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

Preliminaries

```
> library(DAAG)
```

Exercise 1

The data set `cities` lists the populations (in thousands) of Canada's largest cities over 1992 to 1996. There is a division between Ontario and the West (the so-called "have" regions) and other regions of the country (the "have-not" regions) that show less rapid growth. To identify the "have" cities we can specify

```
cities$have <- factor((cities$REGION=="ON")|
                      (cities$REGION=="WEST"))
```

Plot the 1996 population against the 1992 population, using different colors to distinguish the two categories of city, both using the raw data and taking logarithms of data values, thus:

```
plot(POP1996 ~ POP1992, data=cities,
     col=as.integer(cities$have))
plot(log(POP1996) ~ log(POP1992), data=cities,
     col=as.integer(cities$have))
```

Which of these plots is preferable? Explain.
Now carry out the regressions

```
cities.lm1 <- lm(POP1996 ~ have+POP1992, data=cities)
cities.lm2 <- lm(log(POP1996) ~ have+log(POP1992),
                 data=cities)
```

and examine diagnostic plots. Which of these seems preferable? Interpret the results.

The required plots are given below.

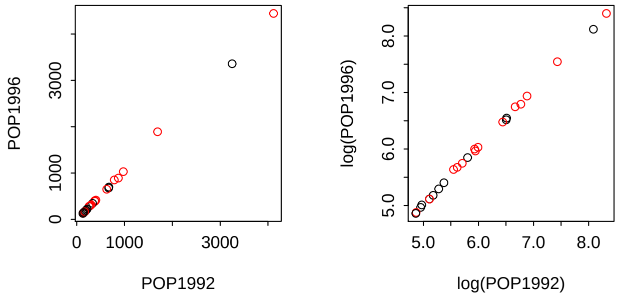

Figure 1: Red circles indicate the 'have' cities, and black circles indicate the 'have-not' cities. In the left panel, data are untransformed, while the right panel uses logarithmic scales.

The second plot is preferable, since it spreads the plotted points out more evenly, while the first plot contains the large cluster of points in one corner. Population comparisons are usually best made using ratios instead of differences; differences of logarithms correspond to logarithms of ratios, which is another reason for preferring the second plot.

We plot residuals against fitted values, first for the untransformed data and then for the transformed data.

```
> par(mfrow=c(1,2))
> cities.lm1 <- lm(POP1996 ~ have+POP1992, data=cities)
> cities.lm2 <- lm(log(POP1996) ~ have+log(POP1992),
+                  data=cities)
> plot(cities.lm1, which=1)
> plot(cities.lm2, which=1)
> par(mfrow=c(1,1))
```

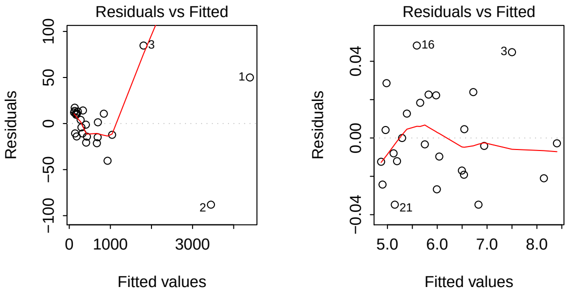

Figure 2: Plots of residuals against fitted values. The left panel is for the model that used untransformed data, while the right panel is for the model that used log-transformed data.

These plots indicate the need for transformation.

It is also a good idea to check plots of the residuals versus the predictors, as in

```
plot(resid(cities.lm2) ~ log(cities$POP1992))
plot(resid(cities.lm2) ~ cities$have)
```

These plots (not shown) and plots of Cook's distance and normal probability plots (also not shown) do not indicate any problems.

Here is the regression summary:

```
> summary(cities.lm2)

Call:
lm(formula = log(POP1996) ~ have + log(POP1992), data = cities)

Residuals:
     Min       1Q   Median       3Q      Max
-0.03478 -0.01698 -0.00332  0.01836  0.04821

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.05565    0.03062   -1.82    0.083
haveTRUE      0.02254    0.01004    2.25    0.035
log(POP1992)  1.01352    0.00523  193.92   <2e-16

Residual standard error: 0.0239 on 22 degrees of freedom
Multiple R-squared: 0.999,        Adjusted R-squared: 0.999
F-statistic: 2.05e+04 on 2 and 22 DF,  p-value: <2e-16
```

This suggests that the 'have' cities grew faster between 1992 and 1996 than the 'have-not' cities.

Exercise 2
In the data set `cement` (*MASS* package), examine the dependence of `y` (amount of heat produced) on `x1`, `x2`, `x3` and `x4` (which are proportions of four constituents). Begin by examining the scatterplot matrix. As the explanatory variables are proportions, do they require transformation, perhaps by taking $\log(x/(100 - x))$? What alternative strategies might be useful for finding an equation for predicting heat?

First, obtain the scatterplot matrix for the untransformed cement data:

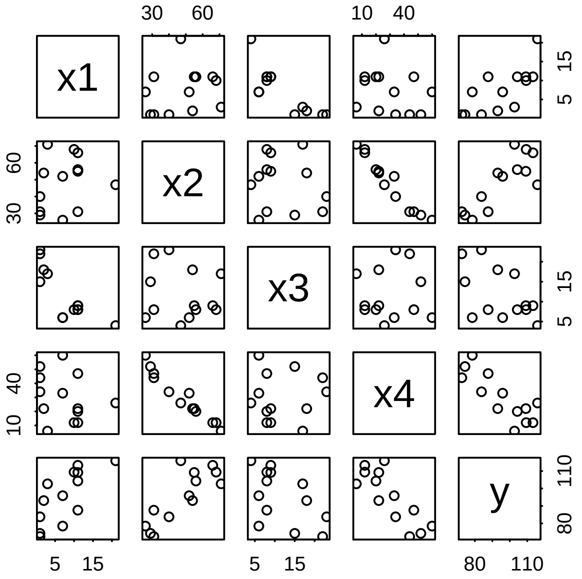

Figure 3: Scatterplot matrix for the cement data.

Since the explanatory variables are proportions, a transformation such as that suggested might be helpful, though the bigger issue is the fact that the sum of the explanatory variables is nearly constant. Thus, there will be severe multicollinearity as indicated by the variance inflation factors:

```
> cement.lm <- lm(y ~ x1+x2+x3+x4, data=cement)
> vif(cement.lm)

    x1     x2     x3     x4
 38.50 254.42  46.87 282.51
```

The scatterplot matrix indicated that `x4` and `x2` are highly correlated, so we may wish to include just one of these variables as in

```
> cement.lm2 <- lm(y ~ x1+x2+x3, data=cement)
> vif(cement.lm2)

   x1    x2    x3
3.251 1.064 3.142
```

The multicollinearity is less severe, and we can proceed. We consult the standard diagnostics using

```
> par(mfrow=c(1,4))
> plot(cement.lm2)
> par(mfrow=c(1,1))
```

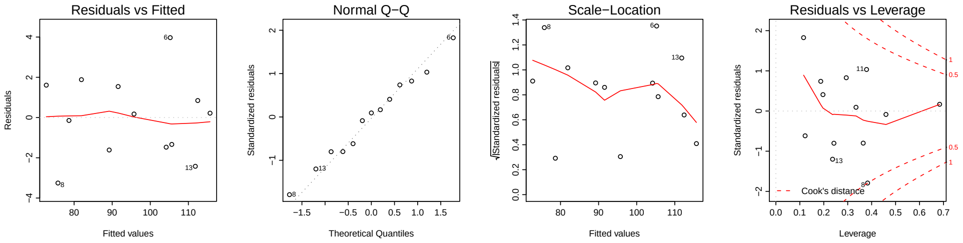

Figure 4: Diagnostic plots for the model cement.lm2

Nothing seems amiss on these plots. The three variable model seems satisfactory. Upon looking at the summary, one might argue in favour of removing the variable x3.

For the logit analysis, first define the logit function:

```
> logit <- function(x) log(x/(100-x))
```

Now form the transformed data frame, and show the scatterplot matrix:

```
> logitcement <- data.frame(logit(cement[,c("x1", "x2","x3","x4")]),
+       y=cement[, "y"])
> pairs(logitcement)
```

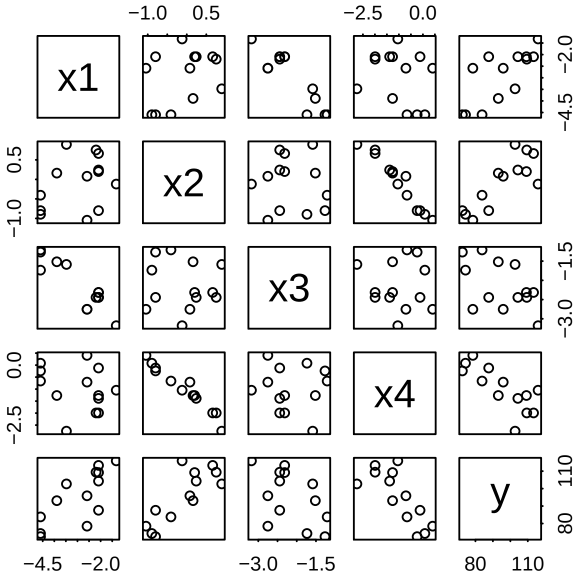

Figure 5: Scatterplot matrix for the logits of the proportions.

Notice that the relationship between x2 and x4 is now more nearly linear. This is helpful; it is advantageous for collinearities or multicollinearities to be explicit.

Now fit the full model, and plot the diagnostics:

```
> logitcement.lm <- lm(y ~ x1+x2+x3+x4, data=logitcement)
> par(mfrow=c(1,4))
> plot(logitcement.lm)
> par(mfrow=c(1,1))
```

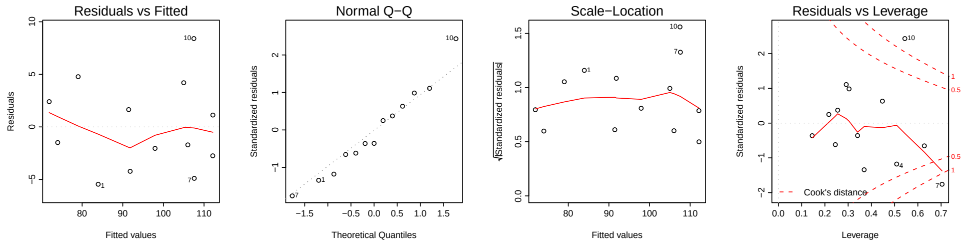

Figure 6: Diagnostic plots for the model that works with logits.

This time, the multicollinearity problem is less extreme, though it is still notable. Some observations have now influential outliers. In this problem, we may be best off not transforming the predictors.

Exercise 3
The data frame hills2000 in our *DAAG* package has data, based on information from the Scottish Running Resource web site, that updates the 1984 information in the data set hills. Fit a regression model, for men and women separately, based on the data in hills2000. Check whether it fits satisfactorily over the whole range of race times. Compare the equation that you obtain with that based on the hills data frame.

```
> hills2K <- hills2000[, -seq(1,6)]
```

This eliminates the first 6 columns which contain the record times for both sexes in terms of hours, minutes and seconds; these columns are extraneous, since we have the record times in seconds in the time and timef variables.

We begin with the same kind of transformed model that we tried in Section 6.3 for the hills data, examining the diagnostic plots.

```
> hills2K.loglm <- lm(log(time) ~ log(dist) + log(climb), data=hills2K)
> par(mfrow=c(1,4))
> plot(hills2K.loglm)
> par(mfrow=c(1,1))
```

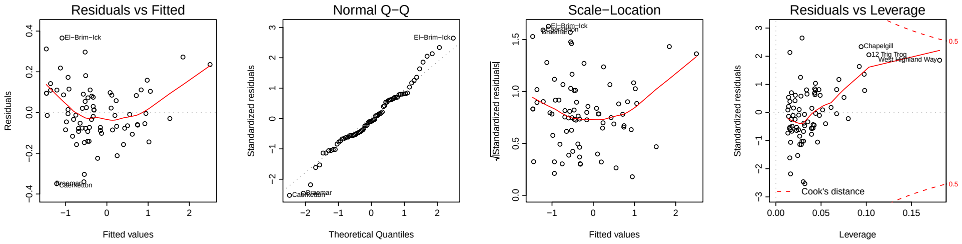

Figure 7: Diagnostic plots for hills2K.loglm

The first of the diagnostic plots (residuals versus fitted values) reveals three potential outliers, identified as 12 Trig Trog, Chapelgill, and Caerketton. A robust fit is however a safer guide. The plot from such a fit shows Eildon Two and Braemar as outliers. El-Brim-Ick stands out as different primarily because there is residual curvature in the plot.

```
> use <- !row.names(hills2K)%in%c("Eildon Two","Braemar")
> hills2Kr.loglm <- lm(log(time) ~ log(dist) + log(climb), data=hills2K[use, ])
> plot(hills2Kr.loglm, panel=panel.smooth, which=1)
```

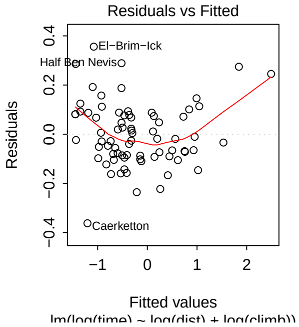

Figure 8: Residuals vs fitted values for hills2Kr.loglm

There is clear evidence of curvature in the plot of residuals. Caerketton now stands out. We will omit that also, for the time being.

Is it hepful to add the interaction term `log(dist):log(climb)`. It turns out that this does not remove the curvature in the plot of residuals versus fitted values.

Additional Note:

A model that uses spline curves to transform the explanatory variables does work well. We include residuals and fitted values for the three omitted races in the plot. The code is

```
> library(splines)
> use <- !row.names(hills2K)%in%c("Eildon Two","Braemar","Caerketton")
> hills2K.bs <- lm(log(time) ~ bs(dist,4)+bs(climb,4), data=hills2K[use, ])
> hat <- predict(hills2K.bs, newdata=hills2K)
> res <- log(hills2K$time)-hat
> par(mfrow=c(1,3), pty="s")
> plot(hat,res)
> text(hat[!use], res[!use], row.names(hills2K)[!use], pos=4)
> plot(hills2K.bs, panel=panel.smooth, which=1)
> termplot(hills2K.bs, partial.resid=TRUE)
> par(mfrow=c(1,1))
```

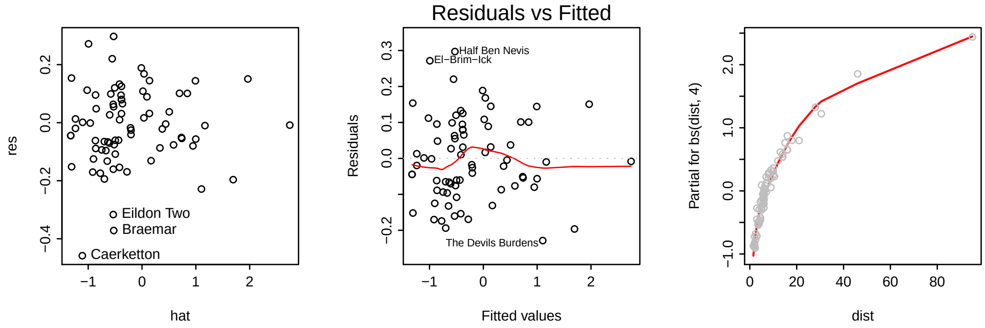

Figure 9: Residuals vs fitted values, and termplots, for hills2K.bs.

The plot of residuals versus fitted values shows no evidence either of trend or of heterogeneity of variance. Caerketton shows the clearest evidence that is should perhaps be identified as an outlier.

To complete the analysis, check the effect of including back in the model (i) all three omitted points except Caerketton, and (ii) all three omitted points. If it makes little difference, they should be included back.

(A further model that may be tried has `time` on the left-hand side. The plot of residuals against fitted values then shows clear evidence of curvature.)

Additional Note: The following may be interesting. We use the spline model, derived from the `hills2K` data, to determine predicted values, and compare these with predicted values from the spline model that is fitted to the `hills` data.

```
> hills2K.bs <- lm(log(time) ~ bs(dist,4)+bs(climb,4), data=hills2K[use, ])
> hills.bs <- lm(log(time) ~ bs(dist,4)+bs(climb,4), data=hills[-18, ])
> fits <- predict(hills.bs)
> fits2 <- predict(hills2K.bs, newdata=hills[-18,])
> plot(fits, fits2, xlab="Fitted values, from hills.bs",
+       ylab="Fitted values, hills2K.bs model")
> mtext(side=3, line=1, "All fitted values are for the hills data")
> abline(0,1)
```

The warnings arise because some values of `climb` for the `hills` data lie outside of the range of this variable for the `hills2K` data.

Exercise 4
Section 6.1 used `lm()` to analyze the `allbacks` data that are presented in Figure 6.1. Repeat the analysis using (1) the function `rlm()` in the *MASS* package, and (2) the function `lqs()` in the *MASS* package. Compare the two sets of results with the results in Section 6.1.

Here are fits, w/wo intercept, using `rlm()`

```
> allbacks.rlm <- rlm(weight ~ volume+area, data=allbacks)
> summary(allbacks.rlm)

Call: rlm(formula = weight ~ volume + area, data = allbacks)
Residuals:
   Min     1Q Median     3Q    Max
```

```
-80.86 -22.18  -9.58  34.54 232.26

Coefficients:
            Value  Std. Error t value
(Intercept)  9.239 40.316      0.229
volume       0.701  0.042     16.641
area         0.514  0.070      7.311

Residual standard error: 39.4 on 12 degrees of freedom
```

```
> allbacks.rlm0 <- rlm(weight ~ volume+area-1, data=allbacks)
> summary(allbacks.rlm0)
```

```
Call: rlm(formula = weight ~ volume + area - 1, data = allbacks)
Residuals:
   Min     1Q Median     3Q    Max
 -86.0  -20.6  -10.3   36.1  231.8

Coefficients:
       Value  Std. Error t value
volume  0.711  0.018     38.511
area    0.517  0.062      8.288

Residual standard error: 39.7 on 13 degrees of freedom
```

Here are plots of residuals against fitted values, for the two models.

```
> par(mfrow=c(1,2))
> plot(allbacks.rlm, which=1)   # residual plot
> mtext(side=3, line=1, "rlm(), intercept included")
> plot(allbacks.rlm0, which=1)  # residual plot
> mtext(side=3, line=1, "rlm(), no intercept")
> par(mfrow=c(1,2))
```

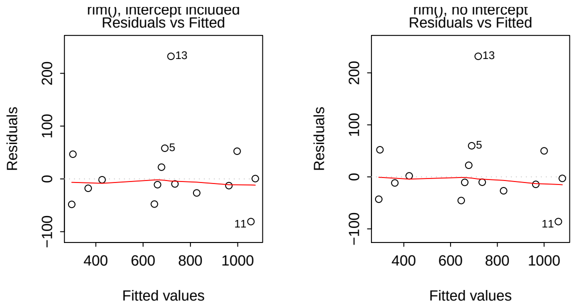

Figure 10: Residuals vs fitted values, for the rlm() models with & without intercept.

Comparison of the coefficients of the intercept and no-intercept with the `lm()` counterparts reveals larger differences in coefficient estimates for the intercept models. The robust method has given smaller coefficient standard errors than `lm()`.

The influence of the outlying observation (the 13th) is reduced using the robust method; therefore, on the residual plots we see this observation featured even more prominently as an outlier than on the corresponding plots for the `lm()` fits.

We next consider the `lqs()` approach. By default, `lqs()` employs a resistant regression method called least trimmed squares regression (lts), an idea due to Rousseeuw

(1984) ("Least median of squares regression." *Journal of the American Statistical Association* 79: 871–888). The method minimizes the sum of the k smallest squared residuals, where k is usually taken to be slightly larger than 50% of the sample size. This approach removes all of the influence of outliers on the fitted regression line.

```
> library(MASS)
> allbacks.lqs <- lqs(weight ~ volume+area, data=allbacks)
> allbacks.lqs$coefficients  # intercept model

(Intercept)      volume       area
   -59.6232      0.7737     0.4709

> allbacks.lqs0 <- lqs(weight ~ volume+area-1, data=allbacks)
> coefficients(allbacks.lqs0)  # no-intercept model

volume   area
0.7117 0.4849
```

The robust coefficient estimates of volume and area are similar to the corresponding coefficient estimates for the `lm()` fit.

Here are plots of residuals against fitted values, for the two models.

```
> par(mfrow=c(1,2))
> plot(allbacks.lqs$residuals ~ allbacks.lqs$fitted.values)
> mtext(side=3, line=1, "lqs(), intercept included")
> plot(allbacks.lqs0$residuals ~ allbacks.lqs0$fitted.values)
> mtext(side=3, line=1, "lqs(), no intercept")
> par(mfrow=c(1,1))
```

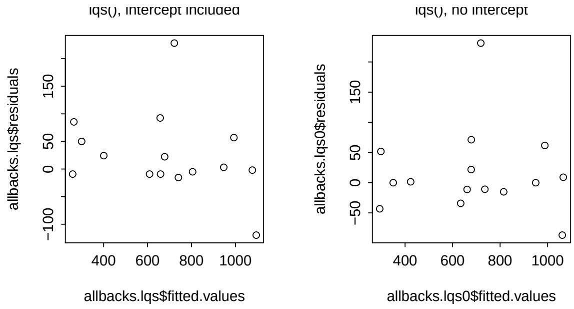

Figure 11: Residuals vs fitted values, for the `lqs()` models with & without intercept.

Because the outlying observation (13) is now not used at all in the final regression coefficient estimates, it has no influence. Neither does observation 11, another outlier. Both points plot farther away from the reference line at 0 than in the corresponding `lm()` residual plots.

Exercise 6
Check the variance inflation factors for `bodywt` and `lsize` for the model `brainwt ~ bodywt + lsize`, fitted to the `litters` data set. Comment.

We can use the function `vif()` to determine the variance inflation factors for the litters data as follows:

```
> litters.lm <- lm(brainwt ~ bodywt + lsize, data=litters)
> vif(litters.lm)
```

```
bodywt  lsize
 11.33  11.33
```

A scatterplot of litter size versus body weight would confirm that the two variables have a relation which is close to linear. The effect is to give inflated standard errors in the above regression, though not enough to obscure the relationship between brain weight and body weight and litter size completely.

It is hazardous to make predictions of brain weight for pigs having body weight and litter size which do not lie close to the line relating these variables.

Exercise 9

(a)
```
> library(MPV)
> plot(y ~ x1, data=table.b3)
```

The scatterplot is suggests a curvilinear relationship.

(b)
```
> library(lattice)
>  xyplot(y ~ x1, group=x11, data=table.b3)
```

This suggests that the apparent nonlinearity is better explained by the two types of transmission.

(c)
```
>  b3.lm <- lm(y ~ x1*x11, data=table.b3)
>  par(mfrow=c(1,4), pty="s")
>  plot(b3.lm)
```

Observation 5 is influential, but it is not an outlier.

(d)
```
> xyplot(resid(b3.lm) ~ x7, group=x11, data=table.b3)
```

This plot demonstrates that observation 5 is quite special. It is based on the only car in the data set with a 3-speed manual transmission.