Math3346 – Additional Laboratory Exercises

John Maindonald

August 5, 2008

Contents

References

Andrews D F and Herzberg A M, 1985. *Data. A Collection of Problems from Many Fields for the Student and Research Worker.* Springer-Verlag. (pp. 339-353)

Wilkinson, G. N. & Rogers, C. E. 1973. *Symbolic description of models in analysis of variance.* Applied Statistics 22: 392-399.

Part I
Linear Models – with Covariates Entering Linearly

The primary function for fitting linear models is `lm()`, where the `lm` stands for linear model. The power of `lm()` is shown to best effect in an expansive view of linear models. While models must be linear in the parameters, reponses cab be highly non-linear in the explanatory variables. For the present attention will be limited to examples where the explanatory variables ("covariates") enter linearly.

R's implementation of linear models uses a symbolic notation, similar to that described in (Wilkinson & Rogers , 1973), that gives a straightforward powerful means for describing models, including quite complex models. Models are encapsulated in a *model formula*. Model formulae that extend and/or adapt this notation are used in R's modeling functions more generally.

1 Fitting Straight Lines to Data

Exercise 1
In each of the data frames `elastic1` and `elastic2`, fit straight lines that show the dependence of `distance` on `stretch`. Plot the two sets of data, using different colours, on the same graph. Add the two separate fitted lines. Also, fit one line for all the data, and add this to the graph.

Exercise 2
In the data set pressure (*datasets*), the relevant theory is that associated with the Claudius-Clapeyron equation, by which the logarithm of the vapor pressure is approximately inversely proportional to the absolute temperature. Transform the data in the manner suggested by this theoretical relationship, plot the data, fit a regression line, and add the line to the graph. Does the fit seem adequate?
[For further details of the Claudius-Clapeyron equation, search on the internet, or look in a suitable reference text.]

Exercise 3
Run the function `plotIntersalt()`, which plots data from the data frame `intersalt` (*DAAGxtras* package). Data are population average values of blood pressure and of salt in the body as measured from urine samples, from 52 different studies. Is the fitted line reasonable? Or is it a misinterpretation of the data? Suggest alternatives to regression analysis, for getting a sense of how these results should be interpreted? What are the populations where levels are very low? What is special about these countries?
[The function `plotIntersalt()` is available from
`http://www.maths.anu.edu.au/~johnm/r/functions/`]
Enter

```
> webfile <- "http://www.maths.anu.edu.au/~johnm/r/functions/plotIntersalt.RData"
> load(con <- url(webfile))
> close(con)
```

Exercise 4
Install the package *gamair* (from CRAN) and examine the help page for the data frame *hubble*. Type `data(hubble)` to bring the data frame into the workspace. (This is necessary because the `gamair` package differs from most other packages in not using the *lazy loading* mechanism for data sets.)

(a) Plot `y` (Velocity in km sec^{-1}) versus `x` (Distance in Mega-parsec).

(b) Fit a line, omitting the constant term; for this the `lm()` function call is

```
lm(y ~ -1 + x, data=hubble)
```

A Mega-parsec (Mpc) is 3.09×10^{-19} km, so that we need to divide the slope by this amount, so that distances in both cases are in km. The inverse of the slope is then the age of the universe, in seconds. Divide this by $60^2 \times 24 \times 365$ to get an estimate for the age of the earth in years.
[The answer should be around 13×10^9 years.]

(c) Repeat the plot, now using logarithmic scales for both axes. Fit a line, now insisting that the coefficient of `log(x)` is 1.0 (Why?) For this, specify

```
lm(log(y) ~ 1 + offset(log(x)), data=hubble)
```

Add this line to the plot. Again, obtain an estimate of the age of the universe. Does this give a substantially different estimate for the age of the universe?

(d) In each of the straight line fits, on an untransformed scale and using logarithmic scales, do any of the points seem outlers? Investigate the effect of omitting any points that seem to be outliers?

(e) Does either plot, on an untransformed scale or using logarithmic scales, seem to show evidence of curvature?
[See further the note at the end of this set of exercises.]

Exercise 5
A plot of heart weight (`heart`) versus body weight (`weight`), for Cape Fur Seal data in the data set `cfseal` (*DAAG*) shows a relationship that is approximately linear. Check this. However variability about the line increases with increasaing weight. It is better to work with `log(heart)` and `log(weight)`, where the relationship is again close to linear, but variability about the line is more homogeneous. Such a linear relationship is consistent with biological allometry, here across different individuals. Allometric relationships are pairwise linear on a logarithmic scale.

Plot `log(heart)` against `log(weight)`, and fit the least squares regression line for `log(heart)` on `log(weight)`.

```
> library(DAAG)
> cflog <- log(cfseal[, c("heart", "weight")])
> names(cflog) <- c("logheart", "logweight")
> plot(logheart ~ logweight, data=cflog)
> cfseal.lm <- lm(logheart ~ logweight, data=cflog)
> abline(cfseal.lm)
```

Use `model.matrix(cfseal.lm)` to examine the model matrix, and explain the role of its columns in the regression calculations.

2 Multiple Explanatory Variables

Exercise 6
For the data frame `oddbooks` (*DAAG*),

(a) Add a further column that gives the density.

(b) Use the function `pairs()`, or the *lattice* function `splom()`, to display the scatterplot matrix. Which pairs of variables show evidence of a strong relationship?

(c) In each panel of the scatterplot matrix, record the correlation for that panel. (Use `cor()`) to calculate correlations).

(d) Fit the following regression relationships:

(i) `log(weight)` on `log(thick)`, `log(height)` and `log(breadth)`.

(ii) `log(weight)` on `log(thick)` and `0.5*(log(height) + log(breadth))`. What feature of the scatterplot matrix suggests that this might make sense to use this form of equation?

(e) Take whichever of the two forms of equation seems preferable and rewrite it in a form that as far as possible separates effects that arise from changes in the linear dimensions from effects that arise from changes in page density.

[NB: To regress `log(weight)` on `log(thick)` and `0.5*(log(height)+log(breadth))`, the model formula needed is `log(weight) ~ log(thick) + I(0.5*(log(height)+log(breadth)))`
The reason for the use of the wrapper function `I()` is to prevent the parser from giving `*` the special meaning that it would otherwise have in a model formula.]

3 Appendix

3.1 Note re exercise 4

According to the relevant cosmological model, the velocity of recession of any galaxy from any other galaxy has been constant, independent of time. Those parts of the universe that started with the largest velocities of recession from our galaxy have moved furthest, with no change from the velocity just after after time 0. Thus the time from the beginning should be s/v, where s is distance, and v is velocity. The slope of the least squares line gives a combined estimate, taken over all the galaxies included in the data frame gamair. More recent data suggests, in fact, that the velocity of recession is not strictly proportional to distance.

Part II
Extending the Linear Model

Ideas that will be important in the expansive view of linear models that will now be illustrated include: *basis function*, *factor*, and *interaction*. The reach of R's *model formulae* is wide.

1 A One-way Classification – Eggs in the Cuckoo's Nest

This demonstrates the use of linear models to fit qualitative effects.

Like many of nature's creatures, cuckoos have a nasty habit. They lay their eggs in the nests of other birds. First, let's see how egg length changes with the host species. This will use the graphics function `stripplot()` from the *lattice* package. The data frame `cuckoos` is in the *DAAG* package.

```
> par(mfrow=c(1,2))
> library(DAAG)
> library(lattice)
> names(cuckoos)[1] <- "length"
> table(cuckoos$species)

hedge.sparrow  meadow.pipit  pied.wagtail         robin    tree.pipit
           14            45            15            16            15
         wren
           15

> stripplot(species ~ length, data=cuckoos)
> ## Look also at the relationship between length and breadth;
> ## is it the same for all species?
> cuckoo.strip <- stripplot(breadth ~ length, groups=species,
+                           data=cuckoos, auto.key=list(columns=3))
> print(cuckoo.strip)
> par(mfrow=c(1,1))
```

Exercise 1
Now estimate the means and standard deviations for each of the groups, using direct calculation:

```
> with(cuckoos, sapply(split(length, species), mean))

hedge.sparrow  meadow.pipit  pied.wagtail         robin    tree.pipit
        23.11         22.29         22.89         22.56         23.08
         wren
        21.12

> with(cuckoos, sapply(split(length, species), sd))

hedge.sparrow  meadow.pipit  pied.wagtail         robin    tree.pipit
       1.0494        0.9196        1.0723        0.6821        0.8801
         wren
       0.7542
```

[The function `split()` splits the lengths into six sublists, one for each species. The function `sapply()` applies the function calculation to the vectors of lengths in each separate sublist.]
Check that the SD seems smallest for those species where the SD seems, visually, to be smallest.

Exercise 2
Obtain, for each species, the standard error of the mean. This is obtained by dividing the standard deviation by the square root of the number of values:

```
> sdev <- with(cuckoos, sapply(split(length, species), sd))
> n <- with(cuckoos, sapply(split(length, species), length))
> sdev/sqrt(n)

hedge.sparrow  meadow.pipit  pied.wagtail         robin    tree.pipit
       0.2805        0.1371        0.2769        0.1705        0.2272
         wren
       0.1947
```

Exercise 3
Now estimate obtain the means for each of the groups froam a model that fits a separate constant term for each species. The model can be specified using several different, but equivalent, formulations, or parameterizations.

- The following the species means directly, as parameters for the model. It forces all parameters to be species means

  ```
  > lm(length ~ -1 + species, data=cuckoos)
  ```

- In the following alternative parameterization, the first parameter estimate is the mean for the first species, while later estimates are differences from the first species. In other words, the first species becomes the baseline.

  ```
  > lm(length ~ species, data=cuckoos)
  ```

Now answer the following

(a) Use the function `fitted()` to calculate the fitted values in each case, and check that they are the same. Reconcile the two sets of results.

(b) Examine and interpret the model matrices for the two different parameterizations.

(c) Use the `termplot()` function to show the effects of the different factor levels. Be sure to call the function with `partial.resid=TRUE` and with `se=TRUE`. Does the variability seem similar for all host species?

Exercise 4
The following creates (in the first line) and uses (second line) a function that calculates the standard error of the mean.

```
> se <- function(x)sd(x)/sqrt(length(x))
> with(cuckoos, sapply(split(length, species), se))

hedge.sparrow  meadow.pipit  pied.wagtail         robin    tree.pipit
       0.2805        0.1371        0.2769        0.1705        0.2272
         wren
       0.1947
```

Exercise 4, continued
The standard error calculation can be done in a single line of code, without formally creating the function `se()`. Instead, the function definition can be inserted as an anonymous function, so that it does not need a name. The function definition is inserted where the function name would otherwise appear:

```
> with(cuckoos, sapply(split(length, species),
+                      function(x)sd(x)/sqrt(length(x))))

hedge.sparrow  meadow.pipit  pied.wagtail         robin    tree.pipit
       0.2805        0.1371        0.2769        0.1705        0.2272
         wren
       0.1947
```

2 Regression Splines – one explanatory variable

This pursues the use of smoothing methods, here formally within a regression context.

Exercise 5
The following is based on the `fruitohms` data frame (in *DAAG*)

(a) Plot `ohms` against `juice`.

(b) Try the following:

```
> plot(ohms ~ juice, data=fruitohms)
> with(fruitohms, lines(lowess(juice, ohms)))
> with(fruitohms, lines(lowess(juice, ohms, f=0.2), col="red"))
```

Which of the two fitted curves best captures the pattern of change?

(c) Now fit a natural regression spline. Try for example:

```
> library(splines)
> plot(ohms ~ juice, data=fruitohms)
> hat <- with(fruitohms, fitted(lm(ohms ~ ns(juice, 4))))
> with(fruitohms, lines(juice,hat, col=3))
> ## Check the locations of the internaal knots.
> attributes(ns(fruitohms$juice, 4))$knots

  25%   50%   75%
18.62 41.00 48.00
```

Experiment with different choices for the number of degrees of freedom. How many degrees of freedom seem needed to adequately capture the pattern of change? Plot the spline basis functions. Add vertical lines to the plot that show the knot locations.

Exercise 6
In the `geophones` data frame (*DAAG*), plot `thickness` against distance. Use regression splines, as in Exercise 6, to fit a suitable curve through the data. How many degrees of freedom seem needed? Add vertical lines to the plot that show the locatons of the knots.

3 Regression Splines – Two or More Explanatory Variables

We begin by using the `hills2000` data (*DAAG* version 0.84 or later) to fit a model that is linear in `log(dist)` and `log(climb)`, thus

```
> lhills2k <- log(hills2000[, 2:4])
> names(lhills2k) <- c("ldist",  "lclimb", "ltime")
> lhills2k.lm <- lm(ltime ~ ldist + lclimb, data = lhills2k)
```

Use `termplot()` to check departures from linearity in `lhills2k.lm`. Note whether there seem to be any evident outliers.

Exercise 7
Now use regression splines to take out the curvature that was evident when `ltime` was modeled as a linear function of `ldist` and `lclimb`. Use `termplot()` to guide your choice of degrees of freedom for the spline bases. For example, you might try

```
> lhills2k.ns <- lm(ltime ~ ns(ldist,2) + lclimb, data = lhills2k)
```

Again, examine the diagnostic plots? Are there any points that should perhaps be regarded as outliers?
Does a normal spline of degree 2 in `ldist` seem to give any benefit above a polynomial of degree 2.

Note: The coefficients of the spline basis terms do not give useful information on what degree of spline curve is required. See exercise 9 below. If one fits a spline of degree 3 to a relationship that is esentially linear, all three coefficients are likely to be highly significant. Rather, check how the residual sum of squares changes as the number of degrees of freedom for the spline curve increases. [F-tests are not strictly valid, as successive models are not nested (the basis functions change), but they may be a helpful guide.]

Exercise 8
The *MASS* package has the function `lqs()` that can be used for a *resistant* regression fit, i.e., the effect of outlying points is attenuated. Try the following

```
> library(MASS)
> lhills2k.lqs <- lqs(ltime ~ ns(ldist,2) + lclimb, data = lhills2k)
> plot(resid(lhills2k.lqs) ~ fitted(lhills2k.lqs))
> big4 <- order(abs(resid(lhills2k.lqs)), decreasing=TRUE)[1:4]
> text(resid(lhills2k.lqs)[big4] ~ fitted(lhills2k.lqs)[big4],
+      labels=rownames(lhills2k)[big4], pos=4)
```

Try the plot without the two largest outliers. Does this make any difference of consequence to the fitted values?

Exercise 10
Try the following:

```
x <- 11:20
y <-  5 + 1.25*x+rnorm(10)
summary(lm(y ~ ns(x,2)))$coef
summary(lm(y ~ ns(x,3)))$coef
summary(lm(y ~ ns(x,4)))$coef
```

Note that the coefficents are in all cases very mcuh larger than their standard errors. It takes both degree 2 spline basis terms, additional to the constant, to fit a line. All three degree 3 terms are required, and so on! Splines do not give a parsimonious representation, if the form of the model is known.

4 Errors in Variables

Exercise 10
Run the accompanying function `errorsINx()` several times. Comment on the results. The underlying relationship between y and x is the same in all cases. The error in x is varied, from values of x that are exact to values of x that have random errors added with a variance that is twice that of the variance of x.

4.1 Function used

This is available from `http://www.maths.anu.edu.au/~johnm/r/functions/`

```
"errorsINx" <-
  function(mu = 8.25, n = 100, a = 5, b = 2.5, SDx=1, sigma = 2,
           timesSDx=(1:5)/2.5){
    mat <- matrix(0, nrow=n, ncol=length(timesSDx)+2)

    x0 <- mu*exp(rnorm(n,0,SDx/mu))/exp(0)

    y <- a + b*x0+rnorm(n,0,sigma)
    mat[, length(timesSDx)+2] <- y
    mat[,2] <- x0
    mat[,1] <- y
    sx <- sd(x0)
    k <- 2
    for(i in timesSDx){
      k <- k+1
      xWITHerror <- x0+rnorm(n, 0, sx*i)
      mat[, k] <- xWITHerror
    }
    df <- as.data.frame(mat)
    names(df) <- c("y", "x", paste("x",timesSDx,sep=""))
    df
  }

## Now use function to simulate y vs x relationships, with several
## different values of timesSDx, which specifies the ratio of the
## errors in x variance to SD[x]
    oldpar <- par(mar=c(3.6,3.1,1.6,0.6), mgp=c(2.5,0.75,0),
                  oma=c(1,1,0.6,1),
                  mfrow=c(2,3), pty="s")
    mu <- 20; n <- 100; a <- 15; b <- 2.5; sigma <- 12.5; timesSigma<-(1:5)/2.5
    mat <- errorsINx(mu = 20, n = 100, a = 15, b = 2.5, sigma = 5,
                     timesSDx=(1:5)/2.5)
    beta <- numeric(dim(mat)[2]-1)
    sx <- sd(mat[,2])
    y <- mat[, 1]
    for(j in 1:length(beta)){
      xj <- mat[,j+1]
      plot(y ~ xj, xlab="", ylab="", col="gray30")
      if(j==1)
        mtext(side=3, line=0.5, "No error in x") else{
          xm <- timesSigma[j-1]
          mtext(side=3, line=0.5, substitute(tau == xm*s[z], list(xm=xm)))
```

```
        }
      if(j>=4)mtext(side=1, line=2, "x")
      if(j%%3 == 1)mtext(side=2, line=2, "y")
      errors.lm <- lm(y ~ xj)
      abline(errors.lm)
      beta[j] <- coef(errors.lm)[2]
      bigsigma <- summary(errors.lm)$sigma
      print(bigsigma/sigma)
      abline(a, b, lty=2)
    }
    print(round(beta, 3))

plotIntersalt <-
function (dset = intersalt1, figno = 2)
{
    oldpar <- par(oma = c(6.5, 0, 0, 0), mar = par()$mar - c(0,
        0, 3.5, 0))
    on.exit(par(oldpar))
    lowna <- c(4, 5, 24, 28)
    plot(dset$na, dset$bp, pch = 15, ylim = c(50, 85),
         xlab = "Median sodium excretion (mmol/24hr)",
         ylab = "Median diastolic BP (mm Hg)", type = "n")
    points(dset$na[-lowna], dset$bp[-lowna], pch = 16)
    points(dset$na[lowna], dset$bp[lowna], pch = 1, lwd = 3)
    u <- lm(bp ~ na, data = dset)
    abline(u$coef[1], u$coef[2])

    figtxt <-
       paste("Fig. ", figno, ": Plot of median blood pressure versus salt",
             "\\n(measured by sodium excretion) for 52 human",
             "\\npopulations. Four results (open circles) are for",
             "\\nnon-industrialised societies with very low salt intake,",
             "\\nwhile other results are for industrialised societies.",
       sep = "")
    mtext(side = 1, line = 6, figtxt, cex = 1.1, adj = 0, at = -20)
}
```

Part III
Multi-level Models

For background, see SPDM: Sections 14-15.

1 Description and Display of the Data

1.1 Description

This laboratory will work with data on corn yields from the Caribbean islands of Antigua and St Vincent. Data are yields from packages on eight sites on the Caribbean island of Antigua. They are a summarized version of a subset of data given in Andrews and Herzberg (1985). The data frames `ant111b` and `vince111b` hold yields for the standard treatment, here identified as `111`, for sites on Antigua and St Vincent respectively. Additionally, there will be some use of the more extensive data in the data frame `antigua`. All three data frames are in recent versions (≥ 0.84) of the *DAAG* package.

The data frame `ant111b` has data for n=4 packages of land at each of eight sites, while `vince111b` data for four packages at each of nine sites. As will be described below, two possible predictions are:

(a) Predictions for new packages of land in one of the existing sites.

(b) Predictions for new packages in a new site.

The accuracies for the second type of prediction may be much less accurate than for the first type. A major purpose of this laboratory is to show how such differences in accuracy can be modeled.

1.2 Display

We begin by examining plots, for the treatment `111`, from the combined data for the two islands. This information for the separate islands is summarized in the datasets `ant111b` and `vince111b` in the *DAAG* package.

A first step is to combine common columns of `ant111b` and `vince111b` into the single data frame `corn111b`.

```
> library(lattice)
> library(DAAG)
> corn111b <- rbind(ant111b[,-8], vince111b)
> corn111b$island <- c("Antigua", "StVincent")[corn111b$island]
```

- The following plot uses different panels for the two islands:

```
> corn.strip1 <- stripplot(site ~ harvwt | island, data = corn111b,
+                          xlab="Harvest weight")
```

- The following plot uses different panels for the two islands, but allows separate ("free" = no relation) vertical scales for the two plots.

```
> corn.strip2 <- stripplot(site ~ harvwt | island, data = corn111b,
+                          xlab="Harvest weight",
+                          scale=list(y=list(relation="free")))
```

- The following uses a single panel, but uses different colours (or, on a black and white device, different symbols) to distinguish the two islands. Notice the use of `auto.key` to generate an automatic key:

```
> corn.strip3 <- stripplot(site ~ harvwt, data = corn111b, groups=island,
+                          xlab="Harvest weight", auto.key=list(columns=2))
```

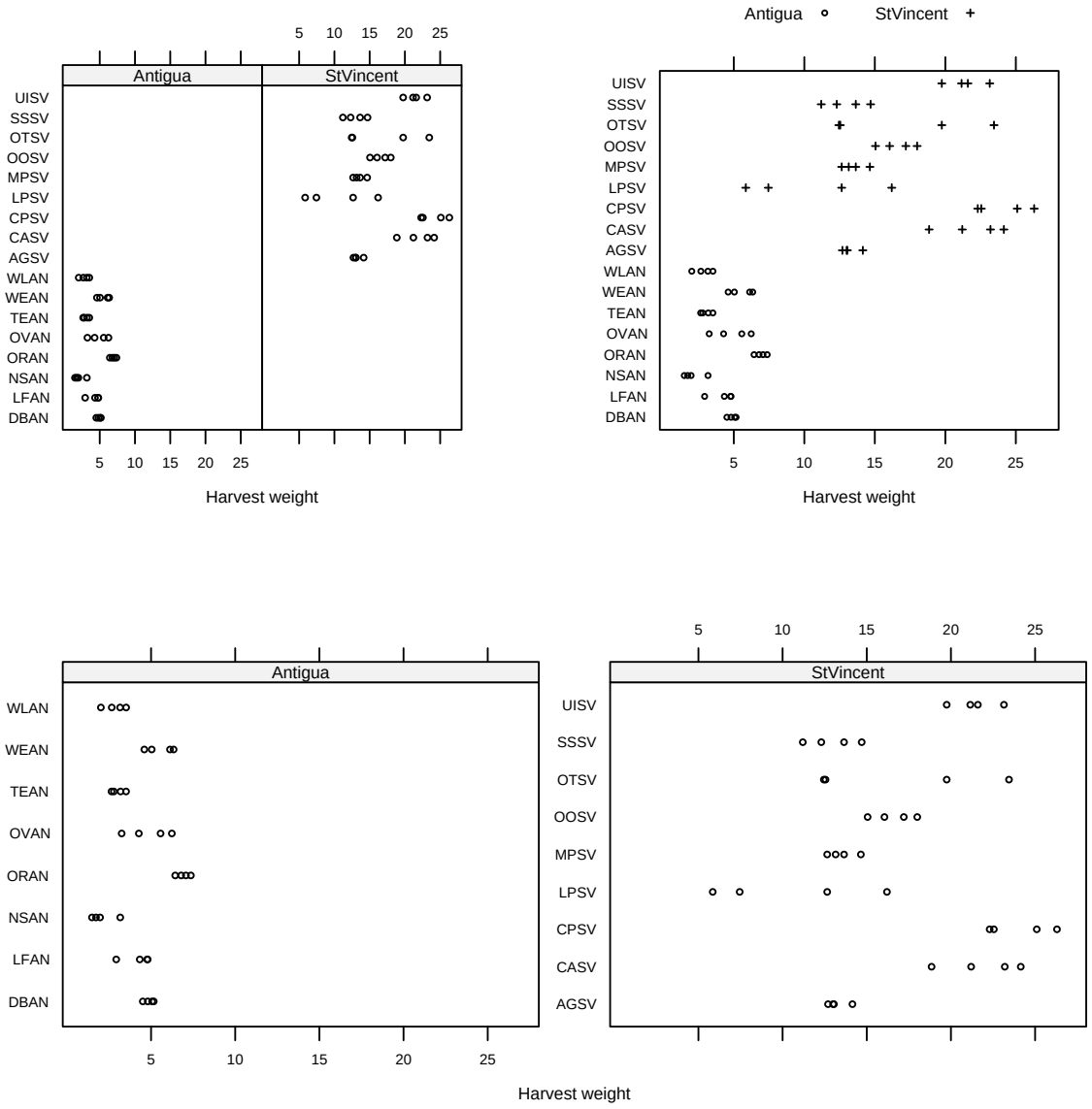

Figure 1: Yields for the four packages of corn on sites on the islands of Antigua and St Vincent.

Next, we will obtain package means for the Antiguan data, for all treatments.

```
> with(antigua,        # antigua is from the DAAG package
+      antp <<- aggregate(harvwt, by = list(site = site,
+                                           package = block,
+                                           trt = trt),
+                         FUN = mean)
+ )
> names(antp)[4] <- "harvwt"
```

Notice the use of the version `<<-` of the assignment symbol to ensure that assignment takes place in the workspace.

Now plot mean harvest weights for each treatment, by site:

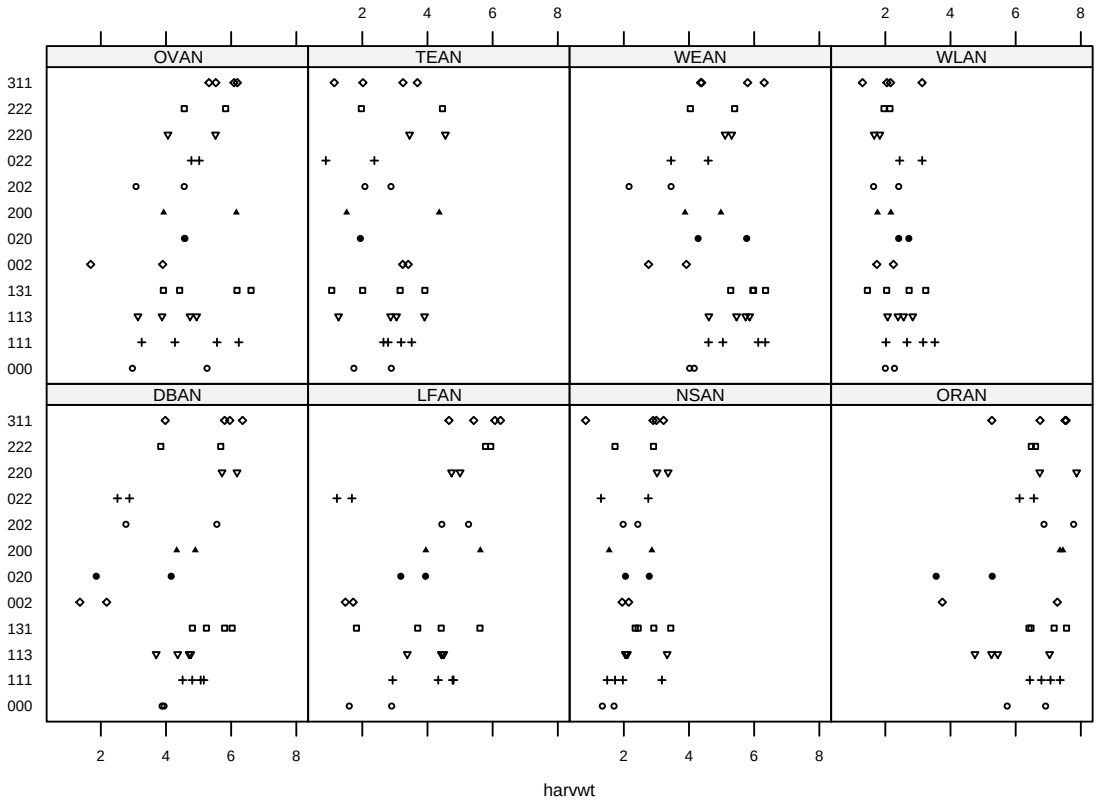

Figure 2: Yields for the four packages of corn on each of eight sites on the island of Antigua.

Questions and Exercises

(a) Which set of sites (Antigua or St Vincent) shows the largest yields?

(b) Create a plot that compares the logarithms of the yields, within and between sites on the two islands. From this plot, what, if anything, can you say about the differet variabilities in yield, within and between sites on each island?

2 Multi-level Modeling

***Analysis using *lme*:** The modeling command takes the form:

```
> library(nlme)
> ant111b.lme <- lme(fixed=harvwt ~ 1, random = ~1|site,
+                    data=ant111b)
```

The only fixed effect is the overall mean. The argument `random = ~1|site` fits random variation between sites. Variation between the individual units that are nested within sites, i.e., between packages, are by default treated as random. Here is the default output:

```
> options(digits=4)
> ant111b.lme
```

```
Linear mixed-effects model fit by REML
  Data: ant111b
```

```
  Log-restricted-likelihood: -47.21
  Fixed: harvwt ~ 1
(Intercept)
      4.292

Random effects:
 Formula: ~1 | site
        (Intercept) Residual
StdDev:       1.539     0.76

Number of Observations: 32
Number of Groups: 8
```

Notice that *lme* gives, not the components of variance, but the standard deviations (`StdDev`) which are their square roots. Observe that, according to *lme*, $\widehat{\sigma_B^2} = 0.76^2 = 0.578$, and $\widehat{\sigma_L^2} = 1.539^2 = 2.369$. The variance for an individual package is $\widehat{\sigma_B^2} + \widehat{\sigma_L^2}$.
Those who are familiar with an analysis of variance table for such data should note that *lme* does not give the mean square at any level higher than level 0, not even in this balanced case.

Note that the yields are not independent between different packages on the same site, in the population that has packages from multiple sites. (Conditional on coming from one particular site, package yields are however independent.)

The take-home message from this analysis is:

- For prediction for a new package at one of the existing sites, the standard error is 0.76
- For prediction for a new package at a new site, the standard error is $\sqrt{1.539^2 + .76^2} = 1.72$
- For prediction of the mean of n packages at a new site, the standard error is $\sqrt{1.539^2 + 0.76^2/n}$. This is NOT inversely proportional to n, as would happen if the yields were independent within sites.

Where there are multiple levels of variation, the predictive accuracy can be dramatically different, depending on what is to be predicted. Similar issues are arise in repeated measures contexts, and in time series. Repeated measures data has multiple profiles, i.e., many small time series.

2.1 Simulation

The following function simulates results from a multilevel model for the case where there are `npackages` packages at each of `nplots` plots.

```
> "simMlevel" <-
+   function(nsites=8, npackages=4, mu=4, sigmaL=1.54, sigmaB=0.76){
+     facSites <- factor(1:nsites)
+     facPackages <- factor(1:npackages)
+     dframe <- expand.grid(facPackages=facPackages, facSites=facSites)
+     nall <- nsites*npackages
+     siteEffects <- rnorm(nsites, 0, sigmaL)
+     err <- rnorm(nall, 0, sigmaB) + siteEffects[unclass(dframe$facSites)]
+     dframe$yield <- mu+err
+     dframe
+   }
```

The default arguments are `sigmaB` = 0.76 and `sigma` = 1.54, as for the Antiguan data.

2.2 Questions and Exercises

(a) Repeat the analysis

  (a) for the Antiguan data, now using a logarithmic scale.

  (b) for the St Vincent data, using a logarithmic scale.

(b) Overlay plots, for each of the two islands, that show how the variance of the mean can be expected to change with the number of packages n.

(c) Are there evident differences between islands in the contributions of the two components of variance? What are the practical implications that flow from such differences as you may observe?

(d) Use the function `simMlevel()` to simulate a new set of data, using the default arguments. Analyse the simulated data. Repeat this exercise 25 or more times. How closely do you reproduce the values of `sigmaL`=1.54 and `sigmaB`=0.76 that were used for the simulation?

3 Multi-level Modeling – Attitudes to Science Data

These data are from in the *DAAG* package for R. The data are measurements of attitudes to science, from a survey where there were results from 20 classes in 12 private schools and 46 classes in 29 public (i.e. state) schools, all in and around Canberra, Australia. Results are from a total of 1385 year 7 students. The variable `like` is a summary score based on two of the questions. It is on a scale from 1 (dislike) to 12 (like). The number in each class from whom scores were available ranged from 3 to 50, with a median of 21.5.

There are three variance components:

```
Between schools 0.00105
Between classes 0.318
Between students 3.05
```

The between schools component can be neglected. The variance for a class mean is $0.318 + 3.05/n$, where n is the size of the class. The two contributions are about equal when n =10.

4 *Additional Calculations

We return again to the corn yield data.

Is variability between packages similar at all sites?:

```
> if(dev.cur()==2)invisible(dev.set(3))
> vars <- sapply(split(ant111b$harvwt, ant111b$site), var)
> vars <- vars/mean(vars)  # Standardize to a variance of one
> qqplot(qchisq(ppoints(vars),3), 3*vars)
```

Does variation within sites follow a normal distribution?:

```
> qqnorm(residuals(ant111b.lme))
```

What is the pattern of variation between sites?

```
> locmean <- sapply(split(log(ant111b$harvwt), ant111b$site), mean)
> qqnorm(locmean)
```

The distribution seems remarkably close to normal.

Fitted values and residuals in *lme*: By default fitted values account for all random effects, except those at level 0. In the example under discussion `fitted(ant111b.lme)` calculates fitted values at level 1, which can be regarded as estimates of the site means. They are not however the site means, as the graph given by the following calculation demonstrates:

```
> hat.lm <- fitted(lm(harvwt ~ site, data=ant111b))
> hat.lme <- fitted(ant111b.lme)   # By default, level=1)
> plot(hat.lme ~ hat.lm, xlab="Site means",
+      ylab="Fitted values (BLUPS) from lme")
> abline(0,1,col="red")
```

The fitted values are known as BLUPs (Best Linear Unbiased Predictors). Relative to the site means, they are pulled in toward the overall mean. The most extreme site means will on average, because of random variation, be more extreme than the corresponding "true" means for those sites. There is a theoretical result that gives the factor by which they should be shrunk in towards the true mean.

Residuals are by default the residuals from the package means, i.e., they are residuals from the fitted values at the highest level available. To get fitted values and residuals at level 0, enter:

```
> hat0.lme <- fitted(ant111b.lme, level=0)
> res0.lme <- resid(ant111b.lme, level=0)
> plot(res0.lme, ant111b$harvwt - hat0.lme)  # Points lie on y=x
```

5 Notes – Other Forms of Complex Error Structure

Time series are another important special case. A first step is, often, to subtract off any trend, and base further analysis on residuals about this trend. Observations that are close together in time are typically more closely correlated than observations that are widely separated in time.

The variances of the mean of n observations with variance σ^2 will, assuming that positive correlation between neighbouring observations makes the major contribution to the correlation structure, be greater than $\frac{\sigma^2}{n}$.

Here is a simple way to generate data that are sequentially correlated. The autocorrelation plot shows how the estimated correlation changes as observations move further apart.

```
> y <- rnorm(200)
> y1 <- y[-1] + 0.5*y[-length(y)]
> acf(y1)
```

Of course the multiplier in `y1 <- y[-1] + 0.5*y[-1000]` can be any number at all, and more complex correlation structures can be generated by incorporating further lags of `y`.