File Glyphs
By Daniel B. Carr
Version November 27, 2010, Revised Nov. 19, 2011, and Nov. 23, 2012.

Purpose Discuss glyph encoding for variables and selected viewing options
Provide examples to illustrate
Scaling for difference purposes: glyph production, rotation
Encoding options and seriation uses
Selected glyphs and alternative views
Generate and view hypersurfaces

Sections

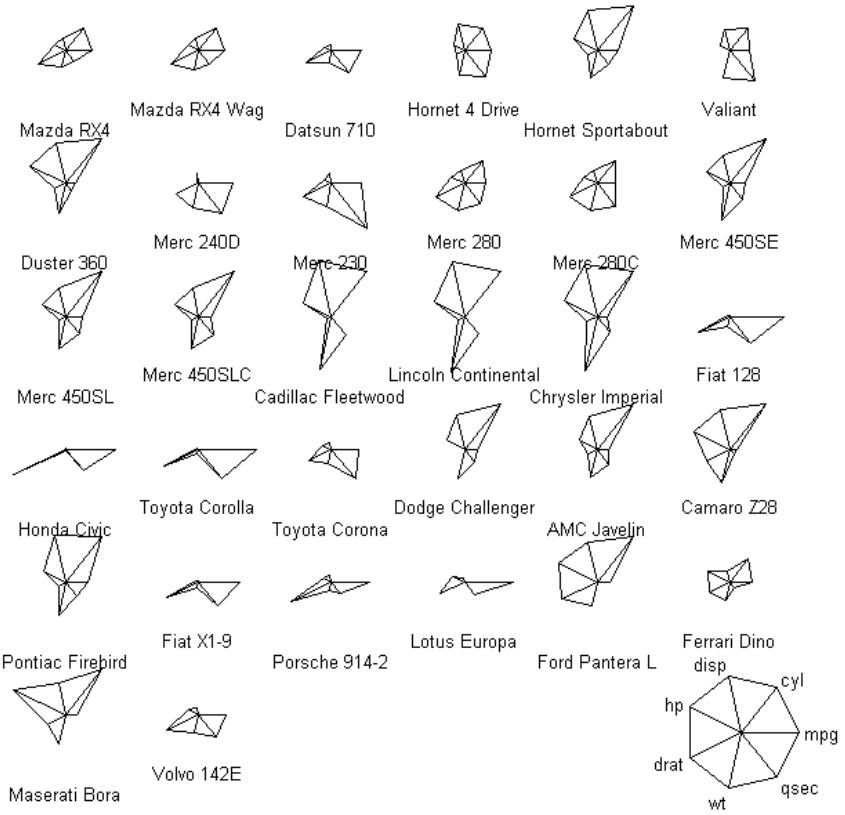
Due
2 One plot: your choice
3 Plot
4 Second plot
8 One plot: your choice
9 Second plot
10 Second plot
11 A beginning or ending still shot
12 Plot

1. Univariate and multivariate glyphs

In this class a **glyph** is graphical representation than encodes variable values for individual cases. This is a less restrictive definition than Anderson's [1960] usage in which there line for each variable and the length of each line is proportional to the variable's value.

Commonly tried glyphs include Chernoff face, star glyphs, profile glyphs that are illustrated in the assignment below. The figure below shows an example of star glyphs representing seven variables each for 32 types of cars. The legend at the lower right provide variable and shows the glyph shape if a car had the maximum value for all seven variable. Each line length represents the variable's magnitude relative to the maximum minus minimum value across the cases.
Historically glyphs were typically shown in a row and column layout, the number case was small and the number of variables small. Efforts to arrange the glyphs to help people see patterns were in frequent. Many people are consumer's and don't think about graph redesign. In this class we could sort the stars by their area put them in a matrix by varying the row index fasted. We could use the seriation method from a previous assignment to order both cases and variables.

Run Sun's [1998] dissertation went further and used rounder cells of hexagon grid as place to put round glyphs. Her approach placed glyphs in 2-D neighboring cells when their distances were relatively small as base on a distance matrix that would be used clustering. She also showed my hexagon cluster graph that separates different clusters of glyphs with empty cells. Finally she develop two sophisticated glyphs that redesign star glyphs. Star glyphs show almost nothing when variables values are near the minima and a long line when values are near the maximum. One of her designs shows a circle about the center dot. The circle represent means and line segments going inside or outside the circle show deviations from the mean.



Of course glyphs have numerous limitations, but before we render a final judgment it seems wise to evaluate graphics after considering redesigns based on contemporary knowledge about human perception and cognition and the use of interactive and dynamic methods that greatly extend our capabilities.

1.1 Glyphs with position-along-a-scale encodings

Over the decades my opinion has been that the most effect multivariate glyphs are those that use the best perceptual accuracy of extraction encodings. By this I mean that position-along-a-scale encodings for continuous variables. I consider a point plot as a glyph plot. The scatterplot is my preferred glyph plot for showing two continuous variables.

For three continuous variables I prefer to study 3-D scatterplots where depth is conveyed using motion or stereo parallax. Slow rotation lets me see different surfaces as the perspective changes and provides time to think about what I am seeing. Stereo keeps the 3-D image in view when I choose to stop the rotation and study or interact with the structure. While occlusion is useful as a depth cue, occlusion can hide much of the 3-D structure. This is the primary legitimate complaint about 3-D scatterplots. Redesigns can address some of the occlusion issues.

Carr et al. [1986] provided color anaglyph stereo line drawing of a 3-D density contour. When thinking from a data density perspective, this indicates one way to address occlusion. In the same book David Scott [1986] described non-parameter density estimates that use line drawing to address both 3-D contours and sequences of line drawing to address 4-D density contours. Scott's [1994] graphics included 2-D perspective and lighting model rendered views of translucent 3-D contour surfaces as well as line drawings. His visually appealing and effective approach should show densities are now more wide accessible using tools such as R and SAS. This doesn't necessarily mean all the desirable interactive and dynamic density exploration methods are available in the visualization environments.

Carr and Nicholson [1987] addressed the display of 4 continuous variables using stereo ray glyphs. For the 4^{th} continuous coordinate, they considered the two good encodings, length and angle, because the best choices were already used. They chose the angle encoding because this seemed to be less ambiguous when glyphs overplotted. An assignment example will show rotating ray glyphs so the motion parallax provide depths cue.

The 1986 Explor4 implementation provided dynamic 4-D density masking and hyperplane slicing to help address the occlusion problem. Density masking hides the selected points in order to reveal density patterns. Slicing and can help reveal internal structure such as holes. As indicated by Furnas and Buja[1994], slicing can be view as lower dimensionality. When the dimensionality of the data structure is smaller than dimensional of the graphics we have a change to see the structure shown. A side by side stereo ray glyph plots in my discussion that accompanied is their publication showed that the 3-D structure of six coordinate data.

The masking idea mentioned above goes back at least to Prim 9. See Fishkeller et al. [1975]. Masking is similar to filtering in that masked points are not plotted. However in my view masking is different than filtering which is a data selection step that occurs before the data is scaled to produce graphics. Thus dynamic filtering can change the graph scales. This can be bad wanting stable context, but may be good extreme values removal improved resolution.

In terms the higher perceptual-accuracy-of-extraction glyphs, I encountered one success story in consulting visit to Merck. I heard they tried faces and several other glyphs for their 6-D data and did not get the insight they sought until tried using stereo ellipsoids. The reference provided was Bayly et al. [1993]. I looked this up in the GMU library long ago. As I vaguely remember, this showed side by side stereo ellipsoids that included a color encoding. One success story is scant evidence. Still, tools are not readily available, few have tied such glyphs, and color publication has been barrier.

My first exposure to ray glyphs was likely from Tukey and Tukey [1981]. These chapters are still worth reading. My contributions include the stereo ray glyphs for 4-D graphics mentioned above, bivariate ray glyphs for maps, angular box plots for maps, and nested scaleed resolution enhance ray used in a nested 5-D plot. This show troughs of local minimum for a 4-D domain,

and show that local minimum in place near the edge of the simulation domain which was centered on what was thought be the global minima.

While things go downhill as we go past 3-D graphics, Furnas and Buja indicate the combinations of project and slicing (prosection view) can help us find structure up to and including six variable graphs.

1.2 Glyph popularization and names

A popular glyph that builds on a scatterplot for representing two coordinates is a circle glyph. This encodes a third coordinate circle area. In recent years the circle has been called a bubble and the plot called a bubble plot. In my world bubbles are three-dimensional. Popularization of such graphics may be helpful from a quantitative literacy perspective, but I worry that names chosen reflect a lack of attention to details in other facets of glyph use. We know area under-represent the ratio of two values and the circle color can be really good showing three classes, that knowledge is not propagated along with the methods.

In my irritation I thought that perhaps that next we could have new and improved bubble with a continuous color gradient from the inside to out or perhaps a little spectral sheen to look like soap bubbles. Of course a further step "forward" would use a volume encoding. The bubble collisions and separations in animation could get interesting. Thinking of solving the minimal surface equations pressed together bubbles. This might be very pleasing to watch, but I am really more inserted see the data. While it lacks the visual impact and elegance of a circle, a ray glyph could be used and support showing somewhat large data sets. Of course direct 3-D plots might be used. Perhaps 3-D TV will make some in road into flatland turf.

I sometimes consider time series lines as glyphs. In fairly recently times such glyphs have been called "spark lines." Like the bubble mentioned above I also assume sparks have a 3-D structure that is being conceptual squashed into flatland. Perhaps sparks are not as wildly varying as lightning. I have seen many sparks and produced a fair number in winter play with static electricity. However, I don't clearly remember the shapes. I would like to see some 3-D pictures of sparks. I suspect the analogy of a time series line is poor both in terms of 3-D structure and very short temporally visibility. The label "time series glyph" could suffice for communication proposes, even though it is uses three words.

Showing small time series is a reasonable idea. Local high acuity vision supports viewing small glyphs. I am a bit trouble about issues of decoding and comparison accuracy when spark lines don't have grid lines so help us judge shifts and slopes. Knowing which lines are which may be issues when the spark lines aren't labeled. When there are grid lines, it is easier to note the close relationship to profile, star, and parallel coordinate glyphs.

1.3 A note on transfer functions

As an implementation note, we often use transfer functions to convert data units in glyph drawing units such as the segment lengths in start glyphs. The following linear transformation is commonly used to encode a continuous variable with positive and negative values.

Glyph unit <- glyphMin + glyphRange * (x – min(x))/ (max(x)-min(x))

In STAT 763 we may take a look at the widgets in GLISTEN that control transfer functions. Rik Littlefield, a computer science collaborator on the ALDS (Analysis of Large Data Sets) project at PNNL, developed an instructive a GUI (graphic user interface) for defining and dynamically manipulating a transfer function. An example described in Carr et al. [1986] was above dynamic bivariate density contour plot.

A pre-interaction step computed the bivariate density estimates based on points in a scatterplot. The estimates were produced at screen pixel coordinates and stored in the respective graphics bit planes.

The graphically defined transfer function from data density to pixel gray level was initially a flat line in a 2-plot that set densities to black. Clicking on four points made an interval so the pixels with density values in that interval would appear white. The left two points went from the black side (bottom of the plot) to the white side (top of the transfer function plot). The right two points went from the white side back to the black side. Binding the four rectangle-defining points together and then dragging them produced a dynamic view showing a changing density contours. This approach exploited the fast color table capabilities that quickly reinterpreted the bit values as pixels colors. Compared to today's graphics cards this was the dark ages. Still the GUI renaissance is still happening.

1.4 Schema for representing distributions and other terms

Wilkinson [2005] suggests using the term "schema" for point and line graphical elements encoding distribution descriptions. He specifically has Tukey's box plots in mind. Carr et al. [1998] use an angular box plot schema used in a mapping context. (The paper is in the class folder.) This is one context where we might have considered using the term glyph, but there an established name.

There are other terms that may refer to glyphs or schema. Murrel [2006] describes graphics production using "grobs" which are graphical objects. Grobs can be used as glyphs. Wickham [2009] uses "geoms" to refer geometric objects. These are often used as glyphs or schema. For example a geom could be a smoothing line on a scatterplot. I like the idea of general purpose geometric graphical objects.

I will endeavor to use term glyph with is represent one for more variables or descriptors for a case. If a scatterplot point locations each show two values and the shape of the state, I call this a glyph. For nation data the symbol could be flag. These are symbols are surrogates for the state or nation names and the case name is variable. Gower and Digby [1981] would likely include all the glyphs described here under the label of a picture or pictorial display.

In more complex situations there might be merit to having labels other than "glyph" or "schema. " I am thinking about a wide variety of scenarios in which each case is associated with one or more distributions of multivariate data.

Consider Level II ecoregions. Each constituent sub-region of an ecoregion can be associated with multivariate values. The collection of sub-region multivariate values along with their size or important weights is whole multivariate distribution of values. Maybe the term multivariate schema might be used or may it should just be called a plot.

Consider a satellite imagery context. Sub-regions many correspond to pixels from satellite imagery and each pixel can have radiance values for many wavelength intervals. (The sub-regions may be satellite footprints that are not really the squares as implied by the pixels.)

Wavelength interval radiances and other variables can be processed to produce a wide variety of estimates. These may be land cover class estimates shown in paper by Carr et al. [1998]. These may be atmospheric geophysical parameters such as temperature, cloud fraction and water vapor values at different altitudes as indicated in the class data exploration lecture entitled: Visualizing cluster –compressed multivariable and multi-altitude atmospheric.

In more general terms exploratory analysis can be pursued at different scales of measurement and aggregation. There are many issues that arise in this context such as the modifiable area unit problem. Simpson's paradox appears in the context of comparing aggregate patterns to conditioned patterns.

The development of good overviews can provide a level of understanding that help guide more detailed exploration. For example in the world of genetics and proteomics the discovery of the double-helix structure of our DNA has provided an overview that helped crystallize previously amorphous thinking. This connects to the powerful (but not the only) perspective that structure drives function. Transcription factor proteins interact with DNA to regulate genes production of RNA and subsequent production of proteins. This interaction is based on shapes that can be altered by different chemical environments.

From genes to the atmosphere and from quarks to quasars, the world of data is so amazing! There is so much to explore and some much to learn!

1.5 Glyph uses and gestalt impressions

In statistical graphics, the hope was that glyphs could serve a variety of multivariate data description tasks. Common graphics tasks include with showing data density patterns (such clusters, holes, paths, and outliers), conveying variable relationships that are of potential use in building models (mental and empirical) and displaying model related case values and in a model criticism context.

Since face recognition is an extremely important in our lives and strongly support by our visual process there was hope the face glyphs would be particularly helpful encoding. This has led to many variants of face glyphs some of which are amazing realistic. There was hope this could provide gestalt impressions as well as provide access to detail. The gestalt part works but we hard time comparing the parts. This is more like an integral encoding and than separable encodings such as color and shape.

There are issues in interpreting the gestalt. While we will respond to facial expressions showing a wide range of emotions the can is still ambiguity even in the context of extreme emotions. There are tears of joy.

1.6 Univariate summaries: problems and comments

Suppose gestalt impressions were one dimensional. Are we capable of coming up with a single satisfactory summary variable to drive the impression? The previous assignment on seriation should increase our doubts. The following observational and questionable comments extend the contexts in which we consider problems.

Long ago the consumer price index (CPI) emerged as a compromise index that could be used to adjust salaries for inflation and other purposes. The CPI is now heavily ingrained in the US culture. This is very rare partial success story. One dimension ordering can be very helpful in human communication and in the making of comparisons. At the same time the CPI has its flaws and is the subject of political battles.

Water quality reporting is still handled one variable at time. There is a standard for a large number of toxic substances. When a substance exceeds the allowed threshold in a sample, I heard that in more than half of such samples had one or more toxic substances exceed their thresholds. Thresholds are not provide and often available for toxic substance interactions.

Efforts have made to boil down measures of hazardous airborne chemical and particulates into single index of air quality. This has not been successful. We do have our code red days here in Fairfax County. I suspect the criteria is narrow. I don't think this relates benzene exposure and at best indirectly to small diesel particulate exposure that can lead to childhood asthma.

An index of danger related to terrorist activity is communicated to the public via color codes. My guess is that code red is bad news but what does it mean and how bad is it? Is it based only of the assessment of event likelihood or does it indicate a risk that combines measure of conjectured loss with its likelihood. Is the loss related property, human life or both? What do code red mean in terms of actions we should take. Should we stay at home? Should we leave town?

The work to produce univariate summary of a multivariate relationships continues. With continuous multivariate data as input this could allow us to produce a univariate dot plot. Such

a plot can be partitioned in to class intervals that can be associated color codes or perhaps a few familiar words such as low, medium and high.

One part of the problem in the risk assessment context is the assessment loss in unified framework. Economists use monetary framework that associates dollars with the loss of pigs, oranges, streets, buildings and human life time. I don't have simple solution so probably should not complain. However my irritation with short term economic thinking motivated the following thought. If aliens wanted to buy all the earth's oxygen and take it with them, wouldn't the economically rational decision be to sell the oxygen and invest the money for future gain.

Maybe a different approach could be to provide some public education and retain a little more in terms of bivariate or multivariate context. Perhaps we could show scatterplot of assessed loss and event likelihoods. Points could be plotted to show previous day daily values and current short term daily predictions. Of course there are issues with this approach.

Direct seriation plot can be using in some context such as archeology, but can fail badly to convey a bigger multivariate picture. My use of seriation seeks to address better layouts for multivariate data and is just one option in tool kit.

1.7 Recap of glyph problems and approaches in cognitive setting

1) Glyphs take up substantial area. The available physical representation area provides a constraint so glyphs don't scale well to even modest data sets. We typically want views in a single page or window. In the large data sets context, glyphs may be useful for as prototypes or exemplar that represent very modest number of clusters or subsets defined by other means.

2) Ware [2008] indicates that we can only keep very simple glyphs in working memory. This undermines the use of complex glyphs that encoding more three variables. The suggest here is to use the position along-a-scale-encoding for the first three variables. This corresponds to using the where pathways of the brain. Then up to three additional variables can be encode using shape and color encodings that are processed using the what pathways of the brain.

3) Leland Wilkinson suggests that face glyphs can provide a gestalt impression. This is potentially useful. At the same time this gestalt formation may complicate tuning our vision to focus on individual variable encodings. If we conceive glyph gestalts as having as one dimensional ordering, the natural thought is to turn to a seriation methods in order to drive the encoding and/or layout of glyph. All the seriation method limitations for representing multivariate methods should come to mind.

4) The layout of glyphs many make a big difference in terms of what we compare, our ease and accuracy of comparison and our observation of the patterns. Ware says "The rapid characterization of a scene is called getting its the *gist*." Our brains are amazingly fast getting the gist of common scenes. Ware says indicates this process is learned and that the learning

starts when we are babies, and continues over our lifetimes. Further he says, “specialists such as race car drivers, visual designers, chefs, dance critics and those who fly fish continue to develop and refine specialized pattern identification skills.

So in the exploratory data analysis setting, how can we encode information and learn to see patterns so that we get the gist and then are in a better a position to look deeper?

Our two general tasks are to look for density patterns and functional relationships. The discussion below provides a bare beginning of a discussion on represent data density in 3-D and 4-D. In terms of functional relationships, the 4-D hypersurface example shown below parallel coordinate and scatterplot matrix plots that show structure. This is in large part revealed by the regularity of domain edges as shown in the collections of 2-D orthographic projections. The rotating ray glyph plot shows much more of the structure. It can be hard to describe the structure but we see that there is structure. In mathematics terms the structure is the abstract functional relations of form f(x1, x2, x3, x4) = 0.

Since noise can hide structure we need tools to help us see the patterns thru the noise. The tools include smoothing and other models.

5) The last problem to address is to having the mathematical/statistical/geometric constructs and additional words to describe the patterns that we see. Both learning and practice are important. Many Ph.D. students are poorly prepared to see and describe the patterns in the tables and plots they put in their dissertations. There are many redesign opportunities.

2. Chernoff faces

Chernoff faces are among the glyphs that tend to draw attention on first exposure and small waves of interest appear over time. The importance of faces in our daily lives is likely behind this interest. The fact that a large part of our brain is devoted to face recognition provides hope that human faces would provide a particularly powerful encoding.

Do faces work very well? So far I don’t think so. Available face encoding now include some that are much more realistic than the examples below. It is still not clear to me that most of the face representations get around the cognitive problems of glyph encodings mention earlier. This is my opinion and not based on a review of scholarly research addresses human performance on specific tasks.

Leland Wilkinson [2005] says faces provide a gestalt impression and shows human faces varying on a sleepiness-activation dimension and an unpleasantness-pleasantness dimension. This is pretty easy to see. While my hopes of using faces effective have dimmed, such examples keep them alive.

Considering gestalt formation raises a question. Does gestalt formation negatively impact our ability to tune our vision to focus on a single variable that is encoded as a feature (such as color) that can otherwise be readily found with preattentive vision?

```
install.packages("aplpack")
library(aplpack)

label <- row.names(mtcars)
n <- nchar(label)
label12 =substring(label,1,12)

windows(w=6,h=6)
faces(mtcars[, 1:7], labels=label12,face.type=0,main = "Motor Trend Cars")

windows(w=6,h=6)
faces(mtcars[, 1:7], labels=label12,face.type=1,main = "Motor Trend Cars")
```

Explanation of parameters:
1-height of face, 2-width of face, 3-shape of face, 4-height of mouth,
5-width of mouth , 6-curve of smile, 7-height of eyes, 8-width of eyes,
9-height of hair, 10-width of hair, 11-styling of hair, 12-height of nose,
13-width of nose, 14-width of ears, 15-height of ears.

3. Star glyphs

The case names and variable legends help to provide context in the examples below. We still do not know about things such as the year, but the example is more about showing capability than about a serious study of the data.

In terms of the layout, the top row of stars makes it evident that the names are below the stars. However the top label appears closer to the stars below. Natural perceptual grouping based on proximity leads to the wrong linkage of label to glyph. An alternative is to draw gray lines in the background and link label glyph together via containment.

The number of cases, 32, is modest. The number of variables shown, 7, is not very high. The actually dimension of the data may be lower than 7 dimensions. There are likely car design constraints that lower dimensionality.

```
# full circle
windows(w=6, h=6.5)
stars(mtcars[, 1:7], key.loc = c(14, 1.5),
```

```
    main = "Motor Trend Cars", flip.labels=FALSE)

# half circle
windows(w=6,h=6.5)
stars(mtcars[, 1:7], key.loc = c(14, 2),
    main = "Motor Trend Cars", full = FALSE)
```

Note that close correspondence between parallel coordinate plots if we were to overplot all the stars. The basic differences are the star plot axes have a common origin, are shown different angles, and the points on the first and last axes are connected.

4. Sector Glyphs

Color fill provides a sense of value added. We can use color to quickly search to compare the values of a specific variable when there are just a few easily distinguished colors. There is a interesting result for colors represent as points CIE diagram (see Ware's book Information Visualization). A color point outside the convex hull constructed from the other vertices ar distinguishable in preattentive vision.

Given that we have located the parts of two glyphs to compare we still have to make comparison and can anticipate that comparison accuracy will decline as distance between the glyphs increases.

How do we compare sectors? I suspect we respond more to sector area than to sector edge length. This may not be the case because length is readily evident. Note that sector area is not a linear function of edge length. My guess is that the perceptual accuracy of extraction is reduced compared to a length encoding.

Another question about the glyph concerns gestalt formation. How difficult is it to make overall comparison before we focus on details? I suspect that comparison is pretty difficult. Working memory only holds about four items, so it may be very difficult to remember more than four sectors in making comparisons. At least color draws attention. Step one in promoted graphics is to get people to look.

```
windows(w=6, h=6.5)
palette(rainbow(12, s = 0.6, v = 0.75))
stars(mtcars[, 1:7], len = 0.8, key.loc = c(12, 1.5), main = "Motor Trend Cars", draw.segments =
TRUE)
palette("default")
```

5. Thin bars and profile glyphs

I think glyphs composed of thin bars with a connecting base line (like a comb on its back) would provide more accurate decoding for individual variables. This is the framework for profile and castle glyphs.

We can exploit Weber's law and add grid lines to help assess the heights of individual variables more accurately. A better version further below adds labels.

```
thinBar <- function(mat, xSpacing, nr, nc, ylab=NULL,
        ylabLoc=0,ybot=.2,ytop=1.0,mycolors,grid="#C0C0C0")
{
  n <- nrow(mat)
  nvar <- ncol(mat)
  if(missing(xSpacing))xSpacing <- seq(.15,.85,length=7)
  if(missing(nr))nr <- floor(sqrt(n))
  if(missing(nc))nc <- ceiling(n/nr)
  if(missing(mycolors))mycolors <- rainbow(7)
  matRange <- apply(mat,2,range)
  matScaled01 <- scale(mat,matRange[1,],diff(matRange))

  px <- as.vector(rbind(xSpacing, xSpacing, rep(NA, nvar)))
  ymin <- rep(ybot, nvar)
  yna <- rep(NA, nvar)
  ydif <- ytop-ybot
  plot(c(0,nc), c(1,nr), type='n', axes=FALSE)

  for(i in 1:n){
    # row wise
    x <- (i-1)%%nc
    y <- nr - (i-1)%/%nc
    ymax <- ymin+matScaled01[i,]*ydif
    rect(x+xSpacing[1],y+ybot,x+xSpacing[nvar],y+ytop,
       density=0,col=grid)
    yMidLines  <- y+ybot+c(.25,.5,.75)*ydif
    segments(rep(x+xSpacing[1],3),yMidLines,
          rep(x+xSpacing[nvar],3),yMidLines,col=grid)
    for(j in 1:nvar)
       lines(x+rep(xSpacing[j],2),y+c(ybot,ymax[j]),
            col=mycolors[j],lwd=3,lend='butt')
    lines(x+xSpacing[c(1,nvar)],rep(y+ybot,2))
    }
}

windows()
mycolors <- rainbow(7)
```

```
mycolors[5] <- mycolors[4]
mycolors[4] <- "#000000"
thinBar(mtcars[,1:7],mycolors=mycolors)
```

The profile glyphs connect the top of the lines. Castle glyphs basically use wider bars that touch so the "skyline is composed of connected horizontal and vertical lines. The "skyline" provides a gestalt shape and a general height for comparison. This version includes labels.

```
thinBar <- function(mat,xSpacing,nr,nc,ylab=NULL,
        ylabLoc=0,ybot=.2,ytop=1.0,mycolors,grid="#C0C0C0",labs)
{
  n <- nrow(mat)
  nvar <- ncol(mat)
  if(missing(xSpacing))xSpacing <- seq(.05,.95,length=nvar)
  if(missing(nr))nr <- floor(sqrt(n))
  if(missing(nc))nc <- ceiling(n/nr)
  if(missing(mycolors))mycolors <- rainbow(7)
  matRange <- apply(mat,2,range)
  matScaled01 <- scale(mat,matRange[1,],diff(matRange))

  px <- as.vector(rbind(xSpacing,xSpacing,rep(NA,nvar)))
  ymin <- rep(ybot,nvar)
  yna <- rep(NA,nvar)
  ydif <- ytop-ybot
  plot(c(0,nc),c(0,nr),type='n',axes=F,xlab='',ylab='')

  for(i in 1:n){
    # row wise
    x <- (i-1)%%nc
    y <- nr - (i-1)%/%nc -1
    ymax <- ymin+matScaled01[i,]*ydif
    rect(x+xSpacing[1],y+ybot,x+xSpacing[nvar],y+ytop,
       col="#CCCCCC",border=NA)
    yMidLines <- y+ybot+c(.25,.5,.75)*ydif
    segments(rep(x+xSpacing[1],3),yMidLines,
          rep(x+xSpacing[nvar],3),yMidLines,col="#909090")
    polygon(x+xSpacing[c(1,1:nvar,nvar)],
         y+c(ytop,ymax,ytop),col="#F0F0FF",border=NA)

    for(j in 1:nvar)
       lines(x+rep(xSpacing[j],2),y+c(ybot,ymax[j]),
            col=mycolors[j],lwd=3,lend='butt')
```

```
    if(!missing(labs))text(x+.5,y+.13,labs[i],adj=.5,cex=.6)
    lines(x+xSpacing[c(1,nvar)],rep(y+ybot,2),lwd=1)
  }
}

windows(width=8,height=6)
par(mai=c(0,0,.7,0))
mycolors <- rainbow(7)
mycolors[5]=mycolors[4]  #playing with colors
mycolors[4]="#000000"    # sticking black in the middle
thinBar(mtcars[, 1:7],mycolors=mycolors,labs=row.names(mtcars))
##End
```

Okay, this glyph plot still lacks a legend to indicate the variables names, the units of measure and an indication of the encoding. It is still beneath our standards but is, at least, a colorful start.

There are still more comparison issues to address. In an interactive setting we might provide many re-expression options.

1) Arrange the glyphs to be near their neighbors in the plot based on the dissimilarity or distance matrix computed from their values.

2) Arrange the glyphs to be in clusters

3) Drag and drop to rearrange variables

4) Provide alternative views
   Scatterplot matrices
   Parallel Coordinate plots
   Q-Qplots for univariate distribution comparison

5) Provide linking and brushing across alternative view

6) Provide focusing techniques in addition to brushing
   Mouse to select one or more cases and
   Highlight cases like this one
   Highlight more cases like this group
   Highlight cases near a path through two cases
   Highlight cases near a path through these three cases

6.1) A slider could control how many cases to highlight
6.2) Consider different highlighting options such as

Blending non-highlighted points with the background
Make highlight points appear closer with 3-D cues

7) Add conventional filtering tools

8) Add variables transformations

6. Generate data on a hypersurface

In mathematics a hypersurface is some kind of submanifold. In algebraic geometry, a hypersurface in projective space of dimension n is an algebraic set that is purely of dimension n - 1.

A hypersurface can be defined based on constraints such as f(x,y,z,w)=0, or by parametric construction as below. Here we create a hypersurface by embedding 3D data in 4 dimensions.

```
# Generate 3-D points
u <- runif(800, -1, .8)
v <- runif(800,- 1, .8)
w <- runif(800, -1, .8)

# Make up function to embed points in 4-D
x1 <- u^2 + 2*v^2 + 3*w^2
x2 <- u + v - w
x3 <- 2*u^2 - v^2 + w^2
x4 <- -u^2 + 7*u*v + v^2 - 2*w^2

hyperSurf <- cbind(x1, x2, x3, x4)
```

7. Stars view of a hypersurface

An interesting question is, can we see the presence of a hypersurface constraint using glyphs that show 4 variables? Can we see that structure is really three dimensional or less?
The hypersurface generation script above does not add noise so we are in the relative simple case of seeing points that are a part of the hypersurface.

As indicated in the companion notes, glyphs don't scale well. Here we use 4 pages to show 800 points. Between change blindness across pages and the lack of sorting within the matrix of glyphs, our changes of seeing a constraint seem pretty slim.

```
hyperSurfR<- apply(hyperSurf, 2, range)
hyperSurf01 <- scale(hyperSurf, hyperSurfR[1, ], scale=diff(hyperSurfR))
apply(hyperSurf01, 2, range)  # check

rowcol <- expand.grid(list(i=1:17,j=1:12))
rowcol <- rowcol[1:200,]

for (page in 1:4){
dat <- hyperSurf01[1:200+200*(page-1),]
windows(width=10,height=8)
par(xpd=T)
stars(dat, scale=F, labels=NULL, nrow=12, ncol=17,
    key.loc=c(34.7, 28.3), key.labels=c('x', 'y', 'z', 'w'),
    main=paste('Hypersurface Part', page))
}
```

The labeling of points is not helpful in this context, so omitted. The plots provide little clue that the data is on a hypersurface. If the stars or other glyphs were carefully sorted to put similar glyphs together then perhaps there would be a chance of seeing structure.

8. Profile glyph view of a hypersurface

This older script produces a more tradition profile plot.

```
# scale the data into [lspace, 1] leaving space for a label
lspace=0  #     lspace = .1
mat <- hyperSurf
matR  <-  apply(mat, 2, range)
nmat <- scale(mat, center= t(matR)[,1] ,
                          scale=diff(matR)/ (1-lspace)  )
nmat <- scale(nmat, center=rep(-lspace, ncol(nmat)), scale=F)

# layout for a page
nr <- 10
nc <- 20
nvar <- ncol(mat)
gap <- .05  # column gap size

# x polygon coordinates
px <- c(gap, seq(gap, 1-gap, length=nvar), 1-gap, NA)
npx <- length(px)

# x polygon coordinates for a full page
x <- rep(rep(0:(nc-1), rep(npx,nc))+px,nr)
ybase <- rep(0:(nr-1), rep(nc*npx, nr))

npage <- (nrow(nmat)-1)%/%(nr*nc) + 1

for (k in 1:npage){
  windows(width=10, height=8)
  plot(c(0, nc), c(0, nr), type='n', axes=F, xlab='', ylab='',
    main=paste("Hypersphere Page", k))
  subs <- seq((k-1)*nr*nc+1,min(k*nr*nc,nrow(nmat)))
  ns <- length(subs)
  locy <-    as.vector(t(cbind(rep(lspace, ns), nmat[subs, ],
               rep(lspace,ns), rep(NA, ns))))
  if(ns < nc){x <- x[1:length(locy)];ybase= ybase[1:length(locy)]}
  polygon(x,locy+ybase,col=4)
  polygon(x,locy+ybase,density=0)
 #text((rep(0:(nc-1),nr)+.5)[1:ns],rep(0:(nr-1),
 #   rep(nc,nr))[1:ns],labels=as.character(subs),cex=.55,adj=.5)
}
```

9. Color matrix views of a hypersurface

A row of four colored rectangles can be considered a glyph. While a certain minimum size rectangle is required to see color, colored boxes take little space. Using this representation makes it easier to get all 800 points on one page.

There are costs associate with using color. Converting continuous values to five color classes looses a lot of information. Further, color is poor encoding even for ordered variables.

```
#    Note: this could be recode to use rect() instead of polygon, or used image

nmat <- hyperSurfS01

pencolor <- matrix(c(
0, 0, 0,
1, 1, 1,
.5, .5, .5,
0, .43, .86,
.5, .75, 1,
.9, .9, .9,
.94, .47, 0,
1, .75, .5),ncol=3, byrow=T)
mycolors <- rgb(pencolor[, 1], pencolor[, 2], pencolor[, 3], max=1)

windows(width=7.5, height=10)
palette(mycolors)
pan <- panelLayout(nrow=1, ncol=8, leftMar=0, colSep=.1)

px <- c(0, 0, 1, 1, NA) - .5
py <- c(0, 1, 1, 0, NA) - .5

px2 <- c(px+1, px+2, px+3, px+4)
polyx <- rep(px2, 100)

py2 <- rep(py, 4)
polyy <- rep(py2,100) + rep(seq(100,1),rep(20,100))

for (i in 1:8){       #process 8 groups of 100 cases
panelSelect(pan,1,i)
panelScale(rx=c(0.5, 4.5), ry=c(0, 101))
ib <- 100*(i-1)+1
ie <- ib+99
```

```
dat <- as.vector(t(nmat[ib:ie,]))   # get 100 cases and
breaks <- seq(0,1,length=6)
pen <- as.vector(cut(dat,breaks, include.lowest=T,labels=F))
polygon(polyx,polyy,col=pen+3,border=F)
polygon(polyx,polyy,col=3,density=0)
}
panelSelect(pan,margin='top')
panelScale()
text(.5,.5,'Hypersurface',cex=1.2,adj=.5)
```

Again the structure is not obvious and again we could complain about the lack of sorting. This time we will sort.

My two favorite multivariate sorting approaches are
1) breadth traversal of a minimal spanning tree
2) first eigenvector of a singular value decomposition

More the one package for R has a minimal spanning tree algorithm, but none that I know of provides a breadth and radial traversal that is available in Splus. I have not finished writing the breadth traversal and have commented out the Splus code from previous classes below.

```
# Splus spanning tree traversal ordering
# I would sort the cases by a minimal spanning traversal
# This uses the data with individually scale columns
#    so corresponds more to the graphical representation
#    that to the original data.
# rmst <- mstree(nmat)
# cmst <- mstree(t(nmat))
# newmat <- nmat[rmst$order[,1],cmst$order[,1]]
```

```
# Singular value decomposition first  singular vector order
# This uses the column scaled data and this might be argued.

# SVD ordering
tmp <- svd(nmat)
ordx <- order( tmp$u[, 1])
ordy <- order( tmp$v[, 1])
newmat <- nmat[ordx, ordy]

windows(width=7.5,height=10)
pan <- panelLayout(nrow=1, ncol=8, leftMar=0,  colSep=.1)


for (i in 1:8){    #process 8 groups of 100 cases
```

```
  panelSelect(pan, 1, i)
  panelScale(rx=c(0.5, 4.5), ry=c(0, 101))
  ib <- 100*(i-1)+1
  ie <- ib+99

  dat <- as.vector( t(newmat[ib:ie, ] ))   # get 100 cases and
  breaks <- seq(0,1,length=6)
  pen <- cut(dat,breaks, include.lowest=T,labels=F)
  polygon(polyx, polyy, col=pen+3, border=3)
}
panelSelect(pan, margin='top')
panelScale()
text(.5,.5,'SVD Sorted Hypersurface', cex=1.2,adj=.5)
```

10. Parallel coordinates

Parallel coordinates provide a way to plot all the data in a single plot.
There is a lot of overplotting, but the edges suggest structure. The internal lines can suggest structure when not too badly obscured by overplotting. (There are density representations for parallel coordinate plots.) One way to thin lines is to bin the data in 4-D and select a representative point from each bin to overplot or use in a separate plot.

Here we select 8 rows from the SVD sorted rows from Section 7 above, put the rows last so they overplot, and plot them in a different color.

```
library('lattice')
# Hypersurface example
windows()
parallel(~mat,col=4,xlab="Hypersurface")
subs <- seq(50,750,by=100)
newframe <- data.frame(rbind(newmat[-subs,],newmat[subs,]),
  Highlight =  as.factor( rep(c('no', 'highlight'), c(792, 8) ) ) )

# Note "highlight" get becomes level 1  and 'no" level 2 in
# the factor due to the default alphabet ordering
# in creating a factor.
# The "no" = level 2 colors and line with below  "green" and lwd=1

windows()
parallelplot(~newframe[1:4], groups=Highlight, data=newframe,
  col=c("purple","green"),
   lwd=c(2,1),
  key=list(  columns=2,
```

```
            text=list(c("Highlight","Rest")),
             lines= list(lwd = c(2,1), col = c('purple','green')),
             title="Highlight 8"
        )
)
```

11. Rotating Ray glyph view of a hypersurface

The sequel to this class will show a tour of 4-D data using stereo ray glyphs.

Here we spin the ray glyphs to provide a 3-D view that is augmented by a ray angle encoding the 4th variable. The construction spins about the y-axis but implementation could be made more general.

Note: if the plot is too slow and you want to stop, click in the plot and then depress the esc key.

In term of spinning 3-D point cloud by mouse it is convenient to think of mousing on an invisible sphere, dragging the mouse conceptually attached to the sphere in the desired direction and speed, and letting it go. This was available in ExplorN.

```
# Scale in [-1 1],  subtract the midrange as a start
matRange <- apply(mat,2,range)
matScaledOne <- scale(mat,(matRange[1,]+matRange[2,])/2,
    scale=diff(matRange)/2)
apply(matScaledOne, 2, range)  # check

xyz  <- matScaledOne[,1:3]
x <- matScaledOne[, 1]
y <- matScaledOne[, 2]
z <- matScaledOne[, 3]
w <- matScaledOne[, 4]

d <- sqrt((xyz*xyz) %*% c(1,1,1)) # largest value from origin of 3D rotation
d <- max(d)
plim <- c(-d,d)

dx  <-  .03*d*cos(w*pi/2)
dy  <-  .03*d*sin(w*pi/2)

windows(width=6.5, height=6.5)
par(pty='s',mai=c(.01,.01,.01,.01))
nviews <- 500
for (i in 1:nviews){
```

```
  angle <- (.02*i) %% (2*pi)
  xnew <- x*cos(angle)+z*sin(angle)
  dev.hold(1)
  plot(plim, plim, type='n', axes=F, xlab='', ylab='')
  dev.hold(1)
  points(xnew, y, pch=20,cex=.50)
  dev.hold(1)
  segments(xnew, y, xnew+dx, y+dy)
  dev.flush(3)
}
```

12. Scatterplot matrix

Without further augmentation I do not consider the scatterplot matrix a type of glyph plot. It is often more useful than a glyph plot. This example we see structure that was not evident in most of the glyph plots above.

```
#Hypersurface
splom(~mat,xlab="Hypersurface")
```

Now we see more edges and they are suggestive of constraints and a lower dimensional structure. Overplotting is still a bit of an issue.

We have serious difficult in linking together the 6 bivariate components of a 4-tuple in a scatterplot matrix. That is just one 4-d data point. Brushing, when used, helps us to pair 2 4-D points, but mentally assessing the distance between the 2 4-D points seems really unnatural. The task of assessing the ratio distances between a pair of red 4-D points and a pair of cyan 4-D points is more complex yet.

For 3-D data, I think we judge distances and ratios of distances better in 3-D scatterplot than in at scatterplot matrix. I suspect we can assess the distances and distance ratios somewhat better in a parallel coordinate plot than in a scatterplot matrix.

For 4-D data I think the scatterplot matrix will still be the worst. My conjecture is that the stereo ray glyph would not be very good but better than the parallel coordinate views.

Basically I think that the color link binding is a scatterplot matrix not localized so harder to deal with in working memory. The same coordinate appears more than once. The parallel coordinate points are connected, better localized, and not duplicates. The stereo ray glyph is more of an integral object and very localized. Three of the coordinates are in a framework that we normally used to judge distance, two points space.

Making the comparison of the three plotting approaches was Qi Zhu's (my first

Ph.D. student) intended dissertation topic in the mid 1990's. I had plans for both the experimental design and an ExplorN interface with subject feedback and automated recording of results in mind. Qi had some of her friends looking at example in an informal pilot study when she suddenly left for a job in clinical trials. The research was never done.

It seems me that the ability to assess distance is crucial to much of our pattern perception. There is room to learn about what works best and gain a better idea about when we are just wasting our time and when we can glean some of the structure in modest dimensional data sets.

13. References

Bayly, C. I., P. Cieplak, W. D. Cornell and P. A. Kollman. 1993. A well-behaved electostatic potential based method using charge restraints for deriving atomic charges: the RESP model, *Journal of Physical Chemistry* 97, 10 269–10 280**.**

Carr, D. B., W. L. Nicholson, R. J. Littlefield, and D. L. Hall. 1986. "Interactive Color Display Methods for Multivariate Data." *Statistical Image Processing and Graphics*, eds. E. J. Wegman and D. J. DePriest, pp. 215-250, Marcel Decker, New York. **Stereo rays, contours, texture**

Carr, D. B. and W. L. Nicholson. 1988. "EXPLOR4: A Program for Exploring Four-Dimensional Data." *Dynamic Graphics for Statistics*, eds. W. S. Cleveland and M. E. McGill, pp. 309-329. Wadsworth, Belmont, California. **Stereo rays, parallel coordinate GUI, etc.**

Carr, D. B. 1994. Comments on "Prosection Views: Dimensional Inference though Sections and Projections", *Journal of Computational and Graphical Statistics*, pp 369-376. **Stereo Rays**

Carr, D. B., A. R. Olsen, and D. White. 1992. "Hexagon Mosaic Maps for Display of Univariate and Bivariate Geographical Data." *Cartography and Geographic Information Systems*, Vol. 19, No. 4, pp. 228-236,271. **Bivariate rays and maps**

Carr, D. B., A. R. Olsen, S. M. Pierson, and J. P. Courbois. 1998. "Boxplot Variations in a spatial context: An Omernik Ecoregion and Weather Example," *Statistical Computing & Graphics Newsletter*, Vol. 9 No. 2 pp. 4-13. **Ray angle box plots and maps**

Carr, D. B., R. J. Littlefield, W. L. Nicholson, and J. S. Littlefield. 1987. "Scatterplot Matrix Techniques For Large N." *Journal of the American Statistical Association* 82(398):424-436. **GUI transfer function**

Carr, D. B. 1995. "Scanning a 4-D Domain for Local Minima: A Protein Folding Application," *Statistical Computing & Graphics Newsletter*, Vol. 6 No. 2 pp. 8-12. **Split Scale Rays**

Eick, S. and G. Wills. 1993. "Navigating Large networks with hierarchies", *Proceedings of IEEE Visualization '93*, San Jose, CA, pp. 204-210.

Furnas, G. W. and A. Buja. 1994. "Prosection Views: Dimensional Inference though Sections and Projections", *Journal of Computational and Graphical Statistics*, pp 323- 353.

Gower, J. C. and P. G. N. Digby, 1981. "Expressing Complex Relationships in Two Dimensions," in *Interpreting Multivariate Data*, ed. V Barnett. John Wiley and Sons, New York. pp. 83-118.

Murrel. P. 2006. *R Graphics*. Chapman and Hall. New York.

Sun. R. [2008] *2-D and 3-D Layouts to Aid Human Cognition of Local Structure in Multivariate Data.* Ph.D. Dissertation George Mason University.

Tukey, P. A. and J. W. Tukey. 1981. "Summarization, Smoothing; Supplemented views." in *Interpreting Multivariate Data*, ed. V Barnett. John Wiley and Sons, New York. pp. 245-275

Ware, C. 2008. *Visual Thinking: for Design*. Morgan Kaufmann. New York.)

Wickham H. 2009. *ggplots2 Elegant Graphics for Data Analysis*, Springer, New York.

Wilkinson, L. 2005. *The Grammar of Graphics*. Springer, New York.