Data Analysis & Graphics Using R, 2^{nd} edn – Solutions to Exercises (May 1, 2010)

Preliminaries

```
> library(DAAG)
```

Exercise 1
Carry out the principal components analysis of Section Subsection 12.1.2, separately for males and females. Compare the loadings for the first and second principal components in these new analyses with the loadings obtained in Subsection 12.1.2.

We do the analysis (i) for all observations; (ii) for females; (iii) for males.

```
> all.pr <- princomp(na.omit(possum[, -(1:5)]))
> femp.pr <- princomp(na.omit(possum[possum$sex=="f", -(1:5)]))
> malep.pr <- princomp(na.omit(possum[possum$sex=="m", -(1:5)]))
```

One way to compare the separate loadings is to plot each set in turn against the loadings for all observations. We put the code into a function so that we can easily do the plot for each component in turn. The settings for the two elements of `signs` allow us to switch the signs of all elements, for males and females separately. Loadings that differ only in a change of sign in all elements are equivalent.

```
> compare.loadings <- function(i=1, all.load=loadings(all.pr),
+                              fload=loadings(femp.pr),
+                              mload=loadings(malep.pr), signs=c(1,1)){
+   alli <- all.load[,i]
+   fi <- fload[,i]*signs[1]
+   mi <- mload[,i]*signs[2]
+   plot(range(alli), range(c(fi, mi)), type="n")
+   chw <- par()$cxy[1]
+   points(alli, fi, col="red")
+   text(alli, fi, lab=row.names(fload), adj=0, xpd=T, col="red",
+        pos=2, cex=0.8)
+   points(alli, mi, col="blue")
+   text(alli, mi, lab=row.names(mload), adj=0, xpd=T, col="blue",
+        pos=4, cex=0.8)
+   abline(0,1)
+   }
```

Now compare the loadings for the first and second principal components. From examination of the results for default settings for `signs`, it is obvious that a switch of sign is needed for the female loadings.

```
> par(mfrow=c(1,2))
> compare.loadings(1)                    # Compare loadings on 1st pc
> compare.loadings(2, signs=c(-1,1))     # Compare loadings on 2nd pc
> par(mfrow=c(1,1))
```

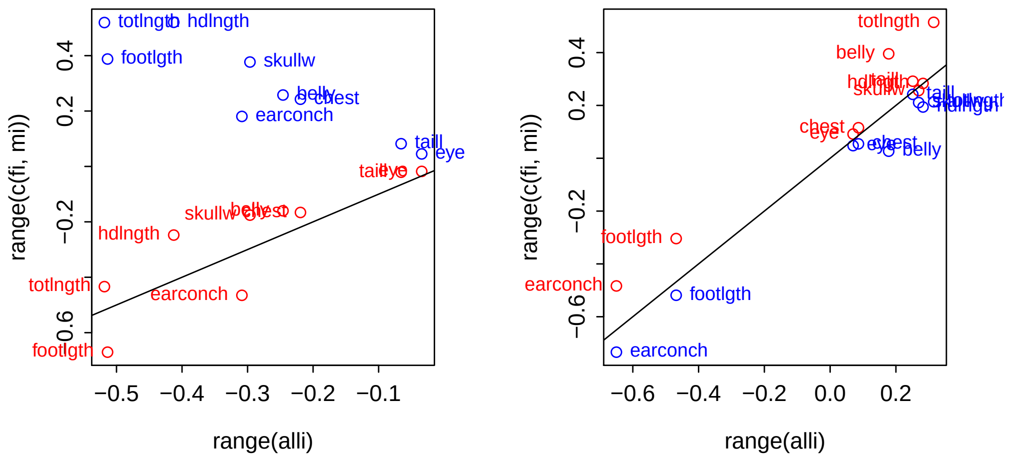

Figure 1: Loadings for females (red) and loadings for males(blue), plotted against loadings for the total data set.

Exercise 2
In the discriminant analysis for the `possum` data (Subsection 12.2.4), determine, for each site, the means of the scores on the first and second discriminant functions. Plot the means for the second discriminant function against the means for the first discriminant function. Identify the means with the names of the sites.

We need only omit the rows that have missing values in columns 6-14. (The variable `age`, in column 4, has two missing values, which are need not concern us.) Hence the use, in the code that follows, of `ccases` to identify rows that have no missing values in these columns. Here is the code used to do the discriminant function calculations:

```
> library(MASS)
> ccases <- complete.cases(possum[,6:14])
> possum.lda <- lda(site ~ hdlngth+skullw+totlngth+ taill+footlgth+
+                    earconch+eye+chest+belly, data=possum[ccases, ])
```

We calculate the means of the scores thus:

```
> possum.fit <- predict(possum.lda)
> avfit <- aggregate(possum.fit$x, by=list(possum[ccases, "site"]),
+                     FUN=mean)
> avfit

  Group.1        LD1        LD2        LD3         LD4         LD5
1       1  4.410258  0.5562407  0.3158968 -0.16740921 -0.06321562
2       2  3.878929 -1.8590986 -0.5402922  0.41948633  0.25835075
3       3 -2.607240  0.6692914  0.5402569  1.06684989 -0.52208918
4       4 -2.554674  1.9662845 -1.3030039  0.23392921  0.57195394
5       5 -3.947575  0.1797326  0.5989668 -0.02540586  0.23510820
6       6 -4.282095 -0.8074082  1.0298269 -0.22913147  0.10259697
7       7 -2.720364 -0.3520005 -1.0986765 -0.29476669 -0.31962856
```

```
           LD6
1  0.005639874
2 -0.022792076
3  0.050718350
4  0.221235010
5 -0.396622367
6  0.302748367
7 -0.033106804
```

The matrix `avfit` has 7 rows (one for each site) and 6 columns (one for each of the six discriminant functions). The row labels can be obtained from the data frame `possumsites`. Here then is the plot:

```
> plot(avfit[,"LD1"], avfit[,"LD2"], xlab="1st discriminant function",
+       ylab="2nd discriminant function")
> chw <- par()$cxy[1]
> text(avfit[,"LD1"]+0.5*chw, avfit[,"LD2"], labels=row.names(possumsites),
+       adj=0, xpd=TRUE)
```

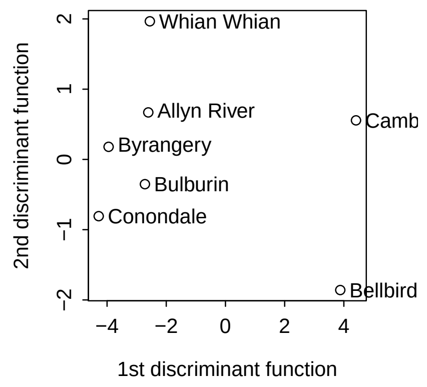

Figure 2: Plot of the second discriminant function against the first discriminant function, for the `possum` data frame. The discriminant functions are designed to discriminate between sites.

Cambarville and Bellbird seem distinguised from the other sites.

Exercise 3
The data frame `possumsites` (*DAAG* package) holds latitudes, longitudes, and altitudes, for the seven sites. The following code, which assumes that the *oz* package is installed, locates the sites on a map that shows the Eastern Australian coastline and nearby state boundaries.

```
library(DAAG); library(oz)
oz(sections=c(3:5, 11:16))
attach(possumsites)
points(latitude, longitude)
chw <- par()$cxy[1]
chh <- par()$cxy[2]
posval <- c(2, 4, 2, 2, 4, 2, 2)
text(latitude+(3-posval)*chw/4, longitude,
     row.names(possumsites), pos=posval)
```

Do the site means that were calculated in Exercise 2 relate in any obvious way to geographical position, or to altitude?

Cambarville and Bellbird, which were distinguished from the main cluster in the plot in Exercise 2, are the southernmost sites.

Exercise 5
Create a version of Figure 12.5B that shows the discriminant line. In the example of Subsection 12.2.1, investigate whether use of `logpet`, in addition to `logwid` and `loglen`, improve discrimination?

Here are the discriminant function calculations:

```
> leaf17.lda <- lda(arch ~ logwid + loglen, data = leafshape17)
> leaf17.fit <- predict(leaf17.lda)
> leaf17.lda$prior

        0         1
0.6721311 0.3278689

> leaf17.lda$scaling

             LD1
logwid 0.1555083
loglen 3.0658277

> leaf17.lda$means

    logwid   loglen
0 1.429422 2.460128
1 1.865537 2.993948
```

The information needed to reconstruct the discriminant function is provided by `leaf17.lda$prior`, `leaf17.lda$means` and `leaf17.lda$scaling`. First we calculate a grand mean, from that the constant term for the discriminant function, and then do a plot (see below) that checks that we are correctly recovering the discriminant function scores. Calculations can be done without matrix multiplication, but are tedious to write down. The following assumes a knowledge of matrix multiplication, for which the symbol is `%*%`:

```
> gmean <- leaf17.lda$prior%*%leaf17.lda$means
> const <- as.numeric(gmean%*%leaf17.lda$scaling)
> z <- as.matrix(leafshape17[,c(5,7)])%*%leaf17.lda$scaling - const
```

Note that R distinguishes between a 1 by 1 matrix and a numeric constant. The final two lines are a check that the discriminant function has been correctly calculated. It has the form $ax + by - c = z$, where the discriminant line is given by $z = 0$. The equation of the line is then $y = -a/bx + c/b$. We have

```
> slope <- -leaf17.lda$scaling[1]/leaf17.lda$scaling[2]
> intercept <- const/leaf17.lda$scaling[2]
```

We now show the plot that checks that we have correctly recovered the discriminant function scores, with the requested plot alongside.

```
> par(mfrow=c(1,2))
> plot(z, leaf17.fit$x[,1]); abline(0,1)
> mtext(side=3, line=1, "Check that z=leaf17.fit$x[,1]")
> plot(loglen ~ logwid, data=leafshape17, xlab="log(leaf width)",
+      ylab="log(leaf length)", pch=leafshape17$arch+1)
> abline(intercept, slope)
> mtext(side=3, line=1, "Fig.12.4B, with discriminant line")
> par(mfrow=c(1,1))
```

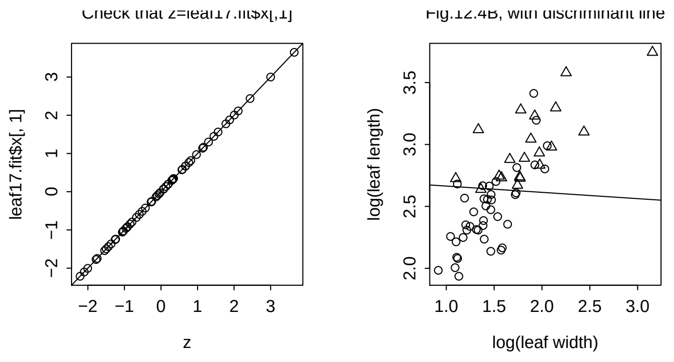

Figure 3: The left panel is a check that calculations are correct. The right panel reproduces Figure 11.4B, adding the discriminant function line.

*Exercise 6**
The data set `leafshape` has three leaf measurements – `bladelen` (blade length), `bladewid` (blade width), and `petiole` (petiole length). These are available for each of two plant architectures, in each of six locations. (The data set `leafshape17` that we encountered in Section 12.2.1 is a subset of the data set `leafshape`.) Use logistic regression to develop an equation for predicting architecture, given leaf dimensions and location. Compare the alternatives: (i) different discriminant functions for different locations; (ii) the same coefficients for the leaf shape variables, but different intercepts for different locations; (iii) the same coefficients for the leaf shape variables, with an intercept that is a linear function of latitude; (iv) the same equation for all locations. Interpret the equation that is finally chosen as discriminant function.

We use the variables `logwid`, `loglen` and `logpet`.

```
> names(leafshape)[4] <- "latitude"
> one.glm <- glm(arch ~ (logwid+loglen+logpet)*location,
+                family=binomial, data=leafshape)
> two.glm <- glm(arch ~ (logwid+loglen+logpet)+location,
+                family=binomial, data=leafshape)
> three.glm <- glm(arch ~ (logwid+loglen+logpet)*latitude,
+                  family=binomial, data=leafshape)
> four.glm <- glm(arch ~ (logwid+loglen+logpet)+latitude,
+                 family=binomial, data=leafshape)
> anova(four.glm, three.glm, two.glm, one.glm)

Analysis of Deviance Table

Model 1: arch ~ (logwid + loglen + logpet) + latitude
Model 2: arch ~ (logwid + loglen + logpet) * latitude
Model 3: arch ~ (logwid + loglen + logpet) + location
Model 4: arch ~ (logwid + loglen + logpet) * location
  Resid. Df Resid. Dev Df Deviance
1       281     193.31
2       278     187.78  3    5.530
3       277     186.30  1    1.481
4       262     148.00 15   38.298
```

It may however, in view of uncertainty about the adequacy of the asymptotic chi-squared approximation for the deviance changes, be better to fit the models using `lda()`, and choose the model that has the smallest cross-validated relative error:

```
> one.lda <- lda(arch ~ (logwid+loglen+logpet)*location, CV=TRUE,
+                data=leafshape)
> two.lda <- lda(arch ~ (logwid+loglen+logpet)+location, CV=TRUE,
+                data=leafshape)
> three.lda <- lda(arch ~ (logwid+loglen+logpet)*latitude, CV=TRUE,
+                  data=leafshape)
> four.lda <- lda(arch ~ (logwid+loglen+logpet)+latitude, CV=TRUE,
+                 data=leafshape)
> table(leafshape$arch, one.lda$class)

      0   1
  0 174  18
  1  24  70

> table(leafshape$arch, two.lda$class)

      0   1
  0 177  15
  1  24  70

> table(leafshape$arch, three.lda$class)

      0   1
  0 179  13
  1  22  72

> table(leafshape$arch, four.lda$class)
```

```
      0   1
  0 177  15
  1  24  70
```

The smallest cross-validated relative error was for the third model.

Additional Exercises A number of additional exercises are included in the laboratory exercises that are available from the web page `http:www.maths.anu.edu.au/~johnm/courses/dm`