3

Examples — Data analysis with R

Science, statistics & R	What is the role of statistics and statistical analysis in the wider scientific enterprise?
Scatterplot matrices	Scatterplot matrices can give useful insights on data that will be used for regression or related calculations.
Transformation	Data often require transformation prior to entry into a regression model.
Model objects	Fitting a regression or other such model gives, in the first place, a model object.
Generic functions	`plot()`, `print()` and `summary()` are examples of *generic* functions. With a dataframe as argument `plot()` gives a scatterplot matrix. With an `lm` object, it gives diagnostic plots.
Extractor function	Use an extractor function to extract output from a model object. Extractor fucntions are *generic* functions
List objects	An `lm` model object is a list object. Lists are used extensively in R.

This chapter will use examples to illustrate common issues in the exploration of data and the fitting of regression models. It will round out the discussion of Chapters 1 and 2 by adding some further important technical details.

Issues that will be noted include the use of *generic* functions such as `plot()` and `print()`, the way that regression model objects are structured, and the use of extractor functions to extract information from model objects.

The notes that follow assume a knowledge of basic statistical ideas and methods, to the extent that readers will be comfortable with output that includes details of standard errors, *t*-statistics, and *p*-values. At one point, there is a mention of Bayesian prior probability.

There is use, in several places, of cross-validation for assessing predictive accuracy. For this, data is split up into several subsets (10 is a common choice). Each subset is then left out in turn for use to check predictive accuracy when the model is fitted to the remaining data. Once the process is complete, predictions made independently of the observed values are available for all observations. Differences between observed values and predictions made independently of those values can then be used to assess predictive accuracy.

In bootstrap resampling, repeated with replacement samples are taken from the data. The model can then be refitted to each such re-sample, generating multiple estimates of each coefficient or other statistic of interest. Standard error estimates can then be based on variation between the multiple estimates.

All the approaches that have been noted assume that data can be treated as a random sample from the population to which results will be applied. That may be a heroic assumption.

Notation, when referring to datasets

Data will be used that is taken from several different R packages. The notation `MASS::mammals`, which can be used in code as well as in the textual description, makes it clear that the dataset `mammals` that is required is from the *MASS* package. Should another attached package happen to have a dataset `mammals`, there is no risk of confusion.

3.1 Science, statistics, and R

How does statistical analysis fit into the wider scientific enterprise? While not a central focus of the present notes, the issues that will now be noted are too important to be ignored.

The R system is an enabler that allows users to do effective data analysis, and much else besides. These notes hint at its scope, primarily for data manipulation, for data analysis, and for graphics.

Note, however, the chapters on map overlays and on text mining — these extend into areas that would not ordinarily be described as "data analysis".

For the purposes of this next, the terms "data science" and "statistics" are different names for an endeavour whose concern is to extract meaning from data, leading for example to results that might be reported in a scientific paper, or that might form the basis for a business or government policy. Statistical issues and ideas are fundamental to any use of data to make generalizations that extend beyond the particular data that have been collected, or that are otherwise available. They are fundamental, in that sense, to any scientific use of data. It is, at the same time, important to acknowledge that there are strict limits to what statistical analysis can achieve. Statistical analysis is a partner to, and not a substitute for, robust scientific processes. The use of experimental data provides the simplest context in which to explore this point further.

For experimental work, over and above what may emerge from a statistical analsysis, the demand is that results be replicable. Laboratory studies have repeatedly shown shown that drinking regular caffeinated coffee increases blood pressure, though with minimal long term effects.[1] It is accepted that there is this effect, not from the statistical analysis of results from any individual trial, but because the effect has been demonstrated in repeated trials. The role of statistical analysis has been:

[1] Green et al. (1996)

1. to demonstrate that, collating the evidence from repeated trials, the effect does appear real;
2. to assess the magnitude of the effect.

Worldwide, hundreds of thousands of randomised trials are conducted annually. What do they tell us? In clinical medicine, follow-up trials are common, and clear conclusions will often emerge from the

careful collation of evidence that, in important cases, is likely to follow. In many other areas follow-up trials have until recently been uncommon. This is now changing, and for good reason. Independent replication of the experimental process provides checks on the total experimental process, including the statistical analysis. It is unlikely that the same mistakes in experimental procedure and/or statistical analysis will be repeated.

Papers that had a key role in getting attention to reproducibility concerns have been Prinz et al. (2011) and (Begley and Ellis, 2012). The first (6 out of 53 "landmark" studies reproduced) related to drug trials, and the second (19 out of 65 "seminal" studies) to cancer drug trials. Since those studies appeared, results have appeared from systematic attempts to reproduce published work in psychology (around 40%), in laboratory economics (11 of 18), and in social science (12 of 18).

These replication rates are so low, in the areas to which these papers relate, that they make nonsense of citations to published individual trial results as evidence that a claimed effect has been scientifically demonstrated.

For research and associated analyses with observational data, the absence of experimental control offers serious challenges. In a case where the aim is to compare two or more groups, there are certain to be more differences than the difference that is of interest. Commonly, regression methods are used to apply "covariate adjustments". It is then crucial that all relevant covariates and covariate interactions are accounted for, and that covariates are measured with adequate accuracy. Do covariates require transformation (e.g., x, or $\log(x)$, or x^2) for purposes of use in the regression model?

In a hard-hitting paper titled "Cargo-cult statistics and scientific crisis", Stark and Saltelli (2018) comment, quoting also from Edwards and Roy (2017):

> While some argue that there is no crisis (or at least not a systemic problem), bad incentives, bad scientific practices, outdated methods of vetting and disseminating results, and techno-science appear to be producing misleading and incorrect results. This might produce a crisis of biblical proportions: as Edwards and Roy write: “If a critical mass of scientists become untrustworthy, a tipping point is possible in which the scientific enterprise itself becomes inherently corrupt and public trust is lost, risking a new dark age with devastating consequences to humanity.”

Statistical issues are then front and centre in what many are identifying as a crisis, but are not the whole story. The crisis is one that scientists and statisticians need to tackle in partnership.

In a paper that deserves much more attention than it has received, (Tukey, 1997), John W Tukey argued that, as part of the process of fitting a model and forming a conclusion, there should be incisive and informed critique of the data used, of the model, and of the inferences made. It is important that analysts search out available information about the processes that generated the data, and consider critically

how this may affect the reliance placed on it. Other specific types of challenge (this list is longer than Tukey's) may include:

- For experiments, is the design open to criticism?
- Look for biases in processes that generated the data.
- Look for inadequacies in laboratory procedure.
- Use all relevant graphical or other summary checks to critique the model that underpins the analysis.
- Where possible, check the performance of the model on test data that reflects the manner of use of results. (If for example predictions are made that will be applied a year into the future, check how predictions made a year ahead panned out for historical data.)
- For experimental data, have the work replicated independently by another research group, from generation of data through to analysis.
- Have analysis results been correctly interpreted, in the light of subject area knowledge.

Exposure to diverse challenges will build (or destroy!) confidence in model-based inferences. We should trust those results that have withstood thorough and informed challenge.

Data do not stand on their own. An understanding of the processes that generated the data is crucial to judging how data can and connot reasonably be used. So also is application area insight. Numerous studies found that women taking combined hormone replacement therapy (HRT) also had a lower incidence of coronary heart disease (CHD), relative to women no taking HRT. Randomized trials showed a small but clear increase in risk. The risk was lower in the population as a whole because, for reasons associated with socio-economic status, women taking HRT were on the whole eating healthier food and getting more exercise. In analyses of the population data, account had not been taken of lifestyle effects.

3.1.1 What does R add to the mix?

R clearly has a huge range of abilities for manipulating data, fitting and checking statistical models, and for using graphs and tables to present results. More than this, it has extensive reproducible reporting abilities that can be used to allow others to repeat the data manipulation, analysis, and steps in the processing of output that have led to an eventual paper or report. A file is provided that mixes code with the text for the eventual document, and that is then processed ("woven" or "knitted") to provide the final document, complete with analysis output, tables, graphs, and code (if any) that is to be included in the final document. This makes it straightforward for referees, or for anyone with an interest in the work, to check the analysis and/or try modifi-

cations. The publication of data and code is an important step on the way to making results more open and transparent, and to encouraging informed post-publication critique.

3.2 Generalization beyond the available data

A common statistical analysis aim is to assess the extent to which available data supports conclusions that extend beyond the circumstances that generated the data. The hope is that available data – the sample values – can be used as a window into a wider population.

3.2.1 Models for the random component

Introductory statistics courses are likely to focus on models that assume that *error* terms are independently and identically normally distributed — they make the *iid normal assumption*. This involves the separate assumptions of independence, assumptions of homogeneity of variance (i.e., the standard deviations of all measurements are the same), and normality.

Strict normality is commonly, depending on the use that will be made of model results, unnecessary. Commonly, the interest is in parameter estimates and/or in model predictions. Standard forms of statistical inference then rely on the normality of the the relevant sampling distributions. Central Limit Theorem type effects will then, given enough data, often work to bring these distributions close enough to normality for purposes of the required inferences.

Much of the art of applied statistics lies in recognizing those assumptions that, in any given context, are important and need careful checking. Models are said to be *robust* against those assumptions that are of relatively minor consequence.

Most of the standard elementary methods assume that all population values are chosen with equal probability, independently between sample values. Or, if this is not strictly the case, departures from this assumption will not be serious enough to compromise results. In designed experiments and in sample surveys, randomization mechanisms are used to ensure the required independence. Failures in the randomization process will compromise results.

Where there has not been explicit use of a randomization mechanism, it is necessary to consider carefully how this may have affected the data. Is some form of dependence structure likely? Temporal and spatial dependence arise because values that are close together in time or space are relatively more similar. Is there clustering that arises because all individuals within chosen streets or within chosen families have been included? Two individuals in the same family or in the

same street may be more similar than two individuals chosen at random from the same city suburb. Model choice and model fitting then needs to account for such effects.

Models account for both *fixed* and *random* effects. In a straight line model, the fixed effect is the line, while the random effect accounts for variation about the line. The random part of the model can matter a great deal.

3.2.2 R functions for working with distributions

For each distribution, there are four functions, with names whose first letter is, respectively, `d` (probability or probability **d**ensity), `p` (cumulative **p**robability), `q` (**q**uantile), and `r` (generate a **r**andom sample). Functions that have `d` as their first letter are probabilities for distributions that are discrete, or **d**ensities) for continuoua distributions.

Common discrete distributions are the binomial and the Poisson. The betabinomial is often used to model binomial type data that have a larger than binomial variance. The negative binomial is widely used to model count data that have a larger than Poisson variance. Flexible implementations of the betabinomial, including the ability to model the scale parameter that controls the variance, have appeared only recently. Alternatives to the betabinomial have been little explored. By constrast, for count data, there has been extensive investigation of the negative binomial, as well as of other alternatives to the Poisson.

Easily the most widely used distribution for continuous data is the normal. Data where the error term is not normal will often come close to normal after transformation. The most widely used transformation for this purpose is the logarithmic.

Another name for the normal distribution is the Gaussian distribution.

Other continuous distributions are common in the context of special types of model, e.g., the exponential distribution, or the Weibull of which it is a special case, in waiting time models.

3.3 The Uses of Scatterplots

```
## Below, the dataset MASS::mammals will be required
library(MASS, quietly=TRUE)
```

3.3.1 Transformation to an appropriate scale

A first step is to elicit basic information on the columns in the data, including information on relationships between explanatory variables. Is it desirable to transform one or more variables?

Among other issues, is there a wide enough spread of distinct values that data can be treated as continuous.

Transformations are helpful that ensure, if possible, that:

- All columns have a distribution that is reasonably well spread out over the whole range of values, i.e., it is unsatisfactory to have most values squashed together at one end of the range, with a small number of very small or very large values occupying the remaining part of the range.
- Relationships between columns are roughly linear.
- the scatter about any relationship is similar across the whole range of values.

It may happen that the one transformation, often a logarithmic transformation, will achieve all these at the same time.

The scatterplot in Figure 3.1A, showing data from the dataset `MASS::mammals`, is is an extreme version of the common situation where positive (or non-zero) values are squashed together in the lower part of the range, with a tail out to the right. Such a distribution is said to be "skewed to the right".

Code for Figure 3.1A

```
plot(brain ~ body, data=mammals)
mtext(side=3, line=0.5, adj=0,
      "A: Unlogged data", cex=1.1)
```

Figure 3.1B shows the scatterplot for the logged data. Code for Figure 3.1B is:

```
plot(brain ~ body, data=mammals, log="xy")
mtext(side=3, line=0.5, adj=0,
      "B: Log scales (both axes)", cex=1.1)
```

Where, as in Figure 3.1A, values are concentrated at one end of the range, the small number (perhaps one or two) of values that lie at the other end of the range will, in a straight line regression with that column as the only explanatory variable, be a leverage point. When it is one explanatory variable among several, those values will have an overly large say in determining the coefficient for that variable.

As happened here, a logarithmic transformation will often remove much or all of the skew. Also, as happened here, such transformations often bring the added bonus that relationships between the resulting variables are approximately linear.

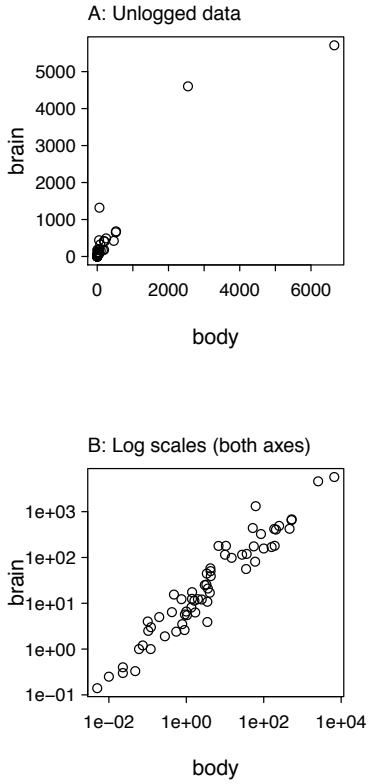

Figure 3.1: Brain weight (g) versus Body weight (kg), for 62 species of mammal. Panel A plots the unlogged data, while Panel B has log scales for both axes, but with axis labels in the original (unlogged) units.

3.3.2 *The Uses of Scatterplot Matrices*

Subsequent chapters will make extensive use of scatterplot matrices. A scatterplot matrix plots every column against every other column, with the result in the layout used for correlation matrices. Figure 3.2 shows a scatterplot matrix for the `datasets::trees` dataset.

The `datasets` package is, in an off-the-shelf installation, attached when R starts.

Interpreting Scatterplot Matrices:

For identifying the axes for each panel

- look across the row to the diagonal to identify the variable on the vertical axis.
- look up or down the column to the diagonal for the variable on the horizontal axis.

Each below diagonal panel is the mirror image of the corresponding above diagonal panel.

```
## Code used for the plot
plot(trees, cex.labels=1.5)
  # Calls pairs(trees)
```

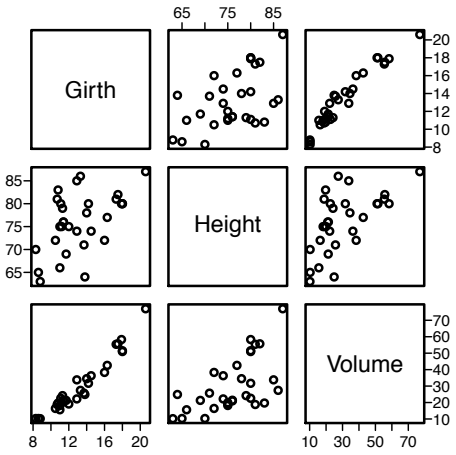

Figure 3.2: Scatterplot matrix for the `trees` data, obtained using the default `plot()` method for data frames. The scatterplot matrix is a graphical counterpart of the correlation matrix.

Notice that `plot()`, called with the dataframe `trees`, has in turn called the plot method for a data frame, i.e., it has called `plot.data.frame()` which has in turn called the function `pairs()`.

The scatterplot matrix may be examined, if there are enough points, for evidence of:

1. Strong clustering in the data, and/or obvious outliers;
2. Clear non-linear relationships, so that a correlation will underestimate the strength of any relationship;
3. Severely skewed distributions, so that the correlation is a biased measure of the strength of relationship.

The scatterplot matrix is best used as an initial coarse screening device. Skewness in the individual distributions is better checked using plots of density estimates.

3.4 World record times for track and field events

The first example is for world track and road record times, as at 9th August 2006. Data, copied down from the web page http://www.gbrathletics.com/wrec.htm, are in the dataset `DAAG::worldRecords`.

Note also the use of these data in the exercise at the end of Chapter 2 (Section 1.9.2)

Data exploration

First, use `str()` to get information on the data frame columns:

```
library(DAAG, quietly=TRUE)
```

```
str(worldRecords, vec.len=3)
```

```
'data.frame':   40 obs. of  5 variables:
 $ Distance   : num  0.1 0.15 0.2 0.3 0.4 0.5 0.6 0.8 ...
 $ roadORtrack: Factor w/ 2 levels "road","track": 2 2 2 2 2 2 2 2 ...
 $ Place      : chr  "Athens" "Cassino" "Atlanta" ...
 $ Time       : num  0.163 0.247 0.322 0.514 ...
 $ Date       : Date, format: "2005-06-14" "1983-05-22" ...
```

Distinguishing points for track events from those for road events is easiest if we use lattice graphics, as in Figure 3.3.

```
## Code
library(lattice)
xyplot(Time ~ Distance, scales=list(tck=0.5),
       groups=roadORtrack, data=worldRecords,
       auto.key=list(columns=2), aspect=1)
## On a a colour device the default is to use
## different colours, not different symbols,
## to distinguish groups.
```

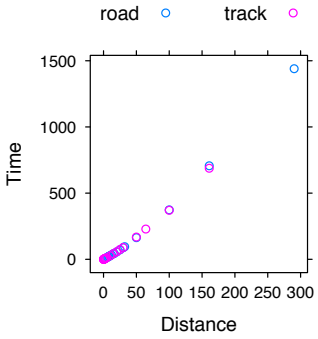

Figure 3.3: World record times versus distance, for field and road events.

Clearly increases in `Time` are not proportional to increases in `Distance`. Indeed, such a model does not make sense; velocity decreases as the length of the race increases. Proportionality when logarithmic scales are used for the two variables does make sense.

Figure 3.4 uses logarithmic scales on both axes. The two panels differ only in the labeling of the scales. The left panel uses labels on scales of $\log_e$, while the right panel has labels in the orginal units. Notice the use of `auto.key` to obtain a key.

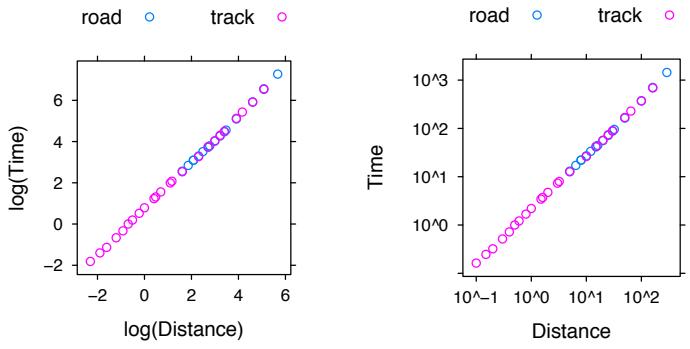

Figure 3.4: World record times versus distance, for field and road events, using logarithmic scales. The left panel uses labels on scales of $\log_e$, while in the right panel, labeling is in the orginal units, expressed as powers of 10.

```
## Code for Left panel
xyplot(log(Time) ~ log(Distance),
       groups=roadORtrack, data=worldRecords,
       scales=list(tck=0.5),
       auto.key=list(columns=2), aspect=1)
## Right panel
xyplot(Time ~ Distance, groups=roadORtrack,
```

```
       data=worldRecords,
       scales=list(log=10, tck=0.5),
       auto.key=list(columns=2), aspect=1)
```

Fitting a regression line

The plots suggest that a line is a good fit. Note however that the data span a huge range of distances. The ratio of longest to shortest distance is almost 3000:1. Departures from the line are of the order of 15% at the upper end of the range, but are so small relative to this huge range that they are not obvious.

The following uses the function `lm()` to fit a straight line fit to the logged data, then extracting the regression coefficients:

The name `lm` is a mnemonic for linear model.

```
worldrec.lm <- lm(log(Time) ~ log(Distance),
                  data=worldRecords)
coef(worldrec.lm)
```

```
  (Intercept) log(Distance)
       0.7316        1.1248
```

The equation gives predicted times:

$$\widehat{\text{Time}} = e^{0.7316} \times \text{Distance}^{1.1248} = 2.08 \times \text{Distance}^{1.1248}$$

This implies, as would be expected, that kilometers per minute increase with increasing distance. Fitting a line to points that are on a log scale thus allows an immediate interpretation.

There is no difference that can be detected visually between the track races and the road races. Careful analysis will in fact find no difference.

3.4.1 *Summary information from model objects*

In order to avoid recalculation of the model information each time that some different information is required, we store the result from the `lm()` calculation in the model object `worldrec.lm`.

Note that the function `abline()` can be used with the model object as argument to add a line to the plot of `log(Time)` against `log(Distance)`.

The name `worldrec.lm` is used to indicate that this is an `lm` object, with data from `worldRecords`. Use any name that seems helpful!

Plot points; add line:

```
plot(log(Time) ~ log(Distance),
     data = worldRecords)
abline(worldrec.lm)
```

Diagnostic plots

Panel A is designed to give an indication whether the relationship really is linear, or whether there is some further systematic component that should perhaps be modeled. It does show systematic differences from a line.

The largest difference is more than a 15% difference.[2] There are mechanisms for using a smooth curve to account for the differences from a line, if these are thought important enough to model.

The plot in panel B allows an assessment of the extent to which individual points are influencing the fitted line. Observation 40 does have both a very large leverage and a large Cook's distance. The plot

[2] A difference of 0.05 on a scale of $\log_e$ translates to a difference of just over 5%. A difference of 0.15 translates to a difference of just over 16%, i.e., slightly more than 15%.

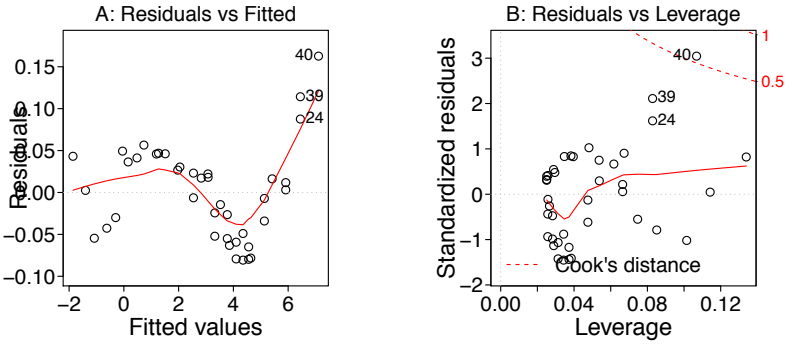

Figure 3.5: First and last of the default diagnostic plots, from the linear model for log(record time) versus log(distance), for field and road events.

on the left makes it clear that this is the point with the largest fitted time. Observation 40 is for a 24h race, or 1440 min. Examine

```
worldRecords["40", ]
```

```
   Distance roadORtrack Place Time       Date
40    290.2        road Basle 1440 1998-05-03
```

3.4.2 *The model object*

Functions that are commonly used to get information about model objects are: `print()`, `summary()` and `plot()`. These are all *generic* functions. The effect of the function depends on the class of object that is printed (ie, by default, displayed on the screen) or or plotted, or summarized.

The function `print()` may display relatively terse output, while `summary()` may display more extensive output. This varies from one type of model object to another.

Compare the outputs from the following:

```
print(worldrec.lm)    # Alternatively, type worldrec.lm
```

```

Call:
lm(formula = log(Time) ~ log(Distance), data = worldRecords)

Coefficients:
  (Intercept)  log(Distance)
        0.732          1.125
```

```
summary(worldrec.lm)
```

```

Call:
lm(formula = log(Time) ~ log(Distance), data = worldRecords)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-0.0807 -0.0497  0.0028  0.0377  0.1627

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.73160    0.01241      59   <2e-16
log(Distance)  1.12475    0.00437     257   <2e-16

Residual standard error: 0.0565 on 38 degrees of freedom
Multiple R^2:  0.999,    Adjusted R^2:  0.999
F-statistic: 6.63e+04 on 1 and 38 DF,  p-value: <2e-16
```

Used with `lm` objects, `print()` calls `print.lm()`, while `summary()` calls `summary.lm()`. Note that typing `worldrec.lm` has the same effect as `print(worldrec.lm)`.

Internally, `summary(wtvol.lm)` calls `UseMethod("summary")`. As `wtvol.lm` is an lm object, this calls `summary.lm()`.

3.4.3 *The `lm` model object is a list*

The model object is actually a list. Here are the names of the list elements:

```
names(worldrec.lm)
```

```
 [1] "coefficients"  "residuals"     "effects"
"rank"
 [5] "fitted.values" "assign"        "qr"
"df.residual"
 [9] "xlevels"       "call"          "terms"
"model"
```

These different list elements hold very different classes and dimensions (or lengths) of object. Hence the use of a list; any collection of different R objects can be brought together into a list.

The following is a check on the model call:

```
worldrec.lm$call
```

```
lm(formula = log(Time) ~ log(Distance), data = worldRecords)
```

Commonly required information is best accessed using generic extractor functions. Above, attention was drawn to `print()`, `summary()` and `plot()`. Other commonly used extractor functions are `residuals()`, `coefficients()`, and `fitted.values()`. These can be abbreviated to `resid()`, `coef()`, and `fitted()`.

Use extractor function `coef()`:

```
coef(worldrec.lm)
```

3.5 *Regression with two explanatory variables*

The dataset `nihills` in the *DAAG* package will be used for a regression fit in Section 8.6. This has record times for Northern Ireland mountain races. Overview details of the data are:

```
str(nihills)
```

```
'data.frame':   23 obs. of  4 variables:
 $ dist : num  7.5 4.2 5.9 6.8 5 4.8 4.3 3 2.5 12 ...
 $ climb: int  1740 1110 1210 3300 1200 950 1600 1500 1500 5080 ...
 $ time : num  0.858 0.467 0.703 1.039 0.541 ...
 $ timef: num  1.064 0.623 0.887 1.214 0.637 ...
```

Figure 3.6 uses the lattice function `splom()` (from the *lattice* package) to give scatterplot matrices, one for the unlogged data, and the other for the logged data. The left panel shows the unlogged data, while the right panel shows the logged data:

The function `splom()` is a *lattice* alternative to `pairs()`, giving a different panel layout.

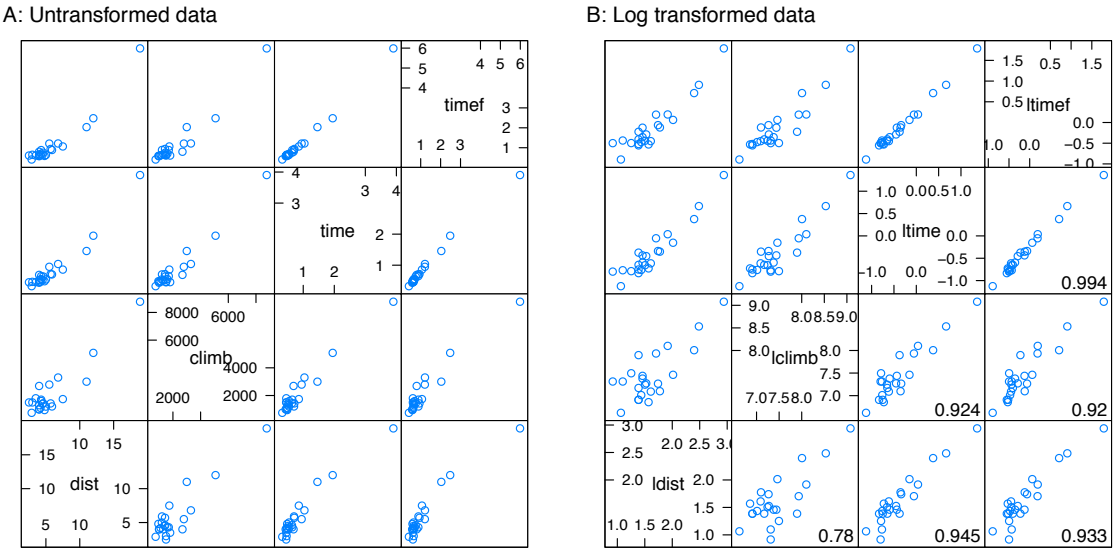

Figure 3.6: Scatterplot matrices for the Northern Ireland mountain racing data. The left panel is for the unlogged data, while the right panel is for the logged data. Code has been added that shows the correlations, in the lower panel.

The following panel function was used to show the correlations:

```
showcorr <- function(x,y,...){
    panel.xyplot(x,y,...)
    xy <- current.panel.limits()
    rho <- paste(round(cor(x,y),3))
    eps <- 0.035*diff(range(y))
    panel.text(max(x), min(y)+eps, rho,
               pos=2, offset=-0.2)
}
```

Code for the scatterplot matrix in the left panel is:

```
## Scatterplot matrix; unlogged data
library(lattice)
splom(~nihills,  xlab="",
       main=list("A: Untransformed data", x=0,
       just="left", fontface="plain"))
```

For the right panel, create a data frame from the logged data:

```
lognihills <- log(nihills)
names(lognihills) <- paste0("l", names(nihills))
## Scatterplot matrix; log scales
splom(~ lognihills, lower.panel=showcorr, xlab="",
       main=list("B: Log transformed data", x=0,
       just="left", fontface="plain"))
```

Unlike `paste()`, the function `paste0()` does not leave spaces between text strings that it pastes together.

Note that the data are positively skewed, i.e., there is a long tail to the right, for all variables. For such data, a logarithmic transformation often gives more nearly linear relationships. The relationships between explanatory variables, and between the dependent variable and explanatory variables, are closer to linear when logarithmic scales are used. Just as importantly, issues with large leverage, so that the point for the largest data values has a much greater leverage and hence much greater influence than other points on the the fitted regression, are greatly reduced.

Notice also that the correlation of 0.913 between `climb` and `dist` in the left panel of Figure 3.6 is very different from the correlation of 0.78 between `lclimb` and `ldist` in the right panel. Correlations where distributions are highly skew are not comparable with correlations where distributions are more nearly symmetric. The statistical properties are different.

The following regresses `log(time)` on `log(climb)` and `log(dist)`:

```
nihills.lm <- lm(ltime ~ lclimb + ldist,
                 data=lognihills)
```

3.6 *One-way Comparisons*

The dataset `tomato` has weights of plants that were grown under one of four different sets of experimental comditions. Five plants were grown under each of the treatments:

A common strategy for getting a valid comparison is to grow the plants in separate pots, with a random arrangement of pots.

- `water only`
- `conc nutrient`
- `2-4-D + conc nutrient`
- `x conc nutrient`

Figure 3.7, created using the function `quickplot()` from the *ggplot2* package, shows the plant weights. Are the apparent differences between treatments large enough that they can be distinguished statistically?

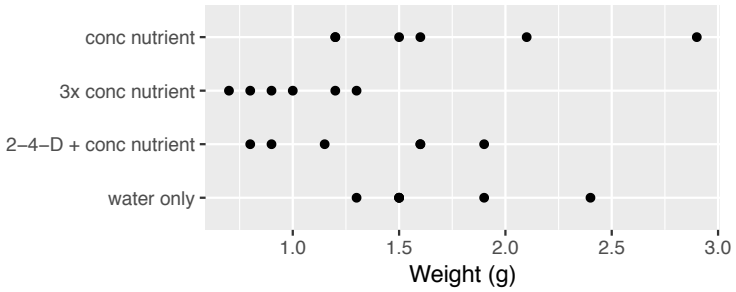

Figure 3.7: Weights (g) of tomato plants grown under four different treatments.

```
## Code
library(ggplot2)
tomato <- within(DAAG::tomato,
                 trt <- relevel(trt, ref="water only"))
quickplot(weight, trt, data=tomato,
          xlab="Weight (g)", ylab="")
```

Notice that "water only" is made the reference level. This choice makes best sense for the analysis of variance calculations that appear below.

The command `aov()`, followed by a call to `summary.lm()`, can be used to analyse these data, thus:

```
tomato.aov <- aov(weight ~ trt, data=tomato)
round(coef(summary.lm(tomato.aov)), 3)
```

Observe that, to get estimates and SEs of treatment effects, `tomato.aov` can be treated as an `lm` (regression) object.

```
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                  1.683      0.187   9.019    0.000
trt2-4-D + conc nutrient    -0.358      0.264  -1.358    0.190
trt3x conc nutrient         -0.700      0.264  -2.652    0.015
trtconc nutrient             0.067      0.264   0.253    0.803
```

Because we made "`water only`" the reference level, "`(Intercept)`" is the mean for `water only`, and the other coefficients are for differences from `water only`.

A randomized block comparison

Growing conditions in a glasshouse or growth chamber — temperature, humidity and air movement — will not be totally uniform. This makes it desirable to do several repeats of the comparison between treatments[3], with conditions within each repeat ("block") kept as uniform as possible. Each different "block" may for example be a different part of the glasshouse or growth chamber.

[3] In language used originally in connection with agricultural field trials, where the comparison was repeated on different blocks of land, each different location is a "block".

The dataset `DAAG::rice` is from an experiment where there were six treatment combinations — three types of fertilizer were applied to each of two varieties of rice plant. There were two repeats, i.e., two blocks.

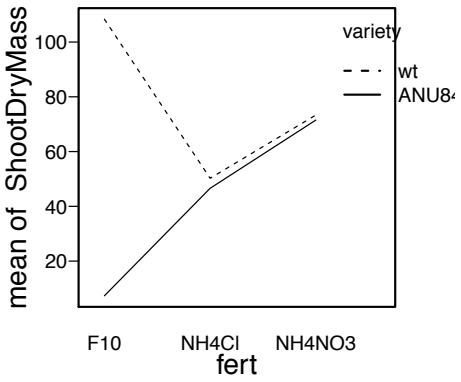

Figure 3.8: Interaction plot for the terms `fert` and `variety`, with `ShootDryMass` as the dependent variable. Notice that for fertilizer F10, there is a huge variety difference in the response. For the other fertilizers, there is no difference of consequence.

```
## Code
library(DAAG)
with(rice, interaction.plot(x.factor=fert,
                            trace.factor=variety,
                            ShootDryMass,
                            cex.lab=1.4))
```

For these data, Figure 3.8 gives a clear picture of the result. For fertilizers `NH4Cl` and `NH4NO3`, any difference between the varieties is inconsequential. There is strong "interaction" between `fert` and `variety`. A formal analysis, accounting for block differences, will confirm what seems already rather clear.

The effect of an appropriate choice of clocks, then carrying out an analysis that accounts for block effects, is to allow a more precise comparison between treatments.

3.7 Time series – Australian annual climate data

The data frame `bomregions2012` from the *DAAG* package has annual rainfall data, both an Australian average and broken down by location within Australia, for 1900 – 2012. Figure 3.9 shows annual rainfall in the Murray-Darling basin, plotted against year.

Data are from the website `http://www.bom.gov.au/climate/change/`

```
## Code
library(DAAG)
plot(mdbRain ~ Year, data=bomregions2018)
## Calculate and plot curve showing long-term trend
with(bomregions2018, lines(lowess(mdbRain ~ Year, f=2/3), lty=2))
## Calculate and plot curve of short-term trends
with(bomregions2018, lines(lowess(mdbRain ~ Year, f=0.1),
                           lty=1, col="gray45"))
```

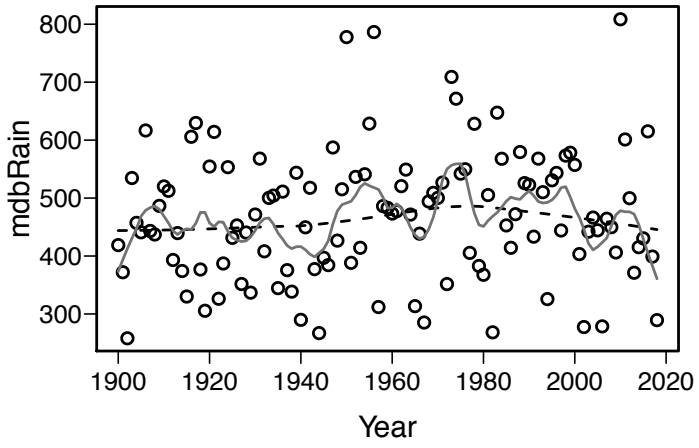

Figure 3.9: Annual rainfall in the Australian Murray-Darling Basin. by year. The `lowess()` function is used to The dashed curve with `f=2/3` captures the overall trend, while the solid curve with `f=0.1` captures trends on a scale of around eleven years. (10% of the 113 year range from 1900 to 2012 is a little more than 11 years.)

The `lowess()` function has been used to fit smooth curves, formed by moving a smoothing window across the data. The dashed curve with `f=2/3` (include 2/3 of the data in the smoothing window) captures the overall trend in the data. The choice of `f=0.1` for the solid curve has the effect that somewhat more than ten years of data are used in determining each fitted value on the smooth.

For each smoothing window, a line or other simple response function is fitted. Greatest weight to points near the centre of the smoothing window, with weights tailing off to zero at the window edge.

This graph is exploratory. A next step might to model a correlation structure in residuals from the overall trend. There are extensive abilities for this. For graphical exploration, note `lag.plot()` (plot series against lagged series).

The functions `acf()` and `pacf()` might be used to examine the correlation structure in the residuals.

The cube root of average rainfall has a more symmetric distribution than rainfall. Thus, use this in preference to average rainfall when fitting models.

3.8 Exercises

1. Plot `Time` against `Distance`, for the `worldRecords` data. Ignoring the obvious curvature, fit a straight line model. Use `plot.lm` to obtain diagnostic plots. What do you notice?

2. The data set `LakeHuron` (*datasets* package) has mean July average water surface elevations (ft) for Lake Huron, for 1875-1972. The following reates a data frame that has the same information:

```
Year=as(time(LakeHuron), "vector")
huron <- data.frame(year=Year, mean.height=LakeHuron)
```

   (a) Plot `mean.height` against year.

(b) To see how each year's mean level is related to the previous year's mean level, use

```
lag.plot(huron$mean.height)
```

This plots the level in each year against the level in the previous year.

(c) *Use the function `acf()` to plot the autocorrelation function. Compare with the result from the `pacf()` (partial autocorrelation). What do the graphs suggest?

For an explanation of the autocorrelation function, look up "Autocorrelation" on Wikepedia.