Chapter 6

Graphs – a miscellany

6.1 Introduction

In this chapter a miscellany of graphical applications are presented. Usage in the literature is variable with some of the approaches illustrated having had little, if any, archaeological application. Nevertheless implementation is usually straightforward and the methods can be quickly applied for exploratory purposes. Labeled pairs plots and two-dimensional contour plots have general application, while ternary diagrams have a specialist niche. Confidence ellipsoids are widely used for display purposes in the archaeometric literature; convex hulls have had limited use. I do not recall seeing any uses of correlation diagrams or of Chernoff faces.

6.2 Enhanced pairs plots

Pairs plots, or scatterplot matrices, are illustrated elsewhere (e.g., Figure 5.6). They provide a tool that merits routine use. Here an enhanced pairs plot, obtained with the `scatterplotMatrix` function from the `car` package, is illustrated.

The chemical compositional data from Table B.1, used in Section 2.2 are revisited. It was shown there that the variables Ca, Fe, K and Mg in the file `tubb.data` did a good job of distinguishing between the regions held in the variable `tubb.region` (Figure 6.1).

```
library(car)   # load package car
scatterplotMatrix( ~Ca+Fe+K+Mg, data = tubb.data, smooth = F,
by.group = F, group = tubb.region,reg.line = F)
```

That the regions are chemically separate is evident, as is the outlying value for K in Region 2 previously noted. The KDEs down the diagonal are optional, and other choices of graphical display are possible, or none at all. The plots are useful

in this instance for emphasizing the multi-modality of the data. The *rug* at the bottom of each KDE displays the individual data points.

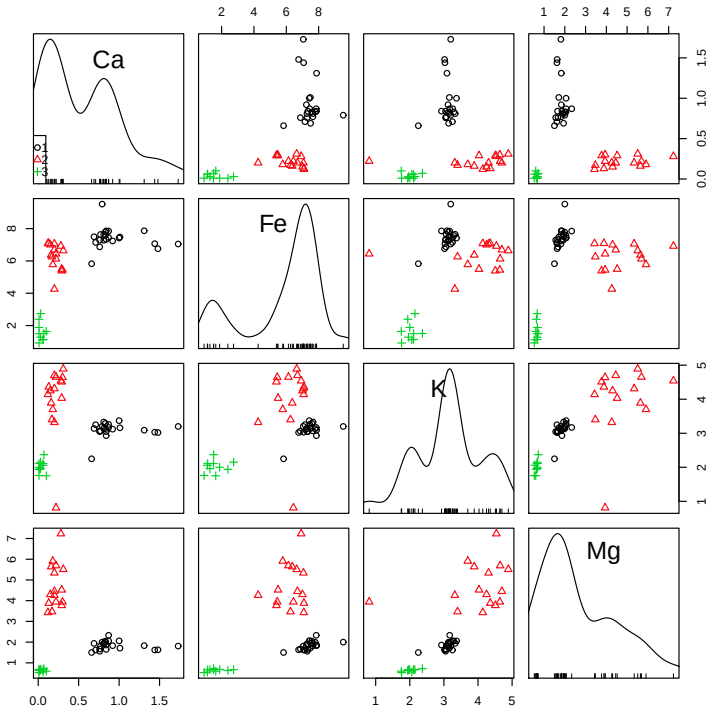

Figure 6.1: *A labeled pairs plot, or scatterplot matrix, for a subset of the oxide compositional data from Table B.1.*

6.3 Graphics with more than one variable

6.3.1 Two-dimensional KDEs

The examples in this section use data on the dimensions of loomweights, from Tables B.3 and B.4. Using KDEs, Baxter and Cool (2008) and Baxter *et al.* (2010) showed that the weights of the loomweights had an unexpected bimodal distribution; the latter paper explores reasons for and some of the implications of this. Baxter and Cool (2010a), using formal statistical tests, establish that the bimodality is not an accidental by-product of sampling variability. The bimodality is reflected in the distribution of `height` and `volume` which, along with `weight`, are the variables used in Figures 6.2 and 6.3. A reduced data set omitting outlying weights of more than 400g or less than 90g was used.

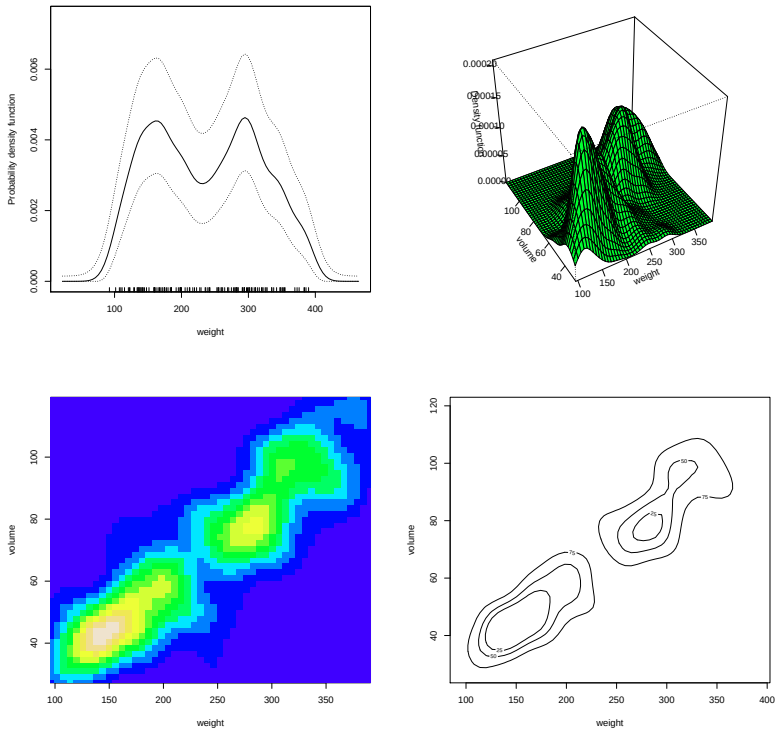

Figure 6.2: *One and two-dimensional KDEs for the loomweight data of Tables B.3 and B.4 for weight volume.*

All the plots were produced using defaults in the `sm` package except that bandwidths were subjectively chosen. They are otherwise estimated separately using automatic methods. The package is associated with the book of Bowman and Azzalini (1997). Other than loading other packages needed and setting up the data appropriately only four lines of code are needed to produce the four graphs.

The upper-left plot, a KDE for `weight`, establishes the bimodality of the variable. The added confidence band provides reassurance that the bimodality is genuine. Remaining plots shows different methods of displaying output based on a two-dimensional kernel density estimate for `weight` and `volume`/10000. An alternative would be to use the cube-root of volume, but essentially the same results are obtained. Perspective, image and contour plots are shown, with all clearly demonstrating bimodality.

6.3.2 Ellipses, convex hulls, contours – one group

Two-dimensional KDEs can also be obtained using the `kde2d` function from the `MASS` package. Similar display methods can be used (Venables and Ripley, 2002: 131), and the contour plot in Figure 6.3 was obtained using this function. In Figure 6.3 different summary displays of a mass of points are illustrated. They are potentially useful if the amount of data makes perception of any pattern difficult, or if the distribution of two or more large groups is to be compared.

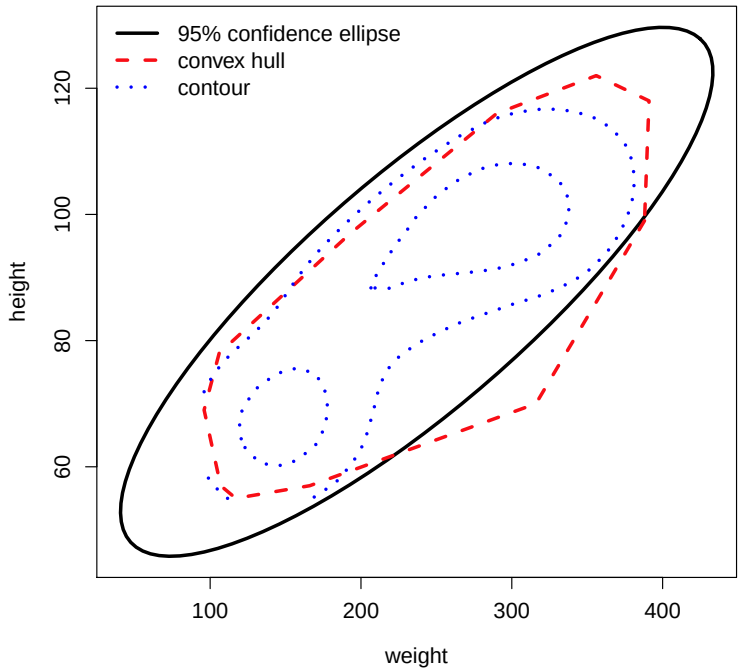

Figure 6.3: *Confidence ellipse (95%), convex hull and contour plot at two levels for the reduced loomweight data using weight and height.*

Confidence ellipsoids are quite widely used in studies of artifact provenance based on chemical compositional data, and elsewhere. In provenance studies, based on an $n \times p$ data matrix where n and p can be quite large, information on groups within the data is often available. The groups may be defined independently of the chemical composition, as in the regional data in Table B.1, or may be derived from some statistical procedure such as cluster analysis (Chapter 10). Bivariate plot may be based on a selection of a pair of variables from the original p, or derived variables such as the linear combinations obtained from a PCA (Chapter 7).

Group separation can be investigated by comparing confidence ellipsoids for the groups on the plot. Where the groups are defined independently of the chemistry visible separation then implies that different provenances are chemically distinct. Where groups are derived from the chemistry the first task is to establish that they are clearly chemically distinct (cluster analyses will produce groups whether they are 'real' or not). Interpreting any grouping in terms of provenance is a separate task, undertaken independently of the chemistry.

Confidence ellipsoids are based on the assumption that within groups the data are sampled from a population with a (bivariate) normal distribution, so that the ellipsoids are *estimates* of the extent of the groups in the population and typically extend beyond the observed limits of the data. The assumption is rarely tested and is not always self-evidently true. Some applications drop, sometimes silently, outlying points that violate the assumption, so that the appearance of a nicely bivariate normal sample of data becomes a 'self-fulfilling prophecy'.

Convex hulls are purely descriptive and summarize the data in the form of an envelope based on the minimum bounding set of points; they can be obtained with the `chull` function. They have been used relatively infrequently, except possibly in the GIS literature (which I am not very familiar with) where delineation of the spatial extent of a set of features or artifacts with something in common is an obvious application. Ringrose (1992) provides an interesting application. A correspondence analysis (Chapter 9) of an $r \times c$ table of data can be used as the basis for a bivariate plot showing the relationship between columns based on coordinates for the first two components. The stabilty of the plot is an issue. Using computer intensive methodology, bootstrapping, Ringrose replicates the data N times, producing a set of N distinct coordinates for each column marker. Convex hulls are used to display the distribution of points for each marker, and these can be compared for overlap or its lack to see how distinct each column marker really is.

6.3.3 Ellipses, convex hulls, contours – several groups

In Figure 6.3, for illustration, only one group was used, displaying a 95% confidence ellipsoid and convex hull for the data, which may be contrasted with the contour plot also shown. Contour plots can reveal sub-regions of dense point scatters that the first two cannot capture. The inner contours contain about 42% of the data.

To illustrate the use of confidence ellipsoids and convex hulls when the comparison of two or more groups is of interest, three groups were defined with Ward's method of cluster analysis (Section 10.3) using three variables `weight`, `height` and `volume`. A PCA was undertaken with these variables and Figure 6.4 shows a bivariate plot based on the first two components, with points labeled by group membership and confidence ellipsoids added for each group.

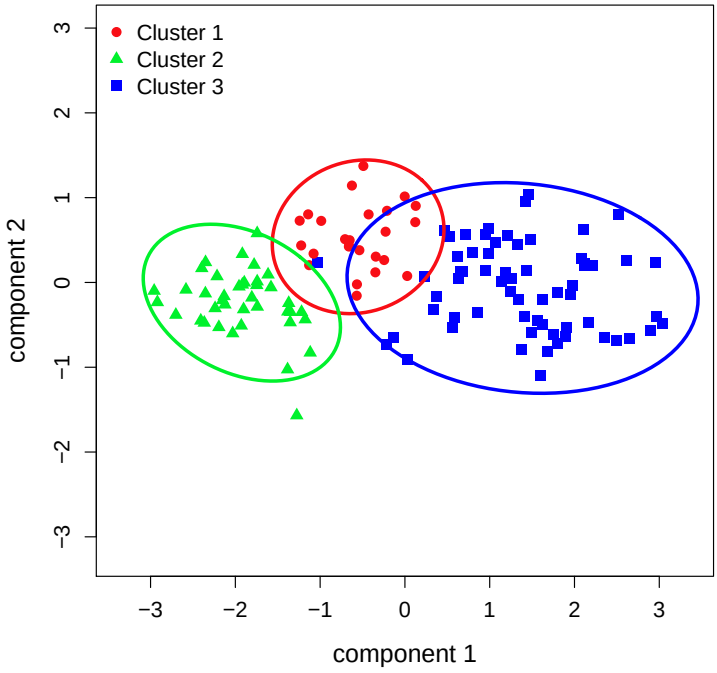

Figure 6.4: *Confidence ellipses (95%) for the reduced loomweight data using weight and height based on a partitions into three groups, determined by a Ward's method cluster analysis. See the text for fuller details.*

With only three variables and three groups defined by cluster analysis – Ward's method in particular – good group separation is to be expected, and this is what we get. There is some overlap between the central group and the other two groups, owing to the extent of the ellipsoids which go beyond the limits of the data. Visually it can be seen that the groups are largely distinct and the ellipsoid representation does not show this as effectively as one might wish.

The comparable plot, Figure 6.5, using convex hulls for the three groups is more satisfactory. The idea of peeling is illustrated. The outer hull is stripped away and a second hull calculated for the remaining data. This can be repeated so long as sufficient data remains. Only one peel is needed here to completely separate groups, the original overlap being attributable to one case.

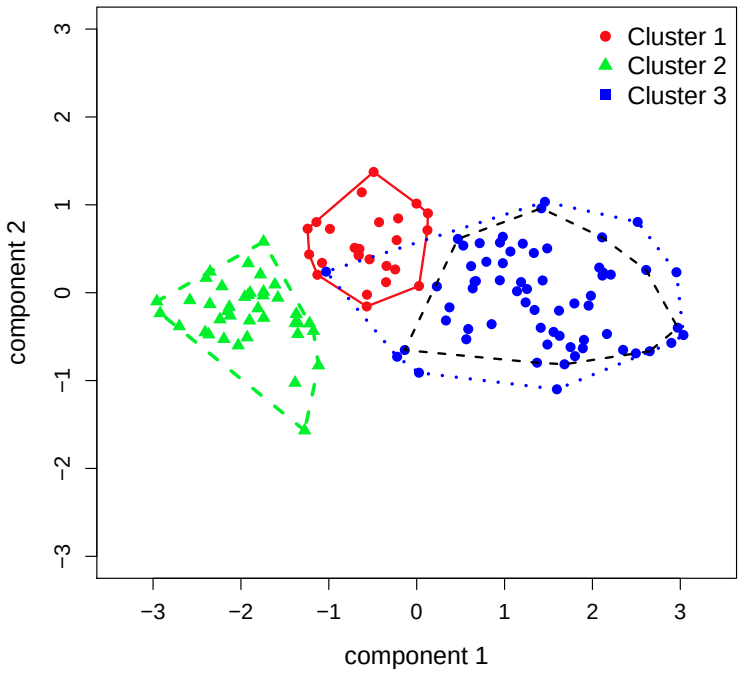

Figure 6.5: *Convex hulls for the reduced loomweight data using weight and height, based on a partition into three groups, determined by a Ward's method cluster analysis. See the text for fuller details.*

6.4 Correlation diagrams

The correlation between two variables can be represented as an ellipse. An elongated ellipse that slopes to the right is associated with a positive correlation; to the left indicates a negative correlation; (near) circularity shows a weak correlation; correlations close to a limit of -1 or +1 show a nearly linear pattern. Correlations can be presented numerically as a table of the correlation matrix; alternatively a correlation diagram, which is a visual summary of the information in the matrix, can be used. Figure 6.6, using the data in Tables B.9 to B.11 on the dimensions of polished Neolithic stone axes (O'Hare, 1990) illustrates the idea.

The general pattern is one of positive correlations and leads us to expect that the first component in the PCA will be interpretable as a *size* component (Section 7.4.1). The thickness variables are very strongly correlated, as are the breadth variables together with the width of the cutting edge, `WC`, while the depth of the cutting edge, `DC`, shows a relatively weaker correlation with most other variables.

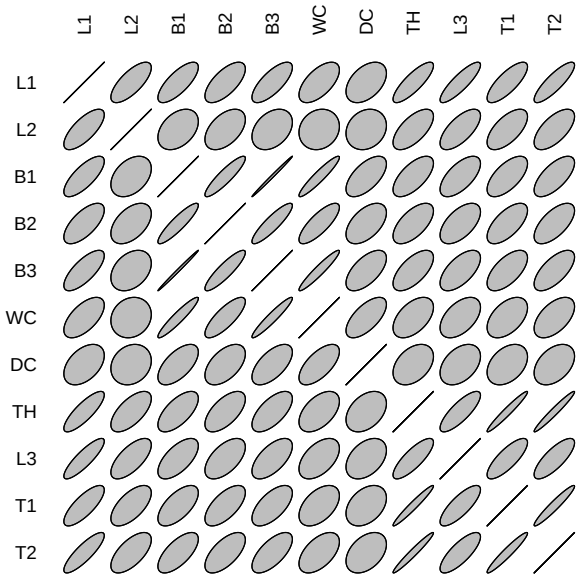

Figure 6.6: *Correlations represented as ellipses for stone axe dimensions.*

It is to be expected, and turns out to be the case, that this will be reflected in components other that the first, which have a *shape* interpretation. To obtain the correlation diagram all that is needed is the following.

```
library(ellipse)   # load package ellipse
plotcorr(cor(stoneaxe.data))
```

6.5 Ternary diagrams

If data are available in the form of an $n \times 3$ data matrix where each row sums to 100% (or 1) they may be represented in the form of a *ternary diagram*[1]. Each row is represented as a point within an equilateral triangle, with proportions represented by the perpendicular distances to the axes of the triangular coordinate system. The geometry is illustrated in Greenacre (2007: 13). To illustrate, data from Doran and Hodson (1975) given in Table 6.1 are used.

[1] Also called triangular, tri-polar or trinary diagrams or plots.

Levels	Cores	Blanks	Tools
25	21	12	70
24	36	52	115
23	126	650	549
22	159	2342	1633
21	75	487	511
20	176	1090	912
19	132	713	578
18	46	374	266
17	550	6182	1541
16	76	846	349
16	17	182	51
14	4	21	14
13	29	228	130
12	133	2227	729

Table 6.1: *Counts of cores, blanks and tools from middle levels of the palaeolithic site at Ksar Akil (Lebanon). This is Table 9.12 from Doran and Hodson (1975).*

Before plotting, the artifact counts in the final three columns must be converted to row proportions so that sample size is ignored. Where $p > 3$, in some applications, a subset of $r = 3$ columns is selected and rescaled, or $r = 3$ new variables that are linear combinations of subsets of the original columns are defined. Several different `R` packages contain functions to plot ternary diagrams and Figure 6.7 shows some of the different plotting options available. The packages used are listed in the caption; there are several others that might equally well have been chosen.

Because the proportions must sum to 1, given any two the third is readily calculated so the data are two-dimensional. The upper-left plot in the figure is three-dimensional, but a view has been chosen to show that the data can be 'captured' in a two-dimensional slice. There are choices that can be made about positions of labels, the inclusion of a grid or not and so on, with different packages differing in the options allowed. What is chosen may be partly a matter of preference. The lower- right plot, for example, only uses the smallest triangle needed to include all the data and this can help in reading a plot as it minimizes 'bunching' of points. It is common in practice to label the vertices at the apexes of the triangle as in the lower-left plot, but placing labels as in the lower-right plot makes it easier to see which axis a label refers to. The plots show some evidence of a seriation (not perfect) with blades tending to increase from levels 25 to 12, while tools decrease. Seriation is an early use to which ternary diagrams were put in archaeology (Meighan, 1959).

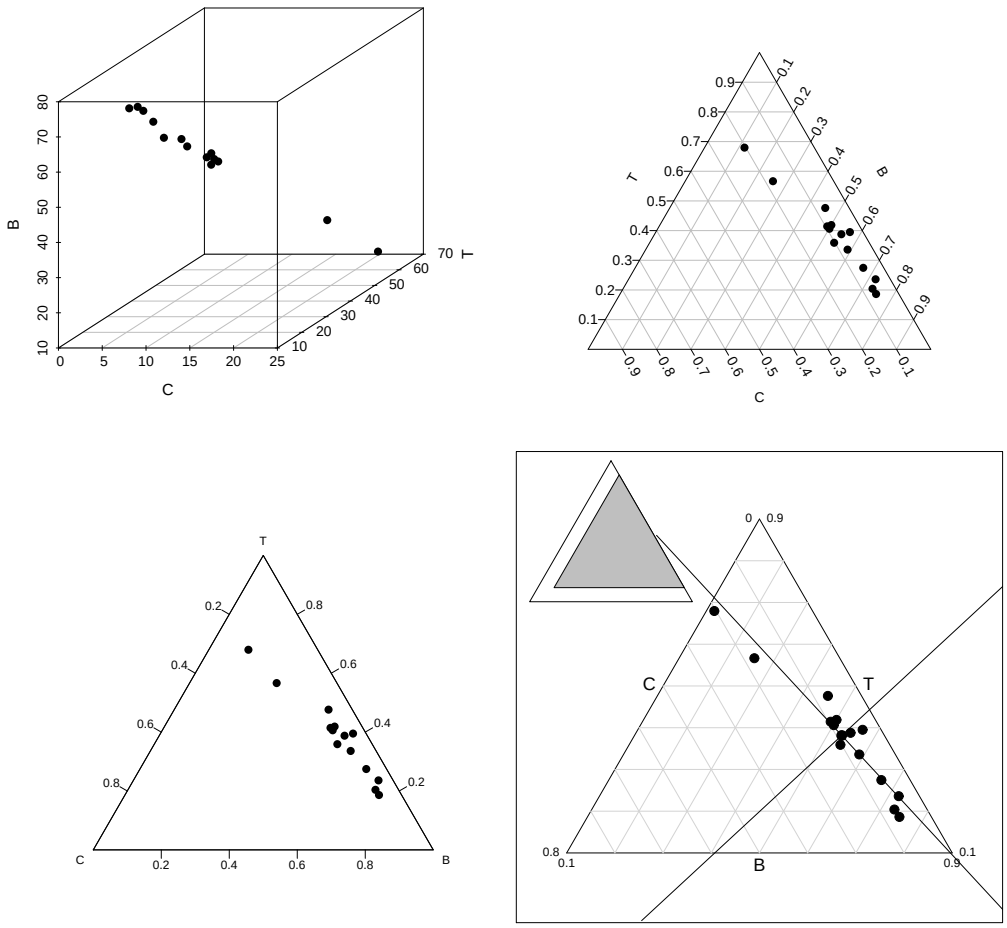

Figure 6.7: *Ternary plots from different packages for the data of Table 6.1. Reading clockwise from the upper-left the functions* `scatterplot3d`, `triax.plot`, `triangle.plot` *and* `plot.acomp` *from the packages* `scatterplot3d`, `plotrix`, `ade4` *and* `compositions` *were used.*

The use of ternary diagrams is scattered, often appearing in 'specialist' publications that draw on the traditions of the specialization involved. One such area is zooarchaeology where, for example the relative proportions of three species such as cattle, sheep/goats and pigs are displayed as points within a ternary diagram.

The data used for illustration are based on Figures 1–4 in Hesse (2011). For four different regions up to six clustered barplots for different site types show the proportions of cattle, sheep/goat and pigs in each assemblage, of which there are 20. Hesse does not use ternary diagrams, but the data are based on King (1999) who does make extensive use of them. The information contained in the 20

clustered barplots shown in Hesse can be recast as in Table B.12 and, minimally, displayed in a single ternary diagram as in the upper-left plot of Figure 6.8.

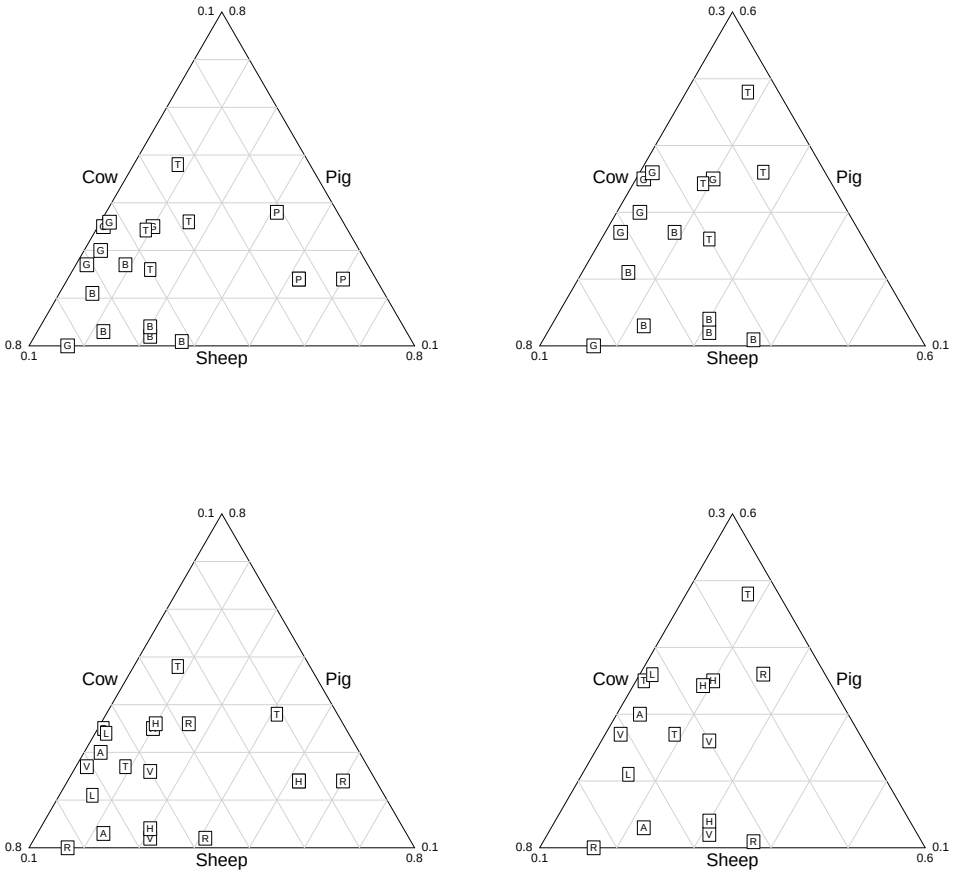

Figure 6.8: *Ternary plots, using different labels, for data derived from Hesse (2011) (Table B.12) based on King (1999). The upper plots are labeled according to the region of the site type, the site type being used to label the lower plots. Plots to the left omit the data from Roman Provence.*

Labeling is by region and two of the assemblages from Roman Provence (P) are identical, so only 19 points are visible. The `triangle.plot` function from the `ade4` package was used. The most obvious feature of the upper-left plot is the complete separation of assemblages from Roman Provence from other regions. Other regions separate reasonably well, though not perfectly. This is a bit clearer in the upper-right plot omitting data for Roman Provence. Relative to other regions Roman

Provence has a higher proportion of sheep/goats and lower proportion of cattle; Roman Britain tends to have a higher proportion of cattle and Roman Germany a lower proportion of sheep; ignoring Roman Provence the Three Gauls tend to have lower proportions of cattle and sheep/goats and hence higher proportions of pig. Variation with respect to site type is harder to discern, and numbers are probably too small to admit generalization.

Hesse (2011: 217–218) reaches essentially these conclusions based on clustered barplots[2]. The ternary diagram(s) are a more economical way of presenting the data.

Another use of ternary diagrams in zooarchaeology is for comparing mortality patterns by plotting age profiles such as juvenile, prime and old. Steele and Weaver (2002) list several papers that have used this approach of which Stiner (1990) is the earliest. Stiner illustrates the division of a ternary diagram into regions corresponding to general types of mortality pattern, allowing the type of an individual assemblage to be characterized. In other contexts such divisions are called phase diagrams. Geoarchaeology (silt/sand/clay diagrams to characterize soils) and archaeometallurgy are other areas of specialist application. Googling 'ternary diagram' with an appropriate choice of other terms will produces plenty of examples. Steele and Weaver (2002: 319) note that ternary diagrams take no account of sample sizes for assemblages, making statistical comparisons between different sets of data problematic. They propose the use of resampling methods (bootstrapping) to simulate the distribution from which a set of data is sampled.

6.6 Chernoff faces

No text of this kind is complete without an illustration of Chernoff faces. In fact I've never seen them used in anger, though they appear quite often in texts on multivariate analysis and are fun. Figure 6.9 illustrates, using the pottery chemical compositional data of Table B.1. To get this use the `apl` package, `library(aplpack)` and `faces` function, `faces(tubb.data)`.

[2]Attention is drawn to the fact that data are averaged across provinces, concealing variation *within* regions.

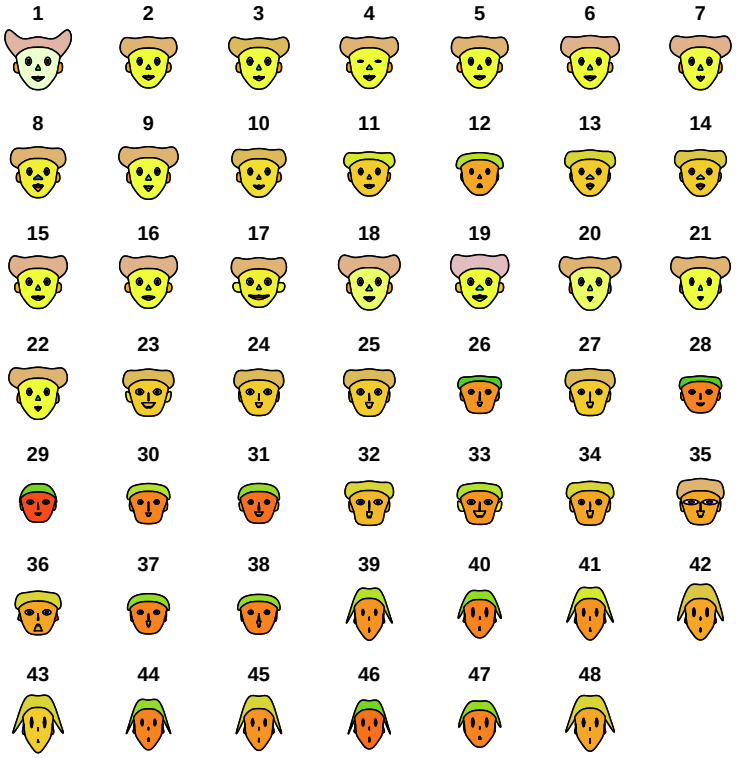

Figure 6.9: *Chernoff faces for the chemical compositional data of Table B.1.*

A list showing how variables correspond to features is returned, but it is the overall impression that is most useful. The three regions from which the pottery comes correspond to cases 1-22, 23-38 and 39-48. The Region 3 faces are distinctive – they have a lean and hungry look and appear to be rather surprised and concerned, perhaps worried about the fact they are growing pigtails.

6.7 R notes

Figure 6.2

```
kde2.plots <- function() {
library(sm)  #load sm package
wt <- loomweights$weight
x <- loomweights[wt > 90 & wt < 400,]
weight <- x[,6]; volume <- x[,7]/10000
win.graph()
sm.density(weight, display = "se", h = 15, lwd = 8)
win.graph()
sm.density(cbind(weight, volume), display = "persp", h = c(15,6))
win.graph()
sm.density(cbind(weight, volume), display = "image", h = c(15,6))
win.graph()
sm.density(cbind(weight, volume), display = "slice", h = c(15,6))
}
kde2.plots()
```

The third and fourth lines select loomweights with weights between 90g and 400g.

Figure 6.3

```
EllipseEtc <- function() {
library(ellipse)
library(MASS)
wt <- loomweights$weight
z <- loomweights[wt > 90 & wt < 400,]
height <- z$height; weight <- z$weight
X <- cbind(weight, height)
m1 <- mean(X[,1]); m2 <- mean(X[,2])

Z <- ellipse(cov(X), centre = c(m1, m2))   # ellipse
plot(Z)

hpts <- chull(X)        # convex hull
hpts <- c(hpts,hpts[1])
lines(X[hpts,])

K <- kde2d(weight, height)   # contour plot
contour(K, add = T, drawlabels = F, nlevels = 5)
}
EllipseEtc()
```

Some presentational arguments and the legend are omitted. Line type and width, color and character expansion can be controlled in all the plotting functions. The ellipse is plotted first as it is the most extensive of the various plots and determines the scale on which they are superimposed. Otherwise the `xlim` and `ylim` arguments need to be experimented with.

Figure 6.4

```
multiple.ellipse <- function() {
library(ellipse)
library(MASS)
wt <- loomweights$weight
z <- loomweights[wt > 90 & wt < 400,]
height <- z[,1]; weight <- z[,6]; volume <- z[,7]
X <- cbind(weight, height, volume)
nclust <- cutree(hclust(dist(scale(X)), method = "w"), k=3)

# Set-up plotting characters and colors
Symbol <- rep(16, length(nclust))
Symbol <- ifelse(nclust == 2, 17, Symbol)
Symbol <- ifelse(nclust == 3, 15, Symbol)

Col <- rep("red", length(nclust))
Col <- ifelse(nclust == 2, "green2", Col)
Col <- ifelse(nclust == 3, "blue", Col)

PCA <- prcomp(scale(X))$x[, 1:2]

eqscplot(PCA[,1], PCA[,2], pch = Symbol, col = Col,
xlim = c(-3.5, 3.5))

X1 <- PCA[nclust == 1, ]
X2 <- PCA[nclust == 2, ]
X3 <- PCA[nclust == 3, ]
m11 <- mean(X1[,1]); m12 <- mean(X1[,2])
m21 <- mean(X2[,1]); m22 <- mean(X2[,2])
m31 <- mean(X3[,1]); m32 <- mean(X3[,2])

Z1 <- ellipse(cov(X1), centre = c(m11, m12)); lines(Z1)
Z2 <- ellipse(cov(X2), centre = c(m21, m22)); lines(Z2)
Z3 <- ellipse(cov(X3), centre = c(m31, m32)); lines(Z3)
}
multiple.ellipse()
```

Figure 6.5

```
chull.loomweights <- function(){
library(MASS)

wt <- loomweights$weight
z <- loomweights[wt > 90 & wt < 400,]
height <- z[,1]
weight <- z[,6]
volume <- z[,7]
X <- cbind(weight, height, volume)

nclust <- cutree(hclust(dist(scale(X)), method = "w"), k=3)
PCA <- prcomp(scale(X))$x[, 1:2]

eqscplot(PCA[,1], PCA[,2], type = "n")
X1 <- PCA[nclust == 1, ]
X2 <- PCA[nclust == 2, ]
X3 <- PCA[nclust == 3, ]
points(X1[,1], X1[,2], pch = 16, col = "red", cex = 1.2)
points(X2[,1], X2[,2], pch = 17, col = "green2", cex = 1.2)
points(X3[,1], X3[,2], pch = 16, col = "blue", cex = 1.2)

hpts <- chull(X1)       #plot for cluster 1
hpts <- c(hpts,hpts[1])
lines(X1[hpts,], lwd = 2, col = "red")

hpts <- chull(X2)       #plot for cluster 2
hpts <- c(hpts,hpts[1])
lines(X2[hpts,], lty = 2, lwd = 3, col = "green2")

hpts <- chull(X3)      #plot for cluster 3
hpts <- c(hpts,hpts[1])
lines(X3[hpts,], lty = 3, lwd = 3, col = "blue")

points = chull(X3)     # One peel of the data for the cluster
X3 <- X3[-points,]
hpts <- chull(X3)
hpts <- c(hpts,hpts[1])
lines(X3[hpts,], lty = 2, lwd = 2, col = "black")
}
chull.loomweights()
```

Figure 6.7

```
Ksar.ternary <- function() {
# Set up data
Ksar <- Ksar_Akil[,-1]
Ksarsum <- apply(Ksar, 1, sum)
Ksar <- Ksar*100/Ksarsum

win.graph(); library(scatterplot3d)
scatterplot3d(Ksar$C, Ksar$T, Ksar$B, xlab = "C", ylab = "T",
zlab = "B")

win.graph(); library(plotrix)
triax.plot(Ksar, cex.ticks = 1.2, show.grid = TRUE)

win.graph(); library(compositions)
plot.acomp(Ksar, axes = TRUE)

win.graph(); library(ade4)
triangle.plot(Ksar, addaxes = TRUE, box = TRUE, cpoi = 2.5)
}
Ksar.ternary()
```

For other than `triangle.plot` the usual presentational arguments are available and not shown. See the `?help` facility for the many variations possible with each type of plot.

Figure 6.8

```
library(ade4)
triangle.plot(king.data, show.position = F,
label = king.region, clabel = .8)

triangle.plot(king.data[-c(13:16),], show.position = F,
label = king.region[-c(13:16)], clabel = .8)

triangle.plot(king.data, show.position = F,
label = king.type, clabel = .8)

triangle.plot(king.data[-c(13:16),], show.position = F,
label = king.type[-c(13:16)], clabel = .8)}
```