix essentially corresponds to the k-ridge. The nested character is a most interesting feature. Essentially the vector direction of the gradient will point to the direction of the highest dimensional ridge, say k-ridge. Once the trajectory is traced to the k-ridge, the trajectory will sharply turn and the gradient vector will point in the direction of the next highest dimensional ridge, in general the (k $-$ 1)-ridge, but not necessarily. This process repeats until the 0-ridge is reached. The number of turns in the trajectory is a diagnostic indicator of the dimensionality of the highest dimensional summary statistic. It may turn out, for example, that the highest dimensional ridge for three-dimensional data is one-dimensional. Moreover by *shaving the trajectories* from the observations to the modes, i.e. by sequentially eliminating the links from $\mathbf{x}_j^{i-1}$ to $\mathbf{x}_j^i$ starting at $i = 1$, one obtains a nested sequence of k-ridge estimators. For at least 0- 1- and 2-dimensional ridges these can be visualized by techniques describe above. In addition, contours of three-dimensional ridges can also be visualized.

We note as indicated in our discussion of multivariate kernel density estimation, kernel estimators have the advantage of that surface normals and gradients can be computed directly with kernels rather than numerically as finite differences. The implication for gradient tracing is important. For straight density estimation, the amount of computation increases exponentially with dimension for a fixed size linear grid. However, the gradient tracing still requires a number of density computations proportional to the sample size no matter the dimension. Thus it may not be feasible to do a density estimate in ten dimensions, but it will still be feasible to do the gradient trace.

Finally, one interesting observation is that with a univariate normal density, the points of inflection occur at $\mu \pm \sigma$. If one computes and plots the magnitude of the gradient, the mode of the density will become a minimum and the points of inflection will become local modes (i.e. 0-ridges). In the same way, if one computes the magnitude of the gradient of a two-dimensional density, the 1-ridge becomes a local minimum and the flanking “lines-of-inflection" become 1-ridges. In general the k-ridges of the function defined as the magnitude of the gradient of the density can be interpreted as confidence bands for the k-ridge of the density. Thus this method not only gives a completely nonparametric, geometrical generalization of regression, but also yields a method for geometrically generating confidence bands for the generalized, nonparametric nonlinear regression estimator.

9.7 Generalized Rotations and the Grand Tour:

The grand tour of a multidimensional data set is a visualization technique for examining structure of high dimensional data using dynamic graphics. The idea, introduced in Asimov (1985) and Buja and Asimov (1985), is to capture, in some sense, the popular meaning of grand tour, that is, to look at a subject from all possible angles. The grand tour notion as a data analytic tool has also been studied by Buja, Hurley and McDonald (1986) and by Hurley and Buja (1990). In a data analytic setting, all of these authors propose to project a

d-dimensional data set into a dense set of the possible two-dimensional planes. If the sequence of projections is smooth, the visual impression is that the data points move in a continuous fashion from frame to frame of a movie. The object of the data analyst is to look for *unusual* configurations of the data, configurations which may reflect some structure. The grand tour, in this sense, shares a common objective with the projection pursuit techniques. The added motion associated with the grand tour literally adds another dimension, the time dimension, which many data analysts find very helpful. Finally, we note that Wegman (1991) discusses a form of the grand tour in general k-dimensional space.

Asimov (1985) describes several implementations of the grand tour. Consider the space of two-planes (two-flats) in a d-dimensional space. The collection of two-planes can be fully described by the two orthonormal basis vectors which form the coordinate axes for the two-plane. We can consider these two basis vectors as being subject to a general rotation in d-space such that they always remain orthogonal. The observations making up the data set are projected onto the plane spanned by the basis vectors and, then, the two axes are mapped into the x-y screen axes of the computer's monitor. The net effect is that while the data space axes are undergoing general motion in d-space, the appearance on the computer's monitor is that the data points are moving continuously within fixed screen coordinates.

The Grand Tour Algorithm in d-Space:

Let $\mathbf{e}_j = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$ be the canonical basis vector of length d. The 1 is in the j^{th} position. The $\mathbf{e}_j$ are the unit vectors for each coordinate axis in the initial position. We want to do a general rigid rotation of these axes to a new position with basis vectors $\mathbf{a}_j(t) = (a_1^j(t), a_2^j(t), \ldots, a_d^j(t))$, where, of course, t is a time index. The strategy then is to take the inner product of each data point, say $\mathbf{x}_i$, $i = 1,\ldots,n$ with the basis vectors, $\mathbf{a}_j(t)$. By convention, d will refer to the dimension and n will refer to the sample size. Of course, the j subscript on $\mathbf{a}_j(t)$ means that $\mathbf{a}_j(t)$ is the image under the generalized rotation of the canonical basis vector $\mathbf{e}_j$. Thus the data vector $\mathbf{x}_i$ is $(x_1^i, x_2^i, \ldots, x_d^i)$, so that the representation of $\mathbf{x}_i$ in the $\mathbf{a}_j$ coordinate system is

$$\mathbf{y}_i(t) = (y_1^i(t), y_2^i(t), \ldots, y_d^i(t)), \; i = 1, \ldots, n \tag{9.6}$$

where

$$y_j^i(t) = \sum_{k=1}^{d} x_k^i a_k^j(t), \tag{9.7}$$

$j = 1, \ldots, d$ and $i = 1, \ldots, n$. The vector $\mathbf{y}_i(t)$ is then the linear combination of basis vectors representing the i^{th} data point in the rotated coordinate system at time t.

The goal thus is to find a generalized rotation, Q, such that $Q(\mathbf{e}_j) = \mathbf{a}_j$. We can think of $\mathbf{Q}$ as either a function or as a matrix $\mathbf{Q}$ where $\mathbf{e}_j \times \mathbf{Q} = \mathbf{a}_j$. We implement this by choosing $\mathbf{Q}$ as an element of the special orthogonal group denoted by SO(d) of orthogonal $d \times d$ matrices having determinant of $+1$. In order to find a continuous, space-filling path through the Grassmannian manifold of d-flats, we must find a continuous, space-filling path through the SO(d). As we have just demonstrated with (9.6) and (9.7), once we have the basis

vectors for the rotated coordinate system, the d-flat is determined and the computation of the data vector in the rotated system is straightforward. It is not obvious how to find a continuous, space-filling path through SO(d), however, so we will need to take the algorithm one step further back.

In general d-dimensional space, there are $d - 2$ axes orthogonal to each two-flat. Thus rather than rotating around an axis as we are used to in ordinary 3-dimensional space, we must rotate in a plane in d-dimensional space. The generalized rotation matrix, **Q**, is built up from a series of Givens rotations in individual two-flats. In d-space, there are d canonical basis vectors and, thus, $\binom{d}{2} = \frac{1}{2}(d^2 - d)$ distinct two-flats formed by the canonical basis vectors. We let $R_{ij}(\theta)$ be the element of SO(d) which rotates in the $e_i e_j$ plane through an angle of θ. We define **Q** by

$$\mathbf{Q}(\theta_{1,2}, \theta_{1,3}, \ldots, \theta_{d-1,d}) \tag{9.8}$$

$$= \mathbf{R_{12}}(\theta_{1,2}) \times \cdots \times \mathbf{R_{d-1,d}}(\theta_{d-1,d}).$$

There are $p = \frac{1}{2}(d^2 - d)$ factors in expression (9.8). The restrictions on θ_{ij} are $0 \leq \theta_{ij} \leq 2\pi$, $1 \leq i < j \leq d$. The vector $(\theta_{1,2}, \theta_{1,3}, \ldots, \theta_{d-1,d})$ can thus be thought of as a point on a p-dimensional torus. This is the origin of the description of this method as *the torus method*. The individual factors $R_{ij}(\theta)$ are $d \times d$ matrices given by

$$\begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \cos(\theta) & \cdots & -\sin(\theta) & \cdots & 0 \\ \vdots & & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \sin(\theta) & \cdots & \cos(\theta) & \cdots & 0 \\ \vdots & & \vdots & & \vdots & & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \tag{9.9}$$

where the cosine and sine entries are in the i^{th} and j^{th} columns and rows. See, for example, Foley et al. (1990, p. 205).

The final step in the algorithm is to describe a space filling path on the p-dimensional torus, T^p. This can be done by a mapping α:$\mathbb{R} \rightarrow T^p$ given below by

$$\alpha(t) = (\lambda_1 t, \lambda_2 t, \ldots, \lambda_p t) \tag{9.10}$$

where $\lambda_1, \ldots, \lambda_p$ is a sequence of linearly independent real numbers and the λ_it are interpreted modulo 2π. The fact that the λ are linearly independent guarantees that the mapping α will describe a space-filling curve on the torus. The composition of α with **Q** will describe a space filling path in SO(d). Thus our final algorithm is given by

$$\mathbf{a}_j(t) = \mathbf{e}_j \times \mathbf{Q}(\lambda_1 t, \ldots, \lambda_p t). \tag{9.11}$$

Equations (9.7), (9.8), (9.9) and (9.11) taken together are the explicit formulation for computing the transformed variable $\mathbf{y}_i(t)$, $i = 1, \ldots, n$.

10. Concluding Remarks

This paper has attempted to cover an enormous range of literature and, in that coverage, to synthesize elements of traditional statistical graphical techniques, computer graphics and the rapidly emerging field of scientific visualization. While these have grown up in somewhat separate literatures, they, in our view, are aspects of the same enterprise. It is obvious that an encyclopedic synthesis of all of these elements would require literally thousands of pages of text and illustrations. We have contributed much less than a thousand pages in this work, but in so doing, we hope to have brought together enough material to describe a vision of how these elements might be integrated.

Further reading and detailed references might be added along several directions. First in the area of **computer graphics**, in addition to the general works of Foley et al. (1990) and Plastock and Kalley (1986), the work of Watt (1989) specifically treats 3-D computer graphics and the work of Glassner (1989) is focussed on ray tracing. Durrett (1987) discusses color theory with special focus on color in computer technology. With respect to **statistical graphics techniques**, in addition to the works by Chambers et al. (1983) and Cleveland (1985), the recent book by Buja and Tukey (1991) is a collection of papers by the leaders in the area of statistical graphics and statistical computing. Cressie (1991) addresses the important issues of the analysis of spatial data including graphical analysis. Graphical representation and analysis of spatial data is an enormously difficult area. We have not discussed this area nor have we discussed the connection with the important area of geographic information systems (GIS). GIS is one of the spin-off technologies from the revolution in computers and computer graphics and undoubtedly will be closely linked to spatial data representation via graphics. Finally we would like to reiterate our enthusiasm for the new book by Scott (1992) which focuses on the strategically important role of densities in addition to scatterplots as data analysis tools particularly in settings with large data sets.

Geometry clearly plays a central role in computer and statistical graphics. The classical treatment of multidimensional geometry by Kendall (1961) is both a pleasure to read and a mind stretching experience in counter-intuitive mathematics. More recently, the work of Taylor (1992) specifically deals with geometry relevant to computer graphics, while that of Weeks (1985) is a fun treatment of the geometry of three-dimensional manifolds. Other books treating multidimensional geometry in a more popular vein include Banchoff (1990), Heiserman (1983) and Rucker (1984). Fractal geometry is another topic we have not touched in the present discussion. However, scatterplots with structure, particularly multidimensional

scatterplots, have fractional or fractal dimension. The work of Barnsley (1988) is one of the accessible mathematical treatments of this area.

If geometry plays a pivotal role in graphics and visualization, an equally pivotal role is played by the area of **vision theory**. We have already mentioned the work of Julesz (1971). A somewhat different perspective is presented in the work of the late David Maar, Maar (1982). **Visualization** has not yet received wide attention in the book format, but in addition to the work of Friedhof and Benzon (1989) already referenced in this paper, the edited work of Nielson et al. (1990) provides a focused collection of papers in the area of scientific visualization. Finally, we mention the recent work of Robbin (1992) which focuses on the interplay of art, visualization and computer graphics, and 4-dimensional geometry.

Acknowledgement:

The authors would like to thank our student, Qiang Luo, for assistance in preparation of the graphic images in this paper as well as for thoughtful contributions to the sections on the display of multidimensional data and density contours, ridges and skeletons whose theory forms part of his Ph.D. dissertation. Qiang Luo is also the principal programmer of Mason Ridge^{TM} and ExplorN^{TM}.

References

Andersen, E. (1957), “A semigraphical method for analysis of complex problems," ***Proceedings of the National Academy of Science***, 13, 923-927. Reprinted in ***Technometrics***, 2, 387-391.

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. Sci. Statist. Comput.***, 6, 128-143.

Bachi, R. (1978), “Graphical statistical methodology in the automation era," Graphical Presentation of Statistical Information: Presented at the 136th Annual Meeting of the American Statistical Association, Soc. Stat. Sess. ***Graphical Methods of Statistical Data***, Boston, MA, 1976, Technical Publication 43, 13-68.

Banchoff, T. (1990), ***Beyond the Third Dimension***, New York: W. H. Freeman and Company.

Barnsley, M. (1988), ***Fractals Everywhere***, Boston: Academic Press, Inc.

Becker, R. A, and Cleveland, W. S. (1987), “Brushing scatterplots," ***Technometrics***, 29, 127-142.

Becker, R. A. and Cleveland, W. S. (1991), “Viewing multivariate scattered data," ***Pixel***, July/August, 36-41.

Becker, R. A. and Cleveland, W. S. (1991), “Take a broader view of scientific visualization," ***Pixel***, July/August, 42-44.

Bertin, J. (1983), ***Semiology of Graphics - Diagrams, Networks, Maps***, (Berg, W. J., translator), Madison, WI: The University of Wisconsin Press.

Bruce, V. and Green, P. R. (1990), ***Visual Perception, Physiology, Psychology, and Ecology***, 2nd Edition, Hillsdale, NJ: Lawrence Erlbaum Associates.

Bui-Tuong, Phong (1975), “Illumination for computer generated pictures," ***Commun. Assoc. Comp. Mach.***, 18, 311-317.

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

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***, 171-174, (T. Boardman, ed.), Alexandria, VA: American Statistical Association.

Buja, A. and Tukey, P. A. (1991), ***Computing and Graphics in Statistics***, New York: Springer-Verlag.

Carr, D. B. and Littlefield, R. J. (1983), “Color anaglyph stereo scatterplots - construction details," ***Computer Science and Statistics: Proceedings of the 15th Symposium on the Interface***, 295-299, New York: North Holland Publishing Company.

Carr, D. B., Littlefield, R. J., Nicholson, W. L. and Littlefield, J. S. (1987), “Scatterplot matrix techniques for large N," ***J. American Statistical Association***, 82, 424-436.

Carr, D. B. and Nicholson, W. L. (1984), “Graphical interaction tools for multiple 2- and 3-dimensional scatterplots," in ***Computer Graphics '84: Proceedings of the 5th Annual Conference and Exposition of the National Computer Graphics Association***, 2, 748-752.

Carr, D. B. and Nicholson, W. L. (1989), “Explor4: a program for exploring four-dimensional data using stereo-ray glyphs, dimensional constraints, rotation, and masking," ***Dynamic Graphics for Statistics***, (Cleveland, W. S. and McGill, M. E., eds.) 309-329, Monterey, CA: Wadsworth and Brooks/Cole Advanced Books and Software.

Carr, D. B., Nicholson, W. L., Littlefield, R. J. and Hall, D. L. (1986), “Interactive color display methods for multivariate data," in ***Statistical Image Processing and Graphics***, (Wegman, E. J. and DePriest, D. J. eds.), 215-250, New York: Marcel-Dekker, Inc.

Carr, D. B., Olsen, A. R. and White, D. (1992), “Hexagon mosaic maps for display of univariate and bivariate geographical data," to appear ***Cartography and Geographic Information Systems***, 19.

Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983), ***Graphical Methods for Data Analysis***, Monterey, CA: Wadsworth Advanced Books and Software.

Chambers, J. M. and Hastie, T. J. (1992), ***Statistical Models in S***, Pacific Grove, CA: Wadsworth and Brooks/Cole Advanced Books and Software.

Chernoff, H. (1973), “Using faces to represent points in k-dimensional space," ***J. American Statistical Association***, 68, 361-368.

Cleveland, W. S. (1985), ***The Elements of Graphing Data***, Monterey, CA: Wadsworth Advanced Books and Software.

Cleveland, W. S. and McGill, R. (1984a), “Graphical perception: theory, experimentation and application to the development of graphical methods," ***J. American Statistical Association***, 79, 531-554.

Cleveland, W. S. and McGill, R. (1984b), “The many faces of a scatterplot," ***J. American Statistical Association***, 79, 807-822.

Cressie, N. (1991), ***Statistics for Spatial Data***, New York: John Wiley and Sons.

Daniel, C. and Wood, F. S. (1980), ***Fitting Equations to Data***, Second Edition, New York: John Wiley and Sons, Inc.

Dent, B. D. (1990), ***Cartography: Thematic Map Design***, Dubuque, IA: Wm. C. Brown Publishers.

Dondis, D. A. (1973), ***A Primer in Visual Literacy***, Cambridge, MA: The MIT Press.
Durrett, H. J. (1987), ***Color and the Computer***, San Diego: Academic Press.

Elderton, W. P. and Johnson, N. L. (1969), ***Systems of Frequency Curves***, Cambridge, England: Cambridge University Press.

Emerson, J. D. and Soto, M. (1983), “Transforming data," ***Understanding Robust and Exploratory Data Analysis***, (Hoaglin, D. F., Mosteller, D. F. and Tukey, J., ed.), 97-128, New York: John Wiley and Sons, Inc.

Feynman, R. P. (1985), ***QED: The Strange Theory of Light and Matter***, Princeton, NJ: Princeton University Press.

Fienberg, S. (1979), “Graphical methods in statistics," ***American Statistician***, 33, 165-178.

Foley, J. D., van Dam, A., Feiner, S. K. and Hughes, J. F. (1990), ***Computer Graphics Principles and Practice***, Second Edition, Reading, MA: Addison-Wesley Publishing Company.

Friedhof, R. M. and Benzon, W. (1989), ***The Second Computer Revolution: Visualization***, New York: W. H. Freeman and Company.

Friedman, J. H., McDonald, J. A. and Stuetzle, W. (1982), "An introduction to real-time graphical techniques for analyzing multivariate data," in ***Proceedings of the 3rd Annual Conference and Exposition of the National Computer Graphics Association***, 1, 421-427.

Fukunaga, K. (1972), ***Introduction to Statistical Pattern Recognition***, New York: Academic Press.

Glassner, A. S. (1989), ***An Introduction to Ray Tracing***, London: Academic Press.

Griffen, H. D. (1958), "Graphic computation of tau as a coefficient of disarray," ***J. American Statistical Association***, 53, 441-447.

Hearne, L. B. and Wegman, E. J. (1991), "Adaptive probability density estimation in lower dimensions using random tessellations," ***Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface***, 241-245.

Hearne, L. B. and Wegman, E. J. (1992), "Maximum entropy density estimation using random tessellations," to appear ***Computing Science and Statistics: Proceedings of the 24th Symposium on the Interface***.

Heiserman, D. L. (1983), ***Experiments in Four Dimensions***, Blue Ridge Summit, PA: TAB Books, Inc.

Hoaglin, D. F., Mosteller, D. F. and Tukey, J. (1983), ***Understanding Robust and Exploratory Data Analysis***, New York: John Wiley and Sons, Inc.

Hoaglin, D. F., Mosteller, D. F. and Tukey, J., eds. (1985), ***Exploring Data Tables, Trends and Shapes***, New York: John Wiley and Sons, Inc.

Hodges, L. F., (1992), "Tutorial: Time-multiplexed stereoscopic computer graphics displays," ***IEEE Computer Graphics and Applications***, 11, 20-30.

Holmes, S. D. (1928), "Appendix B: a graphical method for estimating R for small groups," 391-394, in ***Educational Psychology*** (Peter Sandiford, auth.), New York: Longmans, Green and Co.

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.

Izenman, A. J. (1991), “Recent developments in nonparametric density estimation," ***J. American Statistical Association***, 86, 205-224.

Johnson, N. L. and Kotz, S. (1969), ***Distributions in Statistics: Discrete Distributions***, New York: John Wiley and Sons.

Johnson, N. L. and Kotz, S. (1970), ***Distributions in Statistics: Continuous Univariate Distributions-***I, New York: John Wiley and Sons.

Johnson, N. L. and Kotz, S. (1970), ***Distributions in Statistics: Continuous Univariate Distributions-***II, New York: John Wiley and Sons.

Johnson, N. L. and Kotz, S. (1972), ***Distributions in Statistics: Continuous Multivariate Distributions***, New York: John Wiley and Sons.

Julesz, B. (1986), “Texton gradients: the texton theory revisited," ***Biological Cybernetics***, 54, 245-251.

Julesz, B. (1971), ***Foundations of Cyclopean Perception***, Chicago: University of Chicago Press.

Kendall, M. G. (1961), ***A Course in the Geometry of n-Dimensions,*** London: Charles Griffen and Company, Ltd.

Kronmal, R. A. and Tarter, M. E. (1968), “The estimation of probability densities and cumulatives by Fourier series methods," ***J. American Statistical Association***, 63, 925-952.

Land, E. (1977), “The retinex theory of color vision," ***Scientific American***, 237, 108-128.

Lipton, L. (1982), ***Foundations of the Stereo-Scopic Cinema, A Study in Depth***, New York: Van Nostrand Reinhold.

Littlefield, R. J. (1984), “Basic geometric algorithms for graphic input," in ***Computer Graphics '84: Proceedings of the 5th Annual Conference and Exposition of the National Computer Graphics Association***, 2, 767-776.

Lorensen, W. E. and Cline, H. E. (1987), “Marching cubes: a high resolution 3D surface construction algorithm," ***Computer Graphics***, 21, 163-169.

Luo, Q. and Wegman, E. J. (1992), “Visualization of multivariate structure," invited talk, Computing Science and Statistics: 24th Symposium on the Interface, College Station, TX, March, 1992. Embedded in software for Silicon Graphics workstations, ***Mason Ridge***^{TM},

available through Professional Statisticians Forum, Ltd., P. O. Box 7176, Fairfax Station, VA 22039.

Maar, D. (1982), ***Vision***, New York: W. H. Freeman and Company.

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

Morse, A. (1979), "Some principles for the effective display of data," ***Computer Graphics***, 13, 94-101.

Newman, W. M. and Sproull, R. F. (1979), ***Principles of Interactive Computer Graphics***, Second Edition, New York: McGraw-Hill.

Nielson, G. M., Shriver, B. and Rosenblum, L. J. (1990), ***Visualization in Scientific Computing***, Los Alamitos, CA: IEEE Computer Society Press.

Plastock, R. A. and Kalley, G. (1986), ***Computer Graphics,*** New York: McGraw-Hill Book Company.

Prakasa Rao, B. L. S. (1983), ***Nonparametric Functional Estimation***, Orlando: Academic Press.

Robbin, T. (1992), ***Fourfield: Computers, Art and the 4th Dimension***, Boston: Bullfinch Press.

Robinson, A., Sale, R. and Morrison, J. (1978), ***Elements of Cartography***, Fourth Edition, New York: John Wiley and Sons, Inc.

Rosenblatt, M. (1956), "Remarks on some nonparametric estimates of a density function," ***Annals of Mathematical Statistics***, 27, 832-837.

Rucker, R. (1984), ***The Fourth Dimension: Toward a Geometry of Higher Reality***, Boston: Houghton-Mifflin Company.

Sager, T. W. (1979), "An iterative method for estimating a multivariate mode and isopleths," ***J. American Statistical Association***, 74, 329-339.

Schwartz, S. C. (1967), "Estimation of a probability density by an orthogonal series," ***Annals of Mathematical Statistics***, 38, 1262-1265.

Scott, D. W. (1985), "Average shifted histograms: effective nonparametric density estimators in several dimensions," ***Annals of Statistics,*** 13, 1024-1040.

Scott, D. W. (1986), “Data analysis in three and four dimensions with nonparametric density estimation," in ***Statistical Image Processing and Graphics***, (Wegman, E. J. and DePriest, D. J. eds.), 291-305, New York: Marcel Dekker, Inc.

Scott, D. W. (1992), ***Multivariate Density Estimation: Theory, Practice and Visualization***, New York: John Wiley and Sons.

Shepard, R. N. and Carrol, J. D. (1966), “Parametric representation of nonlinear data structure," ***Multivariate Analysis***, (Krishnaiah, P. R., ed.), 561-592, New York: Academic Press, Inc.

Silverman, B. W. (1986), ***Density Estimation for Statistics and Data Analysis***, London: Chapman and Hall.

Tapia, R. A. and Thompson, J. R. (1978), ***Nonparametric Probability Density Estimation***, Baltimore: The Johns Hopkins University Press.

Taylor, W. F. (1992), ***The Geometry of Computer Graphics***, Monterey, CA: Wadsworth and Brooks and Cole Advanced Books and Software.

Tukey, J. W. (1977), ***Exploratory Data Analysis***, Reading, MA: Addison-Wesley.

Tukey, P. A. and Tukey, J. W. (1981), “Graphical display of data sets in three or more dimensions," Chapters 10, 11 and 12 in ***Interpreting Multivariate Data***, (Barnett, V. ed.), Chichester, UK: J. Wiley and Sons, Ltd.

Tufte, E. (1983), ***The Visual Display of Quantitative Information***, Cheshire, CT: Graphics Press.

Tufte, E. (1990), ***Envisioning Information***, Cheshire, CT: Graphics Press.

Valyus, N. A. (1962), ***Stereoscopy***, New York: The Focal Press.

Watt, A. (1989), ***Fundamentals of Three-Dimensional Computer Graphics***, Reading, MA: Addison-Wesley Publishing Company.

Weeks, J. R. (1985), ***The Shape of Space: How to Visualize Surfaces and Three-Dimensional Manifolds***, New York: Marcel Dekker, Inc.

Wegman, E. J. (1972a), “Nonparametric probability density estimation: I. A summary of available methods," ***Technometrics***, 14, 533-546.

Wegman, E. J. (1972b), “Nonparametric probability density estimation: II. A comparison of density estimation methods," ***J. Statistical Computation and Simulation***, 1, 225-245.

Wegman, E. J. (1974), “Computer graphics in undergraduate statistics," ***International J. Math. Educat. Science Techno.***, 5, 15-23.

Wegman, E. J. (1975), “Maximum likelihood estimation of a probability density," ***Sankhya*** (***A***), 37, 211-224.

Wegman, E. J. (1990), “Hyperdimensional data analysis using parallel coordinates," ***J. 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. (1992), “Wavelets and nonparametric function estimation," Technical Report 79, Center for Computational Statistics, George Mason University, Fairfax, VA 22030.

Wegman, E. J. and DePriest, D. J. (1986), ***Statistical Image Processing and Graphics***, New York: Marcel Dekker, Inc.

Wegman, E. J. and Wright, I. W. (1983), “Splines in statistics," ***J. American Statistical Association***, 78, 351-365.

Yeh, Y. Y. and Silverstein, L. D. (1990), “Limits of fusion and depth judgment in stereoscopic color displays," ***Human Factors***, 32, 45-60.

Zeki, S. (1992), “The visual image in mind and body," ***Scientific American***, Sept., 69-76.

Legends for Figures

Figure 2.1 Translation transformation in two dimensions.

Figure 2.2 Geometric rotation in two dimensions.

Figure 2.3 Scaling transformation in two dimensions with s_x and s_y both equal to two.

Figure 2.4 Composite viewing transformation composed of $V = D \times N$.

Figure 3.1 Alignment of vector V with z-axis by successive rotations.

Figure 3.2 Perspective projection with eye located at C.

Figure 5.1 Diffuse reflection dependent on angle between surface normal and viewing direction.

Figure 5.2 With diffuse reflection, a fixed light energy is incident over a larger area when striking an oblique surface.

Figure 5.3 Specular reflection generates a highlight centered around the reflection and dropping off as $\vec{V}$ departs from $\vec{R}$.

Figure 5.4 Backward ray tracing with hypothetical light ray emanating from the viewer's eye, through the viewing window, to the scene.

Figure 6.1a Three-dimensional stereoscopic visualization illustrating calculation of y'.

Figure 6.1b Geometry for computation of stereoscopic displacement of x for the left eye. Displacement is sum of a and b. Right eye computation is similar.

Figure 6.2 Large angular parallax, p', limits ability to fuse images.

Title

Figure 21. Melanoma of the Skin - Age Specific Rates.
The rates are 1970s rates for the U. S. white
population. Note the difference between sexes.

Legend

Figure 7.1 Terminology. The figure defines terminology. Two curves are superimposed.

Figure 21. Melanoma of the Skin - Age Specific Rates.
The rates are 1970s rates for the U. S. white
population. Note the difference between sexes.

Figure 7.2 Terminology. This figure also defines terminology. Separate panels showing common scale curves have been juxtaposed.

Figure 7.3 A graphic pitfall illustrating two superimposed curves whose apparent closeness on the right-hand side of the chart is an optical illusion. The second panel shows the algebraic distance plotted as a function of x.

Figure 7.4 Section of eye showing the optical system: 1. Cornea, 2. Anterior Chamber, 3. Iris, 4. Posterior Chamber, 5. Crystalline Lens, 6. Gonule of Zinn, 7. Chorloide Plexus, 8. Fovea Centralis, 9. Retina, 10. Sclera, 11. Optic Nerve.

Figure 7.5 The focal length, f, of a simple lens. The lines at the left are parallel. The power of the lens is given by $p = 1/f$.

Figure 8.1 Standard EDA viewplot containing 1) histogram, 2) box plot, 3) kernel density plot and 4) normal q-q plot. This and subsequent examples are based on the National Health and Nutrition Survey (NHANES 1 study of diet and cancer) data. Persons reported here included 3355 men who remained alive and cancer free for at least four years. Albumin measured in grams per liter.

Figure 8.2 Sequence of box plots against age groups. Albumin for 3355 men. Age groups 35-44 and 45-54 have similar values. Otherwise albumin decreases with age.

Figure 9.1 Stereo pairs plot of a simple three-dimensional scatterplot. Some individuals can fuse such images by allowing eyes to cross. Most require a stereopticon-type viewer.

Figure 9.2 Simple scatterplot matrix for four Nhanes variables: 1) Albumin, 2) TIBC (total iron binding capacity), 3) transferin saturation and 4) hemoglobin. Notice banding in the albumin scatterplots indicating quantization of this variable.

Figure 9.3 Chernoff faces as icons representing multivariate data.

Figure 9.4 Ray glyph plot indicating nitrate deposition from acid rain in eastern U. S. and Canada from 1982 to 1987.

Figure 9.5 Stereo scatterplot incorporating ray glyphs as a representation of the fourth variable.

Figure 9.6 Parallel coordinate representation of two d-dimensional points.

Figure 9.7 Cartesian and parallel coordinate plots of two points. The *tu* Cartesian coordinate system is superimposed on the *xy* parallel coordinate system.

Figure 9.8 Parallel coordinate display of Nhanes data. Heavy overplotting limits utility of this plot although the quantization of albumin is very evident. A parallel coordinate density plot for 3355 observations is more appropriate.

Figure 9.9a Wireframe plot of a two-dimensional kernel density estimator. The density is based on observations representing random scatter from an artificially constructed c-curve as illustrated in Figure 9.9b.

Figure 9.9b Artificially generated c-curve with random scatter.

Figure 9.10 A hexagonal binned scatterplot matrix for the Nhanes data. Compare with Figure 9.2 to observe how density plots often show structure better than simple scatterplots.

Figure 9.11 Construction for the computation of the parallel coordinate line density plot. We may take $u_0 = 1$ so that $u = p$.

Figure 9.12a Three-dimensional perspective plot of the parallel coordinate line density associated with a pair of independent uniformly distributed random variables.

Figure 9.12b Contours (isopleths) of the parallel coordinate line density associated with a pair of independent uniformly distributed random variables.

Figure 9.13 Rendered color version of the two-dimensional density estimator in Figure 9.9a. Surface texture is given a metallic characteristic. Gouraud shading and Phong lighting have been applied.

Figure 9.14 Rendered color version of the contours (isopleths) of a kernel smoother of three-dimensional data. The data smoother here are the locations of super-galactic clusters. Measurements are declination, right ascension and redshift. This represents an estimate of the distribution of matter in the universe on the largest scale.

Figure 9.15 Gradient tracing of ridges of the density estimator given in Figures 9.9 and 9.13. This method allows a smooth reconstruction of the original c-curve given in Figure 9.9b.

Figure 9.16 Density contours and gradient vectors arising from treating the kernel density estimator in Figure 9.9a as a potential surface. Note how the 1-ridge and the modes (0-ridges) are attractors.