2

Generalizing from models

Statistical analysis has as a major aim the assessment of the extent to which available data supports conclusions that extend beyond the circumstances that generated the data. This chapter will discuss a range of approaches and perspectives, as they apply to binary comparisons, to one-way comparisons, to contingency tables, and to linear regression. The ideas and issues considered will be important through the remainder of the text.

2.1 Model assumptions

A common model requirement is that *error* terms are independently and identically normally distributed. This *iid* assumption involves the separate assumptions of independence, homogeneity of variance (i.e., the standard deviations of all values are the same), and normality. As was explained in Subsection 1.4.3, strict normality is, depending on the use that will be made of model results, usually unnecessary. Much of the art of statistical analysis lies in recognizing those assumptions that are important and need careful checking. Models are said to be *robust* against those assumptions that are of minor consequence.

2.1.1 Inferences are never assumption free

Subsections 1.8.1 and 1.8.2 discussed permutation test alternatives to t-tests, for use in situations where normality assumptions are in doubt. Rank tests are another such possibility. Contrary to what is sometimes claimed, such tests are not assumption free. Independence assumptions will be no less important than for methods that rely on normality assumptions. Consider carefully what assumptions are made, and how this may affect or limit results.

There is a trade-off between the strength of model assumptions and the ability to find effects. Simple nonparametric approaches may assume less than can reasonably be assumed to be true. If structure in data is ignored, one risks missing insights that parametric methods might provide. Thus, with size and shape (*morphometric*) data from biological organisms, such as the animal `body` and `brain` weight data in Figure 1.5 in Subsection 1.2.3, it makes sense to check for an allometric relationship (linear on logarithmic scales) such as is common with such data.

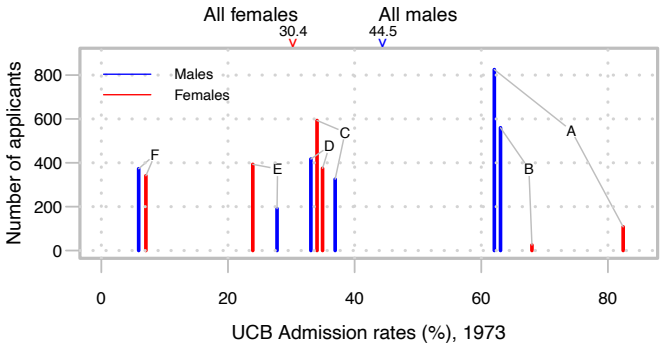


Figure 2.1 The high numbers of males (blue vertical lines) who applied to departments A and B have weighted overall male admission rates towards the high admission rates for those departments (all rates over 60%, both for both males and for the small numbers of female applicants.) The overall female admission rate has been strongly weighted towards the low admission rates (under 40%, both for males and females) in departments C, D, E and F.

2.1.2 Has account been taken of all relevant effects?

Subsection 1.2.7 made the point that failure to account for relevant factors may seriously distort estimated effect(s) for term(s) that remain. There are implications for the conclusions that may be drawn from such standard forms of data summary as multi-way tables or means. The multi-way table `UCBAdmissions` (*datasets* package) provides an example. Figure 2.1 is a graphical summary.

Admission percentages, by sex, totaled across the six largest departments at the University of California at Berkeley in 1973 were (Bickel et al., 1975):

```
## Tabulate by Admit and Gender
byGender <- 100*prop.table(margin.table(UCBAdmissions, margin=1:2), margin=2)
round(byGender,1)
```

```
          Gender
Admit      Male Female
  Admitted 44.5   30.4
  Rejected 55.5   69.6
```

For individual departments, the numbers are:

```
## Admission rates, by department
pcAdmit <- 100*prop.table(UCBAdmissions, margin=2:3)["Admitted", , ]
round(pcAdmit,1)
```

```
        Dept
Gender      A    B    C    D    E    F
  Male   62.1 63.0 36.9 33.1 27.7  5.9
  Female 82.4 68.0 34.1 34.9 23.9  7.0
```

As a fraction of those who applied, females were strongly favored in department A, and males somewhat favored in departments C and E. To understand why the overall proportions favor males, it is necessary to look at the relative numbers applying in the various departments. The numbers were:

```
## Calculate totals, by department, of males & females applying
margin.table(UCBAdmissions, margin=2:3)
```

```
        Dept
Gender     A   B   C   D   E   F
  Male   825 560 325 417 191 373
  Female 108  25 593 375 393 341
```

The overall bias was due to males favoring departments where admission rates were highest. Figure 2.1, which plots admission rates against numbers of applicants, separately for males and females, uses a graph to make this point.

What question is in mind? Is the aim to compare the chances of admission for a randomly chosen female with the chances for a randomly chosen male? The relevant figure is then the overall admission rate of 30.35% for females, as against 44.52% for males. Or, is the interest in the chances of a particular student who has decided on a department? A female had a much better chance than a male in department A, while a male had a slightly better chance in departments C and E.

Here, information was available on the classifying factor on which it was necessary to condition. This will not always be the case. In any tabulation where there is large imbalance in the numbers, the possibility remains open that conditioning on another variable, possibly unobserved, would reverse or modify an observed association. Again, see Aldrich (1995) and Simpson (1951).

In any overall analysis, the classifying (or *conditioning*) factor `Department` must be explicitly incorporated in the model. Section 5.3 demonstrates one suitable approach. See also Exercise 2 at the end of the chapter. The help page that can be accessed by typing `?mantelhaen.test` has further comments and examples.

2.1.3 The limitations of models

Comments in Tukey (1997) are apt:

- Do not assume "that we always know what in fact we never know — the exact probability structure ..."
- "No dataset is large enough to provide complete information about how it should be analyzed."

Subject area knowledge can only to a limited extent make up the deficiency.

Models require critical evaluation to determine the extent to which they give, for their intended use, valid and reliable results. We should trust only those results that have survived informed critical evaluation.

2.1.4 Use the methodology that best suits the task in hand?

In the frequentist approach parameter values are assumed to be unknown constants, and estimates are in most contexts chosen to maximize the 'likelihood'. Frequentist approaches further divide into those that place a strong focus on the use of p-values and on significance testing, and those that focus on comparing likelihoods. Bayesian values treat parameter values as random variables, to be updated as new

data becomes available. An approach that relies on repeated resampling from the one available sample will be the subject of Subsections 1.8.3 and 1.8.4, here treated within a frequentist framework.

The different approaches provide different sources of insight. Likelihood and Bayesian perspectives are important both in their own right, and because they can in principle provide answers to questions that p-values are often wrongly thought to answer.

2.2 t-statistics, binomial proportions, and correlations

This section will note further practical issues for one- and two-sample comparisons. An issue for two-sample comparisons is that the variances may not be equal. One-sample comparisons are commonly appropriate when the interest is in differences of paired values for each of a number n of experimental or observational units.

More generally, the interest may be in correlations between the members of the pair, as a measure of the strength of relationship, in a case where the measures are different quantities such as perhaps hours of sleep and calories consumed.

2.2.1 One- and two-sample t-tests

The one-sample t-test was introduced in Subsection 1.6.2. The discussion that follows will now be extended to two-sample tests, noting also issues that arise for both types of test.

2.2.2 A two-sample comparison

The dataset `DAAG::two65` has amounts of stretch for each of 21 elastic bands, 10 of which were placed in warm water (60-65 degrees C) for four minutes, while the other 11 were left at ambient temperature. After a wait of about ten minutes, the amounts of stretch, under a 1.35 kg weight, were recorded. Means, standard deviations and standard errors, for the stretch of the two sets of bands are:

```
stats2 <- sapply(DAAG::two65,
                 function(x) c(av=mean(x), sd=sd(x), n=length(x)))
pooledsd <- sqrt( sum(stats2['n',]*stats2['sd',]^2)/sum(stats2['n',]-1) )
stats2 <- setNames(c(as.vector(stats2), pooledsd),
                   c('av1','sd1','n1','av2','sd2','n2','pooledsd'))
print(stats2, digits=4)
```

```
    av1      sd1       n1      av2      sd2       n2 pooledsd
253.500    9.925   10.000  244.091   11.734   11.000   11.470
```

The heated versus ambient difference is $\bar{x}_1 - \bar{x}_2 = 9.41$. The pooled standard deviation estimate, calculated as described in Subsection 1.3.3, is $s = 11.47$. The standard error of difference (SED) is thus:

$$\text{SED} = 10.91\sqrt{\frac{1}{10} + \frac{1}{11}} = 5.01$$

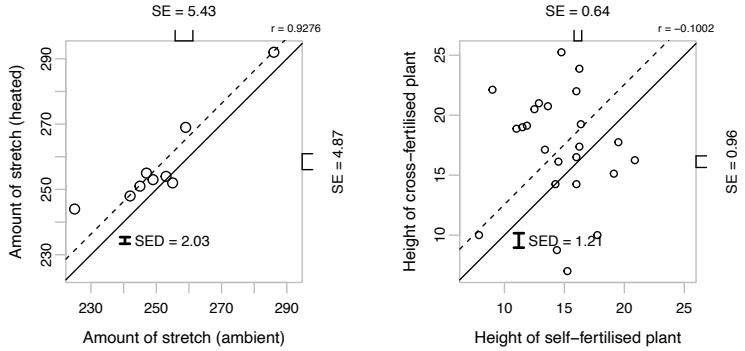


Figure 2.2 Second versus first member, for each pair. The first panel is for the ambient/heated elastic band data (`DAAG::pair65`) from Subsection 1.4.5, while the second is for Darwin's plants (`DAAG::mignonette`).

Use of the function `t.test()` gives:

```
with(DAAG::two65, t.test(heated, ambient, var.equal=TRUE))
```

```
        Two Sample t-test

data:  heated and ambient
t = 2, df = 19, p-value = 0.06
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.5723 19.3905
sample estimates:
mean of x mean of y
    253.5     244.1
```

For a one-sided test, p=0.03, i.e., half of that just given.

This gives, on its own, weak evidence for a difference. The two-sided p-value is at the upper end of a class of results for which $p \leq 0.06$. (It makes as much sense to attach the probability 0.06 to this specific result as it does to argue that, because it is at the lower end of a class of results for which $p \geq 0.06$, it should be regarded as occurring with probability 0.94!)

When is pairing helpful?

Figure 2.2 shows, for two different sets of paired data, a plot of the second member of the pair against the first. The left panel is for the paired elastic band data of Subsection 1.4.5, while the right panel (for the dataset `DAAG::mignonette`) is from the Charles Darwin's experiments that compared the heights of crossed wild mignonette plants with the heights of self-fertilized plants. Plants were paired within the pots in which they were grown, with one plant on one side and one on the other.

For the paired elastic band data there is a clear correlation, and the standard error of the difference is much less than the root mean square of the two separate standard errors. For Darwin's data there is little evidence of correlation. The standard error of differences of pairs is about equal to the root mean square of the two separate

standard errors. For the elastic band data, the pairing was helpful; it led to a low SED. The pairing was not helpful for Darwin's data (note that Darwin gives other datasets where the pairing was helpful, in the sense of allowing a more accurate comparison.)

If the data are paired, then the two-sample t-test corresponds to the wrong model! The one-sample approach is valid, whether or not there is evidence of correlation between members of the same pair.

What if the standard deviations are unequal?

If variances are heterogeneous (unequal variances or standard deviations), the t-statistic based on the pooled variance estimate is inappropriate. The Welch procedure gives, unless degrees of freedom are very small an adequate approximation. The Welch statistic is the difference in means divided by a standard error of difference that allows for unequal variances, i.e.,

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\text{SED}}, \quad \text{where SED} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}.$$

If the two variances are unequal this does not have a t-distribution. However, critical values are quite well approximated by the critical values of a t-distribution with degrees of freedom given by a function of variances and sample sizes that is due to Welch (1949). The test has the name *Welch test*. For details, see Miller (1986). The function `t.test()` has the Welch test as its default; unequal variances are assumed unless the argument `var.equal=TRUE` is given.

If $n_1 = n_2$ then the statistic is the same as for the t-test that is based on the pooled estimate of variance, but with reduced degrees of freedom.

2.2.3 The normal approximation to the binomial

We assume that individuals are drawn independently and at random from a binomial population where individuals are in one of two categories – male as opposed to female, a favorable treatment outcome as opposed to an unfavorable outcome, survival as opposed to non-survival, defective as opposed to non-defective, Democrat as opposed to Republican, etc. Let π be the population proportion. In a sample of size n, the proportion in the category of interest is denoted by p. Then,

$$\text{SE}[p] = \sqrt{\pi(1-\pi)/n}.$$

An approximate 95% confidence bound for the proportion π is:

$$p \pm 1.96\sqrt{\frac{p(1-p)}{n}}.$$

An upper bound for SE$[p]$ is $1/(2\sqrt{n})$. If π is between about 0.35 and 0.65, the inaccuracy in taking SE$[p]$ as $1/(2\sqrt{n})$ is small. This approximation leads to the confidence intervals shown in Table 2.1. See `?binom.test` for calculation of confidence intervals for binomial proportions more generally.

Table 2.1 *Approximate 95% confidence interval, assuming* $0.35 \leq \pi \leq 0.65$.

Sample size n	25	100	400	1000
Approximate 95% confidence interval	$p \pm 20\%$	$p \pm 20\%$	$p \pm 5\%$	$p \pm 3.1\%$

2.2.4 The Pearson or product–moment correlation

The Pearson correlation measures linear association. It equals:

$$\frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2\sum(y-\bar{y})^2}}$$

where the sums are taken over all observations.

The usual interpretation of the magnitude of the Pearson correlation assumes that sample pairs (x, y) have been taken at random from a bivariate normal distribution. Observations must be independent, and the marginal distributions of x and y both approximately normal. If the marginal distributions are highly asymmetric, the correlation is likely to be smaller, with increased statistical variability.

The following is an extreme example that makes a comparison with the Spearman rank correlation. For calculating the Spearman correlation variable values for each of x and y are in each case ranked by order of magnitude (1,2, ...), with the Pearson correlation then calculated between the ranked values.[1]

```
## Pearson correlation between `body` and `brain`: Animals
Animals <- MASS::Animals
rho <- with(Animals, cor(body, brain))
## Pearson correlation, after log transformation
rhoLogged <- with(log(Animals), cor(body, brain))
## Spearman rank correlation
rhoSpearman <- with(Animals, cor(body, brain, method="spearman"))
c(Pearson=round(rho,2), " Pearson:log values"=round(rhoLogged,2),
  Spearman=round(rhoSpearman,2))
```

```
            Pearson  Pearson:log values            Spearman
              -0.01                0.78                0.72
```

The standard error of the correlation coefficient is for most inferential purposes, not a useful statistic. The distribution of the sample correlation, under the usual assumptions (e.g., bivariate normality), is too skew. Where it is reasonable to assume that the joint distribution of (x, y) pairs from which it has been calculated is bivariate normal, Fisher's z-transformation can be used to transform the Pearson statistic to a distribution that is close to normal and can be used for inference. (In practice, assuming independence between different (x, y) pairs, it may be enough to check that both x and y have normal distributions. The test that the population correlation ρ is zero requires only that, for given x, the distribution of y is normal, independently for different values of x.)

Fisher's z-statistic is:

$$z = 0.5\log(\frac{1+r}{1-r})$$

[1] Values that are equal in magnitude are commonly given the average of the relevant ranks.

The standard deviation, again to a close approximation, is $1/\sqrt{n-3}$, where n is the number of observations.

The function `cor.test()` may be used for the calculations, for the Kendall correlation that was mentioned in Subsection 1.3.7 as well as for the Pearson and Spearman correlations. For `method="pearson"` (the default), with at least 4 pairs of (x, y) values, `cor.test()` outputs a confidence interval for the correlation.

Methods for comparing Pearson correlations are implemented in the *cocor* package. An accompanying vignette that reproduces the Diedenhofen and Musch (2015) paper has extensive methodological and historical details.

2.3 Extra-binomial and extra-Poisson variation

Both for the binomial and for the Poisson, one parameter determines both the mean and the variance. This limits the data for which they can provide useful models. The most commonly implemented alternative to the binomial is the betabinomial. There are a number of alternatives to the Poisson that have been widely implemented, with the negative binomial the best known. Attention has mainly been on distributions for *overdispersed* data where the variance is greater than for the binomial or Poisson, with more limited attention on the more unusual *underdispersion*.

Event processes lead to the Poisson distribution and its generalization. Or it may be seen as a limiting case of the binomial as the size n goes to infinity and probability π goes to zero, with the binomial mean constant at $n\pi = \lambda$.

For an event process (e.g., radioactive decay events), the number of counts in any time interval will be Poisson if:

- Events occur independently – the occurrence of one event does not change the probability of a further event, and
- The rate λ at which events occur is constant.

Thus, in radioactive decay, atoms appear to decay independently (and emit ionizing radiation), at a rate that is the same for all atoms. The mean is λ, which is also the variance. If the sample mean and the sample variance differ only by statistical error, data are to this extent consistent with a Poisson distribution.

2.3.1 Checks for extra-binomial and extra-Poisson variation

The probabilities returned by `pbinom()` and `ppois()`, and by other functions that return 'quantiles' for discrete distributions, change in discrete jumps. As an example where the jumps are large, the probabilities of number of heads in two coin tosses change from 0.25 (no heads) to 0.5 (1 head or less) to 1.0 (2 heads or less). This complicates providing an equivalent of the normal quantile-quantile plot that has the same visual effectiveness. Randomized quantile residuals are obtained by replacing the quantiles of the fitted distribution by the equivalent normal quantile. A randomization process is then used to obtain 'residuals' that aside from sampling

variability, have a normal distribution if distributional assumptions are correct. The discussion in Subsection 7.6.2 has further details.

Now consider two datasets, one of which shows systematic clear departures from a binomial fit, while the second shows clear departures from a Poisson fit.

Extra-binomial variation in the male/female balance in large families

The dataset qra::malesINfirst12, from hospital records in Saxony in the nineteenth century, gives the number of males among the first 12 children of family size 13 in 6115 families. The probability that a child will be male varies, within and/or between families. (The 13th child is ignored to counter the effect of families non-randomly stopping when a desired gender is reached.), Data, with fitted binomial values, residuals, and standard deviations of fitted binomial values, are:

```
maleDF <− data.frame(number=0:12, freq=unname(qra::malesINfirst12[["freq"]]))
N <− sum(maleDF$freq)
pihat <− with(maleDF, weighted.mean(number, freq))/12
probBin <− dbinom(0:12, size=12, prob=pihat)
rbind(Frequency=setNames(maleDF$freq, nm=0:12),
      binomialFit=setNames(probBin∗N, nm=0:12),
      rawResiduals = maleDF$freq−probBin∗N,
      SDbinomial=sqrt(probBin∗(1−probBin)∗N)) |>
  formatC(digits=2, format="fg") |> print(digits=2, quote=F, right=T)
```

```
                0   1   2   3   4    5    6    7   8   9  10  11  12
Frequency       3  24 104 286 670 1033 1343 1112 829 478 181  45   7
binomialFit  0.93  12  72 258 628 1085 1367 1266 854 410 133  26 2.3
rawResiduals  2.1  12  32  28  42  -52  -24 -154 -25  68  48  19 4.7
SDbinomial   0.97 3.5 8.4  16  24   30   33   32  27  20  11 5.1 1.5
```

Notice the systematic manner in which the residuals go from positive to negative to positive. This happens because the standard deviation of the binomial number of successes is much smaller in the tails.

We now proceed to examine plots that check, first on the binomial fit to the Saxony data, and then a fit that uses, as an alternative, the betabinomial distribution. The betabinomial distribution has a scale parameter that allows the variance to increase beyond that for a binomial distribution. The function `gamlss::gamlss` is used to fit the model, in preference to implementations in other packages, because of its convenience for use with associated functions that provide plots of residuals.

Figure 2.3 compares, for each model, the quantile-quantile (Q-Q) plot of randomized quantile residuals with the corresponding worm plot. The worm plots show departures from a line with slope 1.0, with the same horizontal scale as for the Q-Q plot. Worm plots can show substantial variation from one plot to the next, so that can be important to repeat the plot several times.

The steady change from negative to positive differences from the line with unit slope in Panel A is very obvious in Panel B. The absence of this clear pattern, apparent in Panel D and very obvious in Panels E and F, suggest a well-fitting model. The curved boundary lines in Panels B and D are pointwise 95% bounds, assuming the fitted distribution.

Code for fitting the models is:

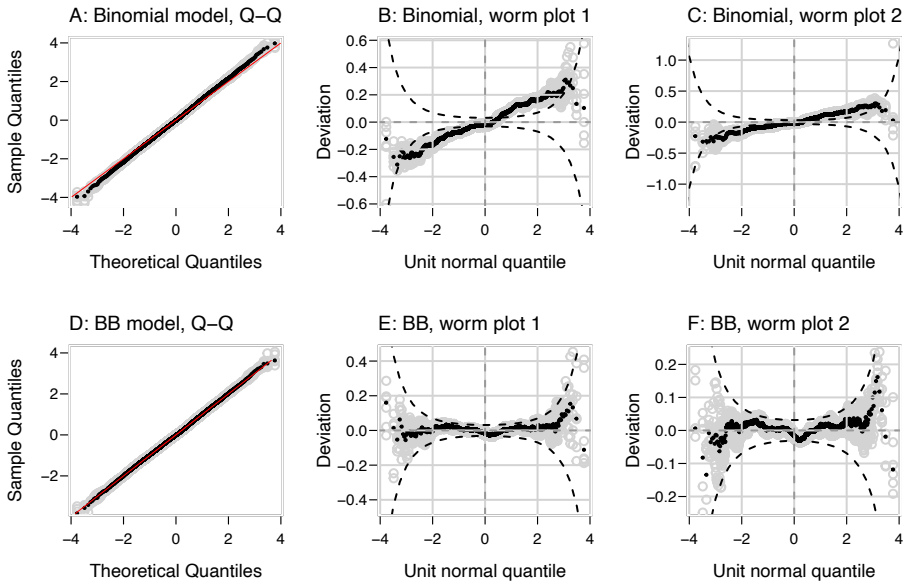


Figure 2.3 Panel A is a quantile-quantile plot and Panels B and C are worm plots, that show medians from 6 sets of randomized quantile residuals, for the fit to a binomial distribution. Panels D, E, and F show the corresponding plots for the fit to a betabinomial distribution.

```
## Fit binomial and betabinomial distributions.
suppressPackageStartupMessages(library(gamlss))
doBI <- gamlss(cbind(number, 12-number)~1, weights=freq,
               family=BI, data=maleDF, trace=FALSE)
doBB <- gamlss(cbind(number, 12-number)~1, weights=freq,
                    family=BB, data=maleDF, trace=FALSE)
```

Code for Panels A and B is:

```
rqres.plot (doBI, plot.type='all', type="QQ", main=""); box(col='white')
rqres.plot (doBI, plot.type='all', type="wp", main=""); box(col='white')
## Plots C, D, E, F: Set object name; set`type="wp" (C, E, F), or`"QQ"` (D)
```

The AIC statistic that was described in Subsection 1.7.1, calculated using the relevant methods for *gamlss* models, shows a substantially smaller value and thus a clear preference for the betabinomial:

```
aicStat <- AIC(doBI, doBB)
rownames(aicStat) <-
  c(doBI="Binomial", doBB="Betabinomial")[rownames(aicStat)]
aicStat$dAIC <- with(aicStat, round(AIC-AIC[1],1))
aicStat
```

```
             df   AIC dAIC
Betabinomial  2 24990  0.0
Binomial      1 25070 80.6
```

Data with strong extra-Poisson variation

Consider now counts of numbers of accidents among 414 machinists from a three months study conducted around the end of World War I (Greenwood and Wood, 1919). In data such as this, it is entirely to be expected that there will be substantial variation – some are much more prone to accidents than others:

```
## Numbers of accidents in three months, with Poisson fit
machinists <- data.frame(number=0:8, freq=c(296, 74, 26, 8, 4, 4, 1, 0, 1))
N <- sum(machinists[['freq']])
lambda <- with(machinists, weighted.mean(number, freq))
fitPoisson <- dpois(0:8, lambda)*sum(machinists[['freq']])
rbind(Frequency=with(machinists, setNames(freq, number)),
      poissonFit=fitPoisson) |>
  formatC(digits=2, format="fg") |> print(quote=F, digits=2, right=T)
```

```
             0   1   2   3    4     5      6       7        8
Frequency  296  74  26   8    4     4      1       0        1
poissonFit 255 123  30 4.8 0.58 0.056 0.0045 0.00031 0.000019
```

The very poor fit for 0 or 1 accidents is obvious, with no need to check residuals.

The negative binomial is the most frequently used generalization of the Poisson that is designed for count data where the variance is greater than the mean. Figure 2.4 shows the worm lots for each of the two models.

Code that fits the models is:

```
doPO <- gamlss(number~1, weights=freq,
               family=PO, data=machinists, trace=FALSE)
doNBI <- gamlss(number~1, weights=freq,
                family=NBI, data=machinists, trace=FALSE)
```

Code for Panels A and B (or C) is:

```
rqres.plot (doPO, plot.type='all', type="QQ", main=""); box(col='white')
## Repeat, changing the argument, for remaining plots
```

There are two variants, labeled in the *gamlss* package as NBI and NBII. Here, with counts for one distributional mean, NBI and NBII provide different ways

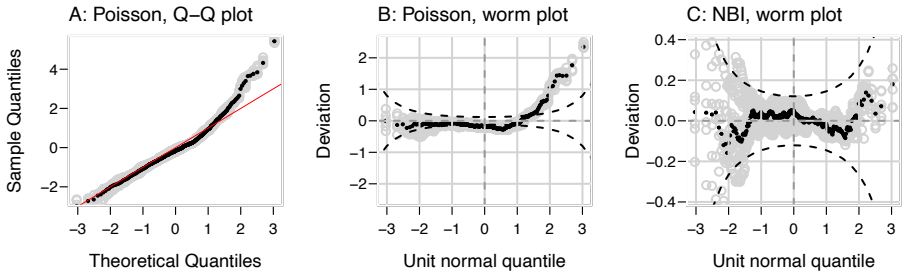


Figure 2.4 Panel A is a worm plot, showing medians from 6 sets of randomized quantile residuals, for the fit to a Poisson distribution. Panels B and C show corresponding plots for the fit to a negative binomial type I distribution.

to parameterize the distribution. In Subsection 5.4.3, where the interest is in the effect of habitat on the densities of two species of moths, NBI and NBII will lead to different fitted values.

2.3.2* Technical details – extra-binomial or extra-Poisson variation

The `stats::glm()` function provides for `quasibinomial` (and `quasipoisson` families. These are not formally defined distributions. Instead, they fit just as for the 'binomial' or 'poisson', but estimate a 'dispersion' Φ from the fitted model that is a multiplier for the variance, allowing it to be larger (or, much less commonly, smaller) than the respective binomial or poisson variance. With Φ thus defined, the variance for the binomial-like number x of 'successes' out of n is $n\pi(1-\pi)\Phi$. The variance for the Poisson-like count x with mean λ is $\lambda\Phi$.

The *gamlss* implementation of the betabinomial has a probability parameter `mu` and 'dispersion' (or 'scale') parameter `sigma`. The parameter σ has its own link function, by default $\log()$. The dispersion index Φ that is the direct analogue of the dispersion as defined for `glm()` models, is $(1+n\sigma)/(1+\sigma)$. The betabinomial distribution is derived by assuming that the probability of 'success', rather than being fixed, varies randomly according according to a beta distribution. As well as the betabinomial the *gamlss* package has also the double binomial. As for the betabinomial, the binomial ($\Phi = 1$) is a limiting case.

Section 7.6 will demonstrate the use of the betabinomial in the *glmmTMB* package. This provides a 'dispersion' parameter ϕ which is the inverse of the parameter σ in *gamlss*, i.e. $\phi = \sigma^{-1}$. As for other distributional families, the way that the variance relates to the particular 'dispersion' (or 'scale') parameter can be different between implementations in different R packages.

There are in addition zero-inflated versions of the distributions noted, and zero-adjusted versions of all except the double binomial. These have a further parameter, named 'nu' in the *gamlss* implementation, that is described as a *shape'* parameter.

An insightful way to relate the different parameterizations of the betabinomial is to express the dispersion parameter as a function of the intra-class correlation ρ. A positive correlation leads to more homogeneous responses within replicates, and manifests itself in greater between replicate differences, leading to a dispersion index Φ that is greater than one. Then:

$$\rho = \frac{\sigma}{\sigma+1} \quad (\sigma \text{ is the dispersion parameter in gamlss})$$

The dispersion index Φ (multiplier for $n\pi(\pi-1)$) is then

$$\Phi = 1 + (n-1)\rho \tag{2.1}$$

$$= \frac{1+n\sigma}{1+\sigma} \tag{2.2}$$

The following calculates the "dispersion index" for the betabinomial fit to the Saxony family male/female split data, equivalent to the 'dispersion' as defined for glm() models with quasibinomial errors:

```
sigma <- exp(coef(doBB, "sigma"))
cat("Phi =", (1+12*sigma)/(1+sigma))
```

```
Phi = 1.165
```

The increase relative to the binomial is small, but because of the large numbers in the dataset, stands out clearly.

For more details on the beta-binomial, see for example Morgan (1992). Morgan and Ridout (2008) is interesting because it compares use of a binomial distribution, a beta-binomial, and a mixture of the two distributions.

As parameterized in the *gamlss* package, the negative binomial type I (NBI) distribution has variance, with multiplier Φ:

$$\text{Variance} = = \mu(1 + \mu\sigma), \quad \text{so that } \Phi = (1 + \mu\sigma)$$

For the machinist accidents data, this equals:

```
mu <- exp(coef(doNBI, "mu"))
sigma <- exp(coef(doNBI, "sigma"))
cat("Phi =", (1+sigma*mu))
```

```
Phi = 2.019
```

For the negative binomial type II (NBII) distribution, the variance is $\mu(1 + \sigma)$, so that $\Phi = 1 + \sigma$.

Where μ varies as a function of factors or other terms in a model, the difference between the negative binomial types NBI and NBII models is of consequence. For NBI the variance changes by a factor $\Phi = (1 + \mu\sigma)$ as μ changes, while for NBII the variance is a constant multiple $1 + \sigma$ of μ.

The packages *gamlss* and *VGAM*, and a number of others, implement a number of other distributions that are designed to model extra-Poisson variation. There is a much wider choice than for distributions that model extra-binomial variation.

2.4 Contingency tables

The `psid3` and `nswdemo` datasets (*DAAG*) relate to US studies that evaluated labor training programs. For details, see `?DAAG::nswdemo`. The following two-way table gives number of observations, classified according to high school graduates versus number of dropouts, and according to non-participation in a labor training program (PSID3 group) or participation (NSW group).

```
## 'Untreated' rows (no training) from psid3, 'treated' rows from nswdemo
nswpsid3 <- rbind(DAAG::psid3, subset(DAAG::nswdemo, trt==1))
degTAB <- with(nswpsid3, table(trt,nodeg))
# Code 'Yes' if completed high school; 'No' if dropout
dimnames(degTAB) <- list(trt=c("PSID3_males","NSW_male_trainees"),
                         deg =c("Yes","No"))
degTAB
```

Table 2.2 *Calculated expected values for a contingency table.*

	High School Graduate			
	Yes	No	Total	Row proportion
PSID3	63 (43.07)	65 (84.93)	128	$128/425 = 0.3012$
NSW74 trainees	80 (99.93)	217 (197.07)	297	$297/425 = 0.6988$
Total	143	282	425	
Column proportion	$143/425 = 0.3365$	$282/425 = 0.6635$		

```
                  deg
trt                Yes  No
  PSID3_males       63  65
  NSW_male_trainees 80 217
```

Table 2.2 is designed to accompany an explanation, given below, of details of the calculations for a chi-squared test for no association. The datasets, and other related datasets, will be further discussed in Subsection 9.7.1.

The table shows a much higher proportion of high school dropouts in the NSW group than in the PSID3 group. The chi-squared test for no association, described in the next subsection, can be used for a formal test of significance:

```
# To agree with hand calculation below, specify correct=FALSE
chisq.test (degTAB, correct=FALSE)
```

```
        Pearson's Chi-squared test

data:  degTAB
X-squared = 20, df = 1, p-value = 0.000008
```

Statistical variability is an unlikely explanation for the much stronger representation of high school dropouts in the NSW data.

The mechanics of the chi-squared test

The null hypothesis is that the proportion of the total in each cell is, to within random error, the result of multiplying a row proportion by a column proportion. The independence assumption, i.e., the assumption of independent allocation to the cells of the table, is crucial. Where it is possible to form replicate tables, the assumption should be checked.

Given I rows and J columns, the expected value in cell (i, j) is calculated as

$$E_{ij} = \text{(proportion for row } i) \times \text{(proportion for column } j) \times \text{total.} \qquad (2.3)$$

It follows that the expected values can be found by multiplying the column totals by the row proportions. (Alternatively, the row totals can be multiplied by the column proportions.) Thus, $128 \times 0.3365 = 43.07$, $128 \times 0.3365 = 84.93$, etc.

A chi-squared residual for each cell of the table can be calculated as

$$\frac{O_{ij} - E_{ij}}{\sqrt{E_{ij}}}, \text{ where } O_{ij} \text{is the observed value in cell } (i, j).$$

Table 2.3
Dreamer and object movements, compiled from Hobson (1988, Table 12.1, p. 248).

	Object moves	
Dreamer moves	Yes	No
Yes	5	17
No	3	85

Squaring these values, and summing over all cells of the table gives a value for the chi-squared statistic that is the same as that returned by the R function `chisq.test()` when the default `correct=TRUE` is changed to `correct=FALSE`. See `?chisq.test` for details.

Under the null hypothesis the chi-squared statistic has an approximate chi-squared distribution with $(I-1)(J-1)$ degrees of freedom. In Table 2.2, the values in parentheses are the expected values E_{ij}.

An example where a chi-squared test may not be valid

Table 2.3 summarizes information that Hobson (1988) derived from drawings of dreams, made by an unknown person whom he names "The Engine Man". For each of 110 drawings Hobson notes whether the dreamer moves, whether an object moves, and other details also. Dreamer movement may occur if an object moves, but is relatively rare if there is no object movement. (Note that the Table 2.3 form of summary is ours, based on the summaries that Hobson provides.)

It may seem natural to do a chi-squared test for no association.[2] This gives $X^2 = 7.1$ (1 d.f.), $p = 0.008$. Note however that there is a time sequence to the dreams. There might well be runs of dreams of the same type. If the sequence in which Hobson records details represents the sequence in time, this will allow a check of the strength of any evidence for runs.

A further point concerns the adequacy of the chi-squared approximation, under the assumption that counts enter independently into the cells of the table. The commonly used rule that all expected values should be at least 2 (Miller, 1986) is satisfied for the data of Table 2.3. A check is to do a Fisher 'exact' test. In this instance the Fisher exact test[3] gives, surprisingly, the same result as the chi-squared test, i.e., $p = 0.008$.

Rare and endangered plant species

The calculations for a test for no association in a two-way table can sometimes give useful insight, even where a formal test of statistical significance would be invalid. The example that now follows illustrates this point. Data are from species lists for various regions of Australia. Species were classified `CC`, `CR`, `RC` and `RR`, with

[2]
```
## Engine man data
engineman <- matrix(c(5,3,17,85), 2,2)
chisq.test(engineman)
```
[3]
```
fisher.test(engineman)
```

C denoting common and R denoting rare. The first code letter relates to South Australia and Victoria, and the second to Tasmania. They were further classified by habitat according to the Victorian register, where D = dry only, W = wet only, and WD = wet or dry.

Data can be entered thus:

```
## Enter the data thus:
rareplants <- matrix(c(37,190,94,23, 59,23,10,141,  28,15,58,16),  ncol=3,
  byrow=TRUE, dimnames=list(c("CC","CR","RC","RR"), c("D","W","WD")))
```

We use a chi-squared calculation to check whether the classification into the different habitats is similar for the different rows. Details of the calculations are:

```
(x2 <- chisq.test(rareplants))
```

```
        Pearson's Chi-squared test

data:  rareplants
X-squared = 35, df = 6, p-value = 0.000004
```

This low p-value should attract a level of skepticism. We do not have a random sample from some meaningful larger population. Some species may be commonly found together, for reasons unconnected with habitat type. As a simplistic thought experiment suppose that species come in closely linked pairs, with both members of the pair always falling into the same cell of the table. This would inflate the chi-squared statistic by a factor of 2 (the net effect of inflating the numerator by 2^2, and the denominator by 2). Available information does not allow estimation of the extent of any such clustering.

Examination of departures from a consistent overall row pattern

The following shows the observed values, together with expected values and residuals, that are generated as part of the `chisq.test()` calculations:

```
## Observed
## values
rareplants
```

```
    D   W WD
CC 37 190 94
CR 23  59 23
RC 10 141 28
RR 15  58 16
```

```
## Expected
## values
x2$expected
```

```
      D     W    WD
CC 39.3 207.2 74.5
CR 12.9  67.8 24.4
RC 21.9 115.6 41.5
RR 10.9  57.5 20.6
```

```
## Standardized
## residuals
residuals(x2)
```

```
       D      W     WD
CC -0.37 -1.196  2.26
CR  2.83 -1.067 -0.28
RC -2.55  2.368 -2.10
RR  1.24  0.072 -1.02
```

Under the null hypothesis, the expected relative numbers in different columns are the same in every row. The chi-squared residuals are measures of departures from this pattern.

The null hypothesis assumption is that allocation to cells is independent, with probabilities (row probabiliy) × (column probability) and expected values as given by Equation 2.3. If numbers are large enough, residuals will then for all practical purposes behave like random normal deviates with mean zero and variance one,

If numbers are large enough the residuals will, assuming that allocation to cells

is independent with probabilities (row probabiliy) × (column probability), behave like random normal deviates with mean zero and variance one, so that the expected values are as given by Equation 2.3

The `CC` species are, relative to the overall average, over-represented in the `WD` classification; the `CR` species are over-represented in the `D` classification; the `RC` species are under-represented in `D` and `WD`, and over-represented in `W`.

Interpretation issues

Having found an association in a contingency table, what does it mean? The interpretation will differ depending on the context. The incidence of gastric cancer is relatively high in Japan and China. Do screening programs help? Here are two ways in which the problem has been studied:

- In a long term follow-up study, patients who had surgery for gastric cancer may be classified into two groups – a 'screened' group whose cancer was detected by mass screening, and an 'unscreened' group who presented at a clinic or hospital with gastric cancer. The five-year mortality may be around 58% in the unscreened group, compared with 72% in the screened group, out of approximately 300 patients in each group.
- In a prospective cohort study, two populations – a screened population and an unscreened population – may be compared. The death rates in the two populations over a ten-year period may then be compared. For example, the annual death rate may be of the order of 60 per 100 000 for the unscreened group, compared with 30 per 100 000 for the screened group, in populations of several thousand individuals.

In the long term follow-up study, the process that led to the detection of cancer was different between the screened and unscreened groups. The screening may lead to surgery for some cancers that would otherwise lie dormant long enough that they would never attract clinical attention. It is necessary, as in the prospective cohort study, to compare all patients in a screened group with all patients in an unscreened group. As patients were not divided randomly between the two groups, results are even so not conclusive.

2.5 Issues for Regression with a single explanatory variable

2.5.1 Iron slag example — check residuals with care!

In the example now considered, there is an evident pattern in residuals from a straight line regression. The data compare two methods for measuring the iron content in slag – a magnetic method and a chemical method (data are from Hand et al., 1993). The chemical method required greater effort and was presumably more expensive, while the magnetic method was quicker and easier.

The plot of residuals against fitted values in Figure 2.5A gives a strong hint of pattern in the residuals. In Panel B, the argument `span` to the function that plots the smooth has been increased from the default, from 2/3 to 0.8, in order to prevent the large positive residual on the left from obscuring the pattern.

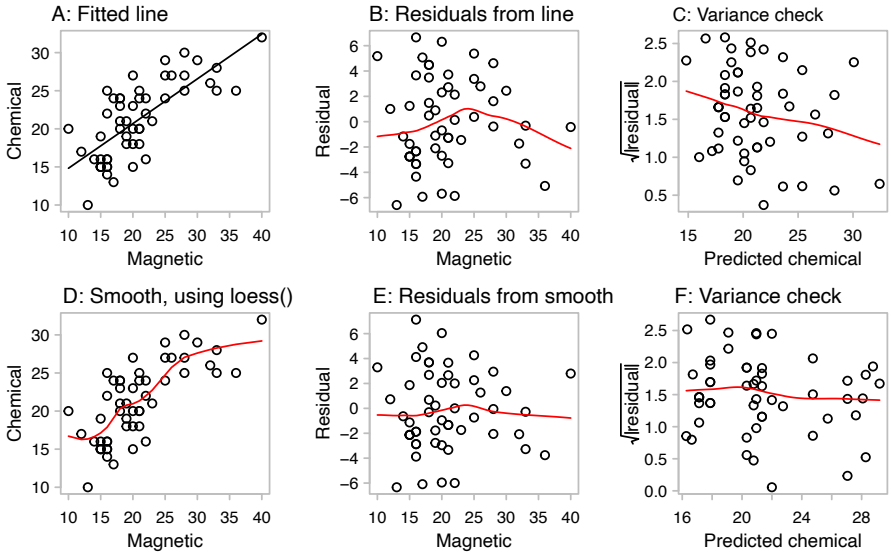


Figure 2.5 Panels A, B, and C (top row) are from adding a line to the plot of `chemical` against `magnetic`. Panels D, E, and F (bottom row) are from adding a loess smooth. In C, the downward slope might appear to suggest lower variance for larger fitted values. Panel F is the equivalent plot from fitting a loess curve. Any suggestion that variance changes with fitted value has gone.

Panel C gives a visual check on whether the error variance is constant. (Taking the square root of absolute values of the residuals symmetrizes their distribution, giving a more meaningful smooth and a plot that makes better visual sense.) Panel C gives a strong, but misleading, suggestion that the variance decreases with increasing value of `magnetic`. The smooth in Panel C, because it relates to the straight line model that Figures 2.5A and B indicate is inappropriate, is misleading.

Now, use the function `loess()` to fit a smooth curve to the data in Figure 2.5A. Figure 2.5D shows the scatterplot, with a smooth curve fitted. Panel E shows the plot of residuals versus `magnetic` that then results, with a smooth curve fitted that is designed to help in checking whether there is any remaining trend. Panel F plots the square root of absolute values of residuals against predicted chemical. There is now no suggestion of variance heterogeneity.

The code used to fit the line and extract the fitted values and residuals was:

```
ord <- order(DAAG::ironslag[["magnetic"]])
ironslag <- DAAG::ironslag[ord,]
slagAlpha.lm <- lm(chemical~magnetic, data=ironslag)
resval <- residuals(slagAlpha.lm)
fitchem <- fitted(slagAlpha.lm)
```

The code used to fit the loess curve and extract the fitted values and residuals was:

```
slag.loess <- loess(chemical~magnetic, data=ironslag, span=0.8)
```

```
resval2 <- slag.loess[[" residuals "]]
fitchem2 <- slag.loess[[" fitted "]]
```

Panels B, C, E and F were plotted using the function `scatter.smooth()`.

Where there is genuine heterogeneity of variance, and an accurate estimate of the variance at each data point is available, data points should be weighted proportionately to the reciprocal of the variance. Getting an estimate to within some constant of proportionality is enough. It may be possible to guess at a suitable functional form for the change in variance with x or (equivalently, since $y = a + bx$) with y. For example, the variance may be proportional to y.

2.5.2 The analysis of variance table

The analysis of variance table breaks the sum of squares for a linear model into two parts: a part accounted for by the deterministic component which is in this case the line, and a part attributed to the noise component or residual. For the lawn roller linear model, the analysis of variance table is:

```
roller.lm <- lm(depression ~ weight, data=DAAG::roller)
anova(roller.lm)
```

```
Analysis of Variance Table

Response: depression
          Df Sum Sq Mean Sq F value Pr(>F)
weight     1    658     658    14.5 0.0052
Residuals  8    363      45
```

The total sum of squares (about the mean) for the 10 observations is 1020.9 (= 658.0 + 362.9; we round to one decimal place). Including weight reduced this by 658.0, giving a residual sum of squares (RSS) equal to 362.9. The column headed `Mean Sq`, (*mean square*) gives a fair comparison. The mean square for `weight` is 658.0; this compares with a mean square of 45.4 for the residual.

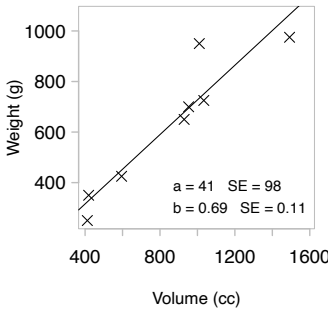
The degrees of freedom can be understood thus: with just two observations determining a line both residuals would be zero, yielding no information about the noise. Every additional observation beyond two yields one additional degree of freedom for estimating the noise variance. Thus with 10 points, 10-2 (= 8) degrees of freedom are available (in the residuals) for estimating the noise variance. (Where a line is constrained to pass through the origin, one point is enough to determine the line, and with 10 points the variance would be estimated with 9 degrees of freedom.)

This table has the information needed for calculating R^2 (also known as the "coefficient of determination") and adjusted R^2. The R^2 statistic is the square of the correlation coefficient, and is the sum of squares due to weight divided by the total sum of squares:

$$R^2 = \frac{658.0}{1020.9} = 0.64.$$

Compare this with

$$\text{adjusted } R^2 = 1 - \frac{362.9/8}{1020.9/9} = 0.60.$$



volume	weight
412	250
953	700
929	650
1492	975
419	350
1010	950
595	425
1034	725

Figure 2.6 Weights (g) versus Volume ($\leq$m^3) for eight softback books, with the fitted regression line added.

Adjusted R^2 takes into account the number of degrees of freedom, and is in general preferable to R^2. A small adjusted R^2 indicates large variability about the fitted line, relative to the variability explained by the line. It is useful as a measure of the extent to which the total scatter of outcome values is explained by the model. Neither R^2 nor adjusted R^2 is appropriate for comparisons between different studies, where the range of values of the outcome variable may be different. Both are likely to be largest in those studies where the range of values of the outcome variable is greatest. AIC and BIC, introduced in Section 1.7, are in general much preferable for model comparison purposes. See also Section 2.6.

2.5.3 Outliers, influence, and robust regression

The data displayed in Figure 2.6, with data shown on the right, are for a collection of eight softback books. Additionally, the figure shows the fitted regression line. Output from the regression calculations is:

```
softbacks.lm <- lm(weight ~ volume, data=DAAG::softbacks)
print(coef(summary(softbacks.lm)), digits=3)
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   41.372     97.559   0.424 0.686293
volume         0.686      0.106   6.475 0.000644
```

Figure 2.7 shows regression diagnostics. Code that gives the plots is:

```
plot(softbacks.lm, fg="gray")
```

For regression with one explanatory variable, plot A is equivalent to a plot of residuals against the explanatory variable. Plot C is designed for examining the constancy of the variance. Plot D plots residuals against leverage. Leverage, which depends only on explanatory variable values, is a measure of the potential for an observation to have a large influence in determining the regression line. Observations that are well away from the mean (or where there are multiple explanatory variables, the *centroid*) can exert a greater pull than those closer to the mean (or centroid). See Subsection 3.4.2 for further details

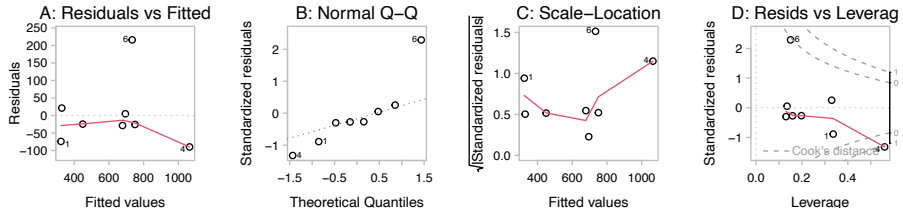


Figure 2.7 Diagnostic plots for Figure 2.6.

Contours of Cook's distances of 0.1 and 1 are shown in red. Cook's distance is a measure of *influence*; it measures the extent to which the line would change if the point were omitted. Observation 4's position at the extreme end of the range of x-values gives it a high *leverage*. Since its y-value is lower than would be predicted by the line, it pulls the line downward. This large leverage, combined with a residual that is the second largest, gives it the largest Cook's distance. Observation 6, which has the largest residual, has a much smaller leverage and, accordingly, a smaller Cook's distance.

Diagnostic plots, such as Figure 2.7, are not definitive. Rather, they draw attention to points that require further investigation. Here, with only eight points, it would not make sense to omit any of them, especially as points 4 and 6 are both, for different reasons, candidates for omission.

It may turn out, upon subsequent checking, that an outlier has arisen from a recording or similar error. Where an outlier seems a genuine data value, it is good practice to do the analysis both with and without the outlier. If retention of an apparent outlier makes little difference to the practical use and interpretation of the results, it is usually best to retain it in the main analysis. If an outlier that seems a genuine data value is omitted from the main analysis, it should be reported along with the main analysis, and included in graphs.

Robust regression

Robust regression offers a half-way house between including outliers and omitting them entirely. Rather than omitting outliers, it downweights them, reducing their influence on the fitted regression line. This has the additional advantage of making outliers stand out more strongly against the line. Available functions include `MASS::rlm()`, `MASS::lqs()`, and `robustbase::lmrob()`. Resistant methods, such as `MASS::lqs()` implements, are a subclass of robust methods that focus directly on ensuring that outliers do not contribute to the regression fit. For all methods, it can be important that residuals have an approximately symmetric distribution. See further, Section 3.4 and Exercise 16 in Section 3.11.

2.5.4 Standard errors and confidence intervals

Recall that since two parameters (the slope and intercept) have been estimated, the error mean square is calculated with $n - 2$ degrees of freedom. As a consequence,

Table 2.4
Observed and fitted values of `depression` *at the given* `weight` *value. SE (*standard error*) is the standard error for a new predicted value. SE.OBS is the standard error for a new observation.*

	Predictor `weight`	Observed `depression`	Fitted	SE	SE.OBS	
1	1.9	2	3.0	3.61	7.6	$\sqrt{3.61^2 + 6.74^2}$
2	3.1	1	6.2	2.98	7.4	$\sqrt{2.98^2 + 6.74^2}$
3	3.3	5	6.7	2.88	7.3	
. . . . 10	12.4	25	31.0	4.92	8.3	

the standard errors for the slope and for predicted values are calculated with $n - 2$ degrees of freedom. Both involve the use of the square root of the error mean square.

Confidence intervals and tests for the slope

A 95% confidence interval for the regression slope is

$$b \pm t_{.975}\text{SE}_b$$

where $t_{.975}$ is the 97.5% point of the t distribution with $n - 2$ degrees of freedom, and SE_b is the standard error of b.

For the `roller.lm` model, a 95% confidence interval may be calculated thus:

```
SEb <- coef(summary(roller.lm))[2, 2]
coef( roller.lm )[2]  + qt(c(0.025,.975 ),  8)*SEb
```

```
[1]  1.1  4.3
```

SEs and confidence intervals for predicted values

There are two types of predictions: prediction of points on the line, and prediction of a new data value. The SE estimates of predictions for new data values account for uncertainty in variation of individual points about the line. It is thus larger, perhaps much larger, than the SE for prediction of points on the line.

Table 2.4 shows expected values of the depression, with values of SE (for points on the line) and of SE.OBS (for new observations), for a range of roller weights. These may be calculated thus:

```
## Code to obtain fitted values and standard errors (SE, then SE.OBS)
 fit.with.se  <- predict(roller.lm,  se.fit =TRUE)
 fit.with.se $ se.fit                                    # SE
sqrt( fit.with.se  [[" se.fit "]]^2+fit.with.se$ residual.scale ^2)  # SE.OBS
```

The SE.OBS estimate accounts both for the standard error for the fitted value (estimated at 3.6 in row 1), and for the noise standard error (estimated at 6.74) for a new observation.

To calculate confidence intervals for predicted values, specify, for example:

```
predict( roller.lm ,  interval ="confidence",  level =0.95)
```

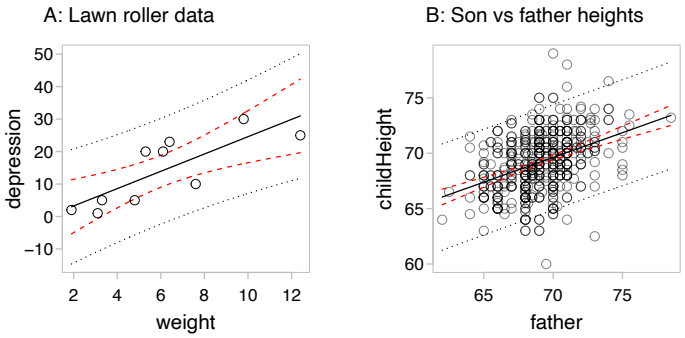


Figure 2.8 Panel A relates to the lawn roller data. Panel B relates to Galton's data that compares son's height with father's height. The two panels both show points, fitted line, 95% pointwise bounds for line (dashed, in red), and 95% pointwise bounds for predicted values.

To obtain confidence bounds for new predictions, replace `interval="confidence"` by `interval="prediction"`. If these are required for new x-values, use the argument `newdata` to supply the name of a data frame that has the new values for `weight`.

Figure 2.8A shows 95% pointwise confidence bounds, both for the fitted line, and for predictions of new data values. Figure 2.8B shows shows the equivalent points, and confidence bounds, for Galton's data that compares son's heights with father's heights. Both panels use the function `investr::plotFit()` to create the graphs simply and directly. Code for Figure 2.8B is:

```
galtonMales <- subset(HistData::GaltonFamilies, gender=="male")
galton.lm <- lm(childHeight~father, data=galtonMales)
investr :: plotFit(galton.lm, interval="both", col.conf="red", hide=FALSE,
                   col=adjustcolor('black',alpha=0.5), fg="gray")
```

For the plot shown in Panel B, summary information on the coefficients is:

```
             Estimate Std. Error t value
(Intercept)    38.36       3.31    11.6
father          0.45       0.05     9.3
```

A t-statistic that equals 9.34 and may seem large contrasts with a small adjusted R^2 statistic that equals 0.15. The fitted line explains only just over 0.15 of the variance of the heights of sons.

It bears emphasizing that the validity of these calculations depends crucially on model assumptions. Confidence bounds for the fitted line rely on normality for the sampling distribution of the slopes, while prediction bounds for future observations assume normally distributed heights.

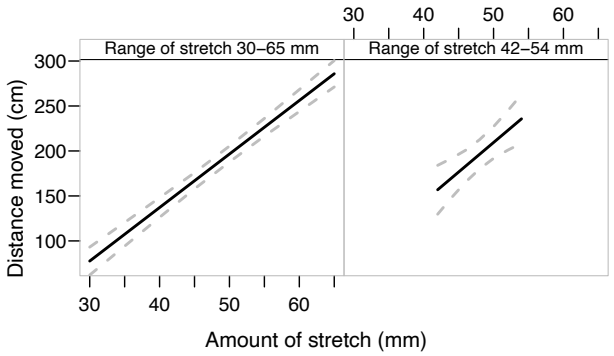


Figure 2.9 Data for the left panel (`elastic1`, 7 points) spanned a much wider range of values than that for the right panel (`elastic2`, 9 points). Even with the slightly larger number of points (9 as against 7, the right panel has much wider confidence bounds.

** Implications for design*

An emphasis of this subsection is that the choice of location of the x-values, which is a design issue, is closely connected with sample size considerations. Increasing the sample size is not the only, or necessarily the best, way to improve precision.

The estimated variance of the slope estimate is

$$\mathrm{SE}_b{}^2 = \frac{s^2}{n s_x^2}, \text{ where we define } s_x^2 = \frac{\sum_i (x_i - \bar{x})^2}{n}.$$

Here s^2 is the error mean square, i.e., s is the estimated SD for the population from which the residuals are taken. The expected value of $\mathrm{SE}_b{}^2$ is:

$$\mathrm{E}[\mathrm{SE}_b^2] = \frac{\sigma^2}{n s_x^2}$$

Now consider two alternative ways to reduce SE_b by a factor of 2:

- By fixing the configuration of x-values, but multiplying by 4 the number of values at each discrete x-value, s_x is unchanged. As n increases by a factor of 4, the expected value of SE_b^2 reduces by a factor of 4, and SE_b by a factor of 2.
- Alternatively, increasing the average separation between x-values by a factor of 2 will reduce SE_b by a factor of 2. Checking for linearity over the extended range of x-values is, however, important.

Reducing SE_b reduces, at the same time, the standard error of the fitted values. Figure 2.9 shows the effect of increasing the range of x-values (the code for both panels is a ready adaptation of the code for Figure 2.8). Both experiments used the same rubber band. The first experiment used a much wider range of values of x (= amount by which the rubber band was stretched). For the left panel of Figure 2.9, $s_x = 10.8$, while for the right panel $s_x = 4.3$.

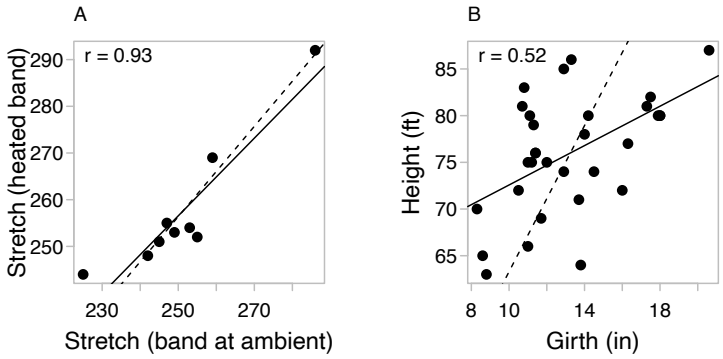


Figure 2.10 Each plot shows both regression lines, for y on x (solid line) and x on y (dotted line). In A, for the `DAAG::pair65` dataset, the lines are quite similar. (Note however that the most extreme point is driving the relatively high correlation.) In B, for two of the columns from the `trees` dataset, the correlation is smaller, with the result that the lines are more different.

2.5.5 There are two regression lines!

At this point, we note that there are two regression lines – a regression line for y on x, and a regression line for x on y. It makes a difference which is the explanatory variable, and which the dependent variable. Writing $b_{y|x}$ for the slope of the regression line of y on x, and $b_{x|y}$ for that for x on y, it follows that:

$$b_{y|x}b_{x|y} = r^2, \text{ where } r \text{ is the Pearson correlation}$$

When $r = 1$, so that the two lines coincide, $b_{x|y} = b_{y|x}^{-1}$. The two lines are quite different if the correlation is small.

Figure 2.10 illustrates the point for two different datasets. The correlation for the data on Panel A reduces sharply, to 0.78, if the point in the top right of the panel is left out.

An alternative to a regression line

There are yet other possibilities. A perspective that makes good sense for the seal organ growth data that will feature in Subsection 2.5.8 is that there is an underlying linear functional relationship. The analysis assumes that observed values of log(organ weight) and log(body weight) differ from the values for this underlying functional relationship by independent random amounts. The line that is obtained will lie between the regression line for y on x and the line for x on y. See Sprent (1966). Exercise 14 demonstrates a method for finding such a line.

2.5.6 Logarithmic and Power Transformations*

The discussion accompanying Figure 1.5 drew attention to the use of a logarithmic transformation to transform data with a strong right skew to give a more nearly

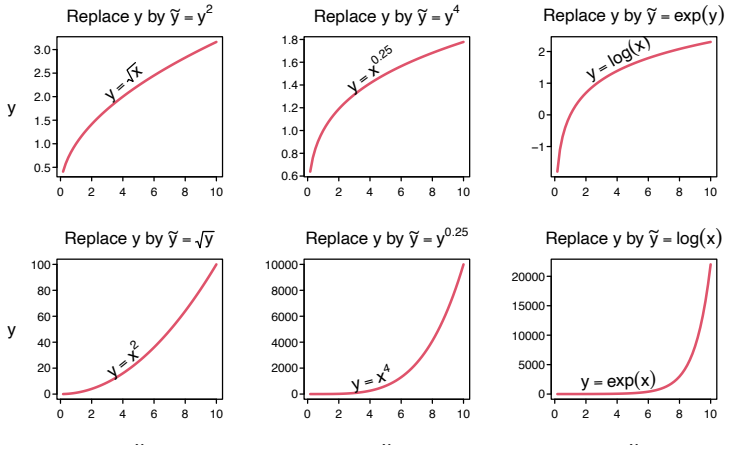


Figure 2.11 The above are some alternative response curves. The formula for $\tilde{y}$ gives the power family transformation of y that will make $\tilde{y}$ a linear function of x. Thus, if $y = \log(x)$, then taking $\tilde{y} = \exp(y)$ will make $\tilde{y}$ a linear function of x. Negative powers are also possible, e.g., $\tilde{y} = y^{-1}$.

symmetric distribution. A logarithmic transformation is often appropriate for size measurements (linear, surface, volume or weight) of biological organisms. If the ratio of largest to smallest data value is greater than 10, and especially if it is more than 100, then the logarithmic transformation should be tried as a matter of course. Check this advice against the response curves shown in Figure 2.11.

The logarithmic transformation belongs to the wider class of power transformations that includes square root and cube root transformations. The square root transformation is sometimes used for counts of rare events, and the cube root for rainfall data. The connection to the logarithmic transformation will be explained shortly. Figure 2.11 shows a number of response curves, and describes the particular transformation that would make the relationship linear.

We have so far mentioned only transformation of y. We might alternatively transform x, or transform both x and y.

*General power transformations — Box-Cox and Yeo-Johnson

For $\lambda \neq 0$, the power transformation replaces a value y by y^λ. The logarithmic transformation corresponds to $\lambda = 0$. In order to make this connection, the Box–Cox transformation (Box and Cox, 1964), makes a location and scale correction, so that the transformation is:

$$y(\lambda) = \frac{y^\lambda - 1}{\lambda}, \quad \text{if } \lambda \neq 0,$$
$$y(\lambda) = \log(y), \quad \text{if } \lambda = 0.$$

- If the small values of a variable need to be spread, make λ smaller.
- If the large values of a variable need to be spread, make λ larger.

Use `summary()` with the object returned by `car::powerTransform()` to get a

range of plausible values for λ. An alternative to `powerTransform()`, with fewer options, is `MASS::boxcox()`. Both `powerTransform()` and `boxcox()` accept as argument a regression formula or regression object where y is the outcome variable. An estimate is then given for λ that makes the distribution of residuals as close as possible (as measured by the likelihood function) to iid (independently and identically distributed) normal with mean 0.

The Yeo-Johnson family of transformations modifies and generalizes the Box-Cox family to handle data where the smallest value of y may be zero or negative. For non-negative values of y, it finds the Box-Cox transformation of $y + 1$. For negative values of y, it is the Box-Cox transformation of $|y| + 1$ with parameter $2 - \lambda$. Use the function `powerTransform()`, specifying `family = "yjPower"`, to obtain an optimal Yeo-Johnson transformation.

As an alternative to a model formula or `lm` regression object, a data frame or matrix can be supplied as argument. For use of a model formula as argument, the left hand side can be a matrix rather than a variable. Residuals from the regressions for all columns are then transformed, each column with its own transformation, so that the joint distribution is as close as possible to multivariate normal.

The use of `boxcox()` and `powerTransform()` is pursued in exercises at the end of the chapter. Subsection 3.3.3 has further details on the use of the function `powerTransform()`, with an example.

2.5.7 General forms of nonlinear response

Low order polynomial fits — quadratics or cubics — are often effective. Checks for quadratic as opposed to linear, and/or for cubic as opposed to quadratic, can be useful as part of a process of checking for variation for which the model has not accounted. Higher order polynomial fits are in general unsatisfactory. The trade-off for accurate approximation of observed data values becomes increasingly, as the order of polynomial increases, an erratic pattern of variation at intermediate data points. Spline approximations, used to fit a general form of curve with a slope that is constrained to change slowly in each local part of the curve, are in general a better choice. Section 4.4 will provide details.

2.5.8 Size and shape data – allometric growth

The logarithmic transformation is commonly important for morphometric data, i.e., for data on the size and shape of organisms. Figure 2.12 uses logarithmic scales to plot heart weight against body weight, for 30 seals that had been snared in trawl nets as an unintended consequence of commercial fishing (Stewardson et al., 1999).

For each animal, the data provide information at just one point in time, when they died. The data thus have limited usefulness for the study of growth profiles through time. At best, if conditions have not changed too much over the lifetimes of the animals in the sample, the data may provide an indication of the average of the population growth profiles. If, e.g., sample ages range from 1 to 10 years, it is pertinent to ask how food availability may have changed over the past 10 years,

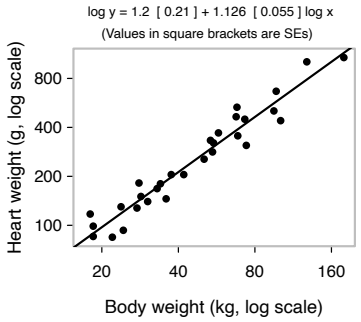


Figure 2.12 Heart weight versus body weight, for 30 Cape fur seals.

and whether this may have had differential effects on the different ages of animal in the sample.

The allometric growth equation

The allometric growth equation is

$$y = ax^b$$

where x may, e.g., be body weight and y heart weight. It may alternatively be written

$$\log y = \log a + b \log x,$$

i.e.,

$$Y = A + bX, \quad \text{where} \quad Y = \log y, \quad A = \log a, \quad \text{and } X = \log x.$$

Summary information on the coefficients of the regression line in Figure 2.12 is:

```
cfseal.lm <- lm(log(heart) ~ log(weight), data=DAAG::cfseal)
print(coef(summary(cfseal.lm)), digits=4)
```

```
            Estimate Std. Error t value  Pr(>|t|)
(Intercept)    1.204    0.21131   5.699 4.121e-06
log(weight)    1.126    0.05467  20.597 1.872e-18
```

The estimate of the exponent b (= 1.126) differs from 1.0 by 2.3 (= 0.126/0.05467) times its standard error. The relative rate of increase appears, then, slightly greater for heart weight than for body weight. (The t-statistic and p-value in the regression output relate to the comparison between b and zero, which is not of interest. Authors sometimes present p-values that focus on the comparison with zero, even though their interest is in the comparison with 1.0. See Table 10 and other similar tables in Gihr and Pilleri (1969, p. 43).) For an elementary discussion of allometric growth, see Schmidt-Nielsen (1984).

2.6 Empirical assessment of predictive accuracy

The *training data* estimate of predictive accuracy, derived by direct application of model predictions to the data from which the regression relationship was derived, gives in general an optimistic assessment. There is a mutual dependence between the model prediction and the data used to derive that prediction. It is because of this dependence that degrees of freedom for the variance are adjusted to take account of the number of parameters estimated.

The issue becomes a more serious concern in contexts, such as the classification models that will be discussed in Chapter 8 and Section 9.4, where no satisfactory theoretical adjustment for the dependence is available. The simple models discussed in the present chapter will be used as a context in which to demonstrate general approaches that address this issue.

2.6.1 The training/test approach, and cross-validation

An ideal is to assess the performance of the model on a new dataset. With a large dataset, it is good practice to split the data into two sets: the *training* set is for developing the model, and the *test* set is for testing predictions. The assumption is, then, that the test data can be treated as a random sample from the population to which results will be applied, or otherwise sampled in the same way. If there are too few data to make it reasonable to divide data into training and test sets then the method of cross-validation can be used.

Cross-validation – a tutorial example

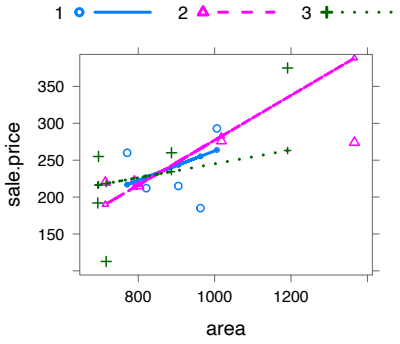
In cross-validation, the data are divided into k subsets where k is typically in the range 3 to 10. Each split between on of the k subsets and remaining data sets up a *fold* in which the subset is used as 'test' data, with remaining data (from the other k -1 subsets) used to fit ('train') the model. The predictive accuracy assessments from the k folds are combined to give a measure of the predictive performance of the model. This may be done for more than one measure of predictive performance. In what follows, the interest will be in sums of squared differences between y-values and the line to which they relate.

For simplicity, we will use 3-fold validation only, with a smallish dataset where a standard form of regression estimate might be preferred. Figure 2.13 relates to data on floor area and sale price for 15 houses in a suburb of Canberra (Australia), in 1999.

```
set.seed(29)          # Generate results shown
rand <- sample(rep(1:3, length=15))
## sample() randomly permutes the vector of values 1:3
for(i in 1:3) cat(paste0(i,":"), (1:15)[rand == i],"\n")
```

```
1: 3 4 8 9 11
2: 1 5 10 12 15
3: 2 6 7 13 14
```

The dataset was randomly divided into three groups of five observations, so that



Row	area	bedrooms	sale.price
1	694	4	192.0
2	905	4	215.0
3	802	4	215.0
	...	...	...
13	771	5	260.0
14	1006	5	293.0
15	1191	6	375.0

Figure 2.13 Graphical summary of 3-fold cross-validation for the house sale data in the dataset `houseprices`. Points were assigned a plot character (and color) according to the *fold* at which they made up the 'test' subset, with remaining points comprising the 'training set'. Line color identifies the line fitted to the corresponding training data.

there were three folds, i.e., test/training splits. In each fold, the sum of squares of the differences of test data y-values from the fitted line gave a contribution to a total the residual sum of squares, with degrees of freedom here equal to 15 — y-values are in each case independent of the fitted value that has been subtracted. Figure 2.13 is a visual summary, obtained by using the function `DAAG::CVlm()` with the default setting `plotit=TRUE`.

The following summary of the cross-validation results includes, for each fold, estimates of the mean square error.

```
fold 1
Observations in test set: 5
                2     6     7    13     14
area        905.0 963.0 821.0 771.0 1006.0
cvpred      243.6 255.2 226.9 216.9  263.8
sale.price  215.0 185.0 212.0 260.0  293.0
CV residual -28.6 -70.2 -14.9  43.1   29.2

Sum of squares = 8684    Mean square = 1737    n = 5

fold 2
Observations in test set: 5
                3    4   8      9     11
area        802.00 1366 714 1018.0 790.00
cvpred      216.81  388 190  282.5 213.16
sale.price  215.00  274 220  276.0 221.50
CV residual  -1.81 -114  30   -6.5   8.34

Sum of squares = 14083    Mean square = 2817    n = 5

fold 3
Observations in test set: 5
                1    5    10    12   15
area        694.0  716 887.0 696.0 1191
```

```
cvpred       216.3  218 234.5 216.5  263
sale.price   192.0  113 260.0 255.0  375
CV residual -24.3 -106  25.5  38.5  112

Sum of squares = 26421    Mean square = 5284    n = 5

Overall (Sum over all 5 folds)
  ms
3279
```

The `DAAG` function `CVlm()` can be used to give either or both of the above graphs as well as the printed summary. To obtain the estimate of the error mean square, the total of the sums of squares is divided by 15.

This gives

$$s^2 = (24\,351 + 20\,416 + 14\,241)/15 = 3934.$$

Actually, what we have is an estimate of the error mean square when we use only 2/3 of the data. Thus, we expect the cross-validated error to be on average larger than the error if all the data could be used. We can reduce the error by doing 10-fold rather than 3-fold cross-validation. Or we can do leave-one-out cross-validation, which for these data is 15-fold cross-validation. Contrast $s^2 = 3934$ with the estimate $s^2 = 2323$ from the model-based estimate in the regression output for the total data.[4]

The methodology can be used in a variety of contexts where the standard least squares theory no longer applies. Thus, in multiple regression, this may happen because there is use of a variable selection process. Valid estimates of the error mean square can be obtained by repeating the variable selection process at each cross-validation fold. Independence assumptions are as important as in standard forms of regression modeling.

2.6.2 Bootstrapping in regression*

We first indicate how resampling methods can be used to estimate the standard error of slope of a regression line. Recalling that the standard error of the slope is the standard deviation of the sampling distribution of the slope, we need a way to approximate this sampling distribution. One approach is to resample the observations or cases directly. For example, suppose five observations have been taken on a predictor x and response y:

$$(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5).$$

Generate five random numbers with replacement from the set $\{1, 2, 3, 4, 5\}$: 3, 5, 5, 1, 2, say. The corresponding resample is then

$$(x_3, y_3), (x_5, y_5), (x_5, y_5), (x_1, y_1), (x_2, y_2).$$

[4] `## Estimate of sigma^2 from regression output`
`summary(lm(sale.price ~ area, DAAG::houseprices))[["sigma"]]^2`

Note we are demonstrating only the so-called case-resampling approach. Another approach involves fitting a model and resampling the residuals. Details for both methods are in Davison and Hinkley (1997, Chapter 6). A regression line can be fit to the resampled observations, yielding a slope estimate. Repeatedly taking such resamples, we obtain a distribution of slope estimates, the bootstrap distribution.

As an example, consider the regression relating `sale.price` to `area` in the `houseprices` data. We will compute a bootstrap estimate of the standard error of the slope. For comparison purposes, note first the estimate given by `lm()`: .0664.

```
houseprices <- DAAG::houseprices
houseprices.lm <- lm(sale.price ~ area, houseprices)
print(coef(summary(houseprices.lm)),digits=2)
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    70.75     60.348     1.2    0.262
area            0.19      0.066     2.8    0.014
```

In order to use the `boot()` function, we need a function that will evaluate the slope for each of the bootstrap resamples:

```
houseprices.fn <-
  function (houseprices, index,
            statfun=function(obj)coef(obj)[2]){
            house.resample <- houseprices[index, ]
            house.lm <- lm(sale.price ~ area, data=house.resample)
            statfun(house.lm)    # slope estimate for resampled data
            }
```

We then use the function `boot::boot()` to make repeated calls to `houseprices.fn()`, with different randomly generated resamples from the data frame `houseprices`.

```
set.seed(1028)      # use to replicate the exact results below
library(boot)       # ensure that the boot package is loaded
## requires the data frame houseprices (DAAG)
(houseprices.boot <- boot(houseprices, R=999, statistic=houseprices.fn))
```

```

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = houseprices, statistic = houseprices.fn, R = 999)


Bootstrap Statistics :
    original  bias    std. error
t1*    0.188 0.00972       0.085
```

The output shows the original slope estimate, a bootstrap estimate of the bias of this estimate and the standard error estimate. The standard error was computed from the standard deviation of the 1000 (data+999 resamples) slope estimates.

By changing the statistic argument in the `boot()` function appropriately, we can compute standard errors and confidence intervals for fitted values. The change of

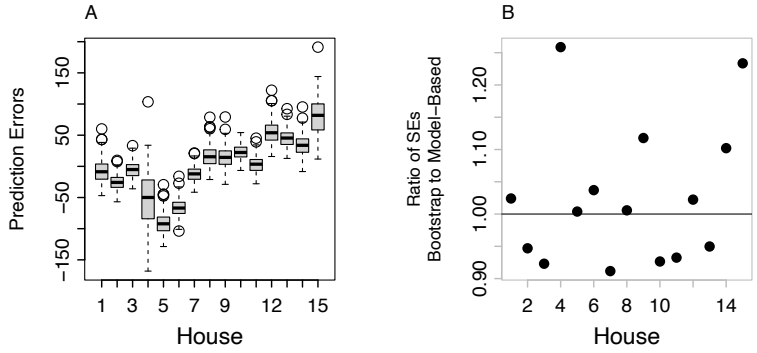


Figure 2.14 (A) Plot of bootstrap distributions of prediction errors for regression relating `sale.price` to `area`, each based on 200 bootstrap estimates of the prediction error. (B) Ratios of bootstrap prediction standard errors to model-based prediction standard errors.

argument can be passed (using the ... mechanism) as a an argument with that name to `boot()`. Thus, the following returns bootstrap predictions for house with a area of 1200 square feet, with the function `boot.ci()` then used to obtain a 95% confidence interval:

```
statfun1200 <− function(obj)predict(obj, newdata=data.frame(area=1200))
price1200.boot <− boot(houseprices, R=999, statistic=houseprices.fn,
statfun=statfun1200)
boot.ci(price1200.boot, type="perc") # "basic" is an alternative to "perc"
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = price1200.boot, type = "perc")

Intervals :
Level     Percentile
95%   (247, 368 )
Calculations and Intervals on Original Scale
```

Regression estimates for each resample can be used to compute predicted values at the original values of the predictor, and compared with model-based predictions. Repeating this procedure a number of times (here $R = 199$) gives a distribution of the prediction errors at each observation. Figure 2.14A displays a prediction error plot for the `houseprices` data.[5] Note the large variability in the prediction error

[5]
```
## Bootstrap estimates of prediction errors of house prices
houseprices2.fn <- function (houseprices, index){
house.resample <- houseprices[index, ]
house.lm <- lm(sale.price ~ area, data=house.resample)
houseprices$sale.price - predict(house.lm, houseprices)
  ## prediction errors from resamples
}
n <- length(houseprices$area);  R <- 199
houseprices2.boot <- boot(houseprices, R=R, statistic=houseprices2.fn)
house.fac <- factor(rep(1:n, rep(R, n)))
```

associated with observation 4. Figure 2.14B shows ratios of the bootstrap standard errors to the model-based standard errors.[6] The standard errors that are calculated from the bootstrap output are generally larger than those given by the `lm` regression model, and should perhaps be used in preference.

We can also compute an estimate of the aggregate prediction error, as an alternative to the cross-validation estimate obtained in the previous subsection. There are a number of ways to do this, and some care should be taken. We refer the interested reader to Davison and Hinkley (1997, Section 6.4).

Commentary

The cross-validation and bootstrap estimates of mean square error require the assumption that the variance is homogeneous. The estimate of predictive error applies only to data that have been sampled in the same way as the data that are used as the basis for the calculations. It assumes that the `target` population will be highly comparable to the `source` population that generated the data.

Here, the estimate of predictive accuracy applies only to 1999 house prices in the same city suburb. Such standard errors may have little relevance to the prediction of house prices in another suburb, even if thought to be comparable, or to prediction for more than a very short time into the future. This point has relevance to the use of regression methods in business "data mining" applications. A prediction that a change will make cost savings of $500 000 in the current year may have little relevance to subsequent years. The point has special force if changes will take years rather than months to implement.

A realistic, though still not very adequate, assessment of accuracy may be derived by testing a model that is based on data from previous years on a test set that is formed from the current year's data. Predictions based on the current year's data may, if other features of the business environment do not change, have a roughly comparable accuracy for prediction a year into the future. If the data series is long enough, we might, starting at a point part-way through the series, compare predictions one year into the future with data for that year. The estimated predictive accuracy would be the average accuracy for all such predictions. A more sophisticated approach might involve incorporation of temporal components into the model, i.e., use of a time series model. See Maindonald (2003) for more extended commentary on such issues.

2.7 One- and two-way comparisons

This section introduces examples where one or more factors, i.e., categorical effects, have the role of explanatory variables.

[6]
```
## Ratios of bootstrap to model-based standard errors
bootse <- apply(houseprices2.boot$t, 2, sd)
usualse <- predict.lm(houseprices.lm, se.fit=TRUE)$se.fit
plot(bootse/usualse, xlab="House",
ylab="Ratio of SEs\nBootstrap to Model-Based", pch=16)
```

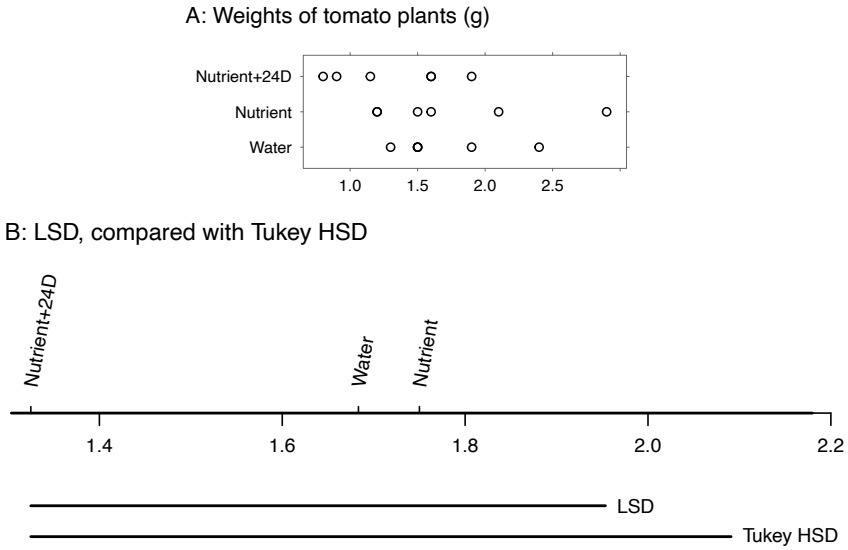


Figure 2.15 Panel A compares weights of tomato plants, after two months of the three treatments. In Panel B means that differ by more than the LSD (least significant difference) are different, at the 5% level, in a t-test that compares the two means. Tukey's Honest Significant Difference (HSD) takes into account the number of means that are compared. See the text for details.

2.7.1 One-way comparisons

Figure 2.15A displays data from a one-way unstructured comparison between three treatments. The weights of the plants were measured after two months on the respective treatments: water, concentrated nutrient, and concentrated nutrient plus the selective herbicide 2,4-D.

```
tomato <- data.frame(Weight = c(1.5, 1.9, 1.3, 1.5, 2.4, 1.5,    # Water
                                1.5, 1.2, 1.2, 2.1, 2.9, 1.6,    # Nutrient
                                1.9, 1.6, 0.8, 1.15, 0.9, 1.6), # Nutrient+24D
  trt = factor(rep(c("Water", "Nutrient", "Nutrient+24D"), c(6, 6, 6))))
## Make `Water` the first level of trt.  In aov or lm calculations, it is
## then taken as the baseline or reference level.
tomato$trt <- relevel(tomato$trt, ref="Water")
```

The above has three treatments only from the four that are in the data frame `DAAG::tomato`.

The strip plots display "within group" variability, as well as giving an indication of differences among the group means. Variances seem similar for the three treatments.

Summary code is:

```
## A: Weights of tomato plants (g)
library(lattice, quietly=TRUE)
gph <- stripplot(trt~Weight, aspect=0.35, scale=list(tck=0.6), data=tomato)

## B: Summarize comparison between LSD and Tukey's HSD graphically
tomato.aov <- aov(Weight ~ trt, data=tomato)
```

```
DAAG::onewayPlot(obj=tomato.aov)
```

Figure 2.15 compares two different statistics that, for data where there is more than one comparison, are commonly used as measures of the difference in treatment means that is 'significant' at the 5% level:

- For the 5% least significant difference (LSD), the 5% is the proportion of all comparisons between treatment means in which the difference can be expected to exceed the LSD.
- For the 5% honest significant difference (HSD), the 5% is the proportion of experiments in which the maximum difference between treatments can be expected to exceed the HSD. Ignoring changes in degrees of freedom and possible associated changes in the standard error, the HSD will increase as the number of treatment groups that are to be compared increases.

The LSD is overly lax, while the HSD may be overly conservative. Note the assumption that the standard error of difference is the same for all treatment comparisons. As all three treatments have the same number of observations, it is enough for the variance to be the same for all treatments.

The function `BHH2::anovaPlot()` offers alternative graphical perspectives:

```
BHH2::anovaPlot(tomato.aov)
```

Two dot plots are presented, one for the residuals and one for the treatment means, suitably rescaled so that they are comparable to the residuals. Such plots, advocated by Box, Hunter, et al. (2005), provide a clear visualization of the strength and direction of the statistical evidence, while also highlighting possible problems, such as outliers, in the data. Braun (2012) describes some graphical alternatives.

The analysis of variance table

The analysis of variance table is given by the `anova()` function, thus:

```
## Do analysis of variance calculations
anova(tomato.aov)
```

```
Analysis of Variance Table

Response: Weight
          Df Sum Sq Mean Sq F value Pr(>F)
trt        2   0.63   0.314     1.2   0.33
Residuals 15   3.91   0.261
```

The residual mean squared error is 0.261. The mean square for the the differences of the 3 group means from the overall mean accounts for 2 degrees of freedom. Each treatment contributes 6-1=5 degrees of freedom to the pooled or residual sum of squares, giving $3 \times 5 = 15$ d.f. in all. Note that 2 (for trt) plus 15 (for residuals) equals 17, which is one less than the number of observations. Estimation of the overall mean accounts for the remaining degree of freedom.

The `Mean Sq` ("mean square") column has estimates of between sample (`trt`) and within sample variability (`Residuals`). The between sample variance can be

calculated by applying the function `var()` to the vector of treatment means, then multiplying by the common sample size, in this case 6. The within sample variability estimate is, effectively, a pooled variance estimate for the 3 treatments. Each mean square is the result from dividing the `Sum Sq` ("sum of squares") column by the appropriate degrees of freedom.

In the absence of systematic differences between the sample means, the two mean squares have the same expected value, with a ratio (the F-statistic) near 1. Systematic differences between the sample means will add extra variation into the treatment mean square, with no effect on the residual mean square, giving an expected F-statistic that is larger than 1. Where the evidence for differences is convincing, interest will then turn to teasing out the nature of those differences. A strategy that is sometimes adopted is to use a preliminary F-test to decide whether to apply the LSD criterion.

In the output above, the F-statistic is 1.2, on 3 and 8 degrees of freedom, with $p = 0.33$. There is no convincing indication that there are indeed differences among the means. With four treatments, as in the complete `DAAG::tomato` dataset, there are six comparisons. The case for accounting for the number of of comparisons made is then very strong, whether or not starting with an overall analysis of variance F-test.

Other multiple comparison tests

Tukey's HSD is one of a number of criteria that may be used for comparisons when more than two group means are compared. See `?p.adjust`, and tests available in the package *agricolae*, for other possibilities.

The discussion will now move on from examples where the number of comparisons that are of interest is relatively small to cases where the number of comparisons is large — hundreds, or thousands, or tens of thousands. The proliferation of p-values then allows the calculation of a false discovery rate. Subsection 2.7.2 that follows will be a brief diversion, before returning to consider the new possibilities that severe multiplicity can offer.

Multiple range tests should not be used when there is an ordering in the explanatory variable that allows the response to be modeled as a line or as a curve. The simulations in Subsection 2.7.2 that now follows illustrate the loss of power that can result.

There is a large literature on multiple range tests and simultaneous inference. See Hothorn et al. (2008), and references given on the help page `?p.adjust`. See also the CRAN *SocialSciences* task view.

2.7.2 Regression versus qualitative comparisons – issues of power

Figure 2.16 plots results from 200 simulations from a straight line model with slope 0.8, SD = 2, and 4 replications at each of the levels 1, 2, ... 5. Both axes use a scale of $\log(p/(1-p))$. The vertical axis shows the p-values for a test for linear trend, while the horizontal axis shows p-values for an analysis of variance test for qualitative differences. Most points (for the simulation shown, 89%) lie below the

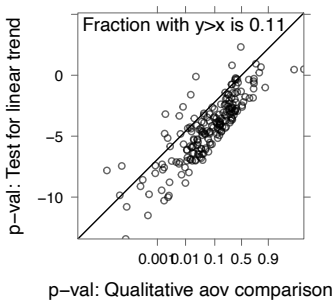


Figure 2.16 The plot compares p-values for a test for linear trend with p-values from an analysis of variance test for qualitative differences, in each of 200 sets of simulated data. The line $y = x$ is superimposed.

line $y = x$, showing the greater power of a test for linear trend. (Note that the test for linear trend is equivalent to using `aov()` to test for a linear contrast when the explanatory term is an ordered factor.)

The function `DAAG::simulateLinear()` automates such simulations. Write the p-values for a test for linear trend as p_l, and the p-values for the analysis of variance test for qualitative differences as p_a. Specifying `type="density"` gives overlaid plots of the densities for the two sets of p-values, both on a scale of $\log(p/(1-p))$, together with a plot of the density of $\log(p_l/(1-p_l)) - \log(p_a/(1-p_a))$. As the data are paired, this last plot is the preferred form of comparison.

Fitting a line (or a curve) allows interpolation between successive levels of the explanatory variable. It may be reasonable to hazard prediction a small distance beyond the range of the data. The pattern of response may give scientific insight.

2.7.3 Severe multiplicity — the false discovery rate*

The dataset `DAAG::coralPval` that is the subject of the following discussion was generated using a microarray gene expression technology. Microarrays are now increasingly being replaced by the more direct measurements of gene activity in the cell that the RNA-Seq technology provides. In either case, a single experiment may yield information on thousands, or tens of thousands, of genes. The present data are from experimental work that was designed to compare gene expression, for the 3042 genes investigated, between two life-stages of coral — the pre-settlement free-swimming stage, and post-settlement. Each of the full complement of six panels (two only are shown in Figure 2.17) had 3072 spots; this included 30 blanks. Where there was an increase, the spot should be fairly consistently red, or reddish, over all six panels. Where there was a decrease, the spot should be fairly consistently green, or greenish. Results from the six sets of comparisons were used to generate 3042 p-values, one for each of 3042 sets of spots.

The methodology that will be described has wide application, to any form of comparison that generates large numbers of p-values — hundreds, or thousands, or more. The multiplicity of p-values allows inferences that an individual p-value does not provide. It allows the estimation of a false discovery rate (FDR).

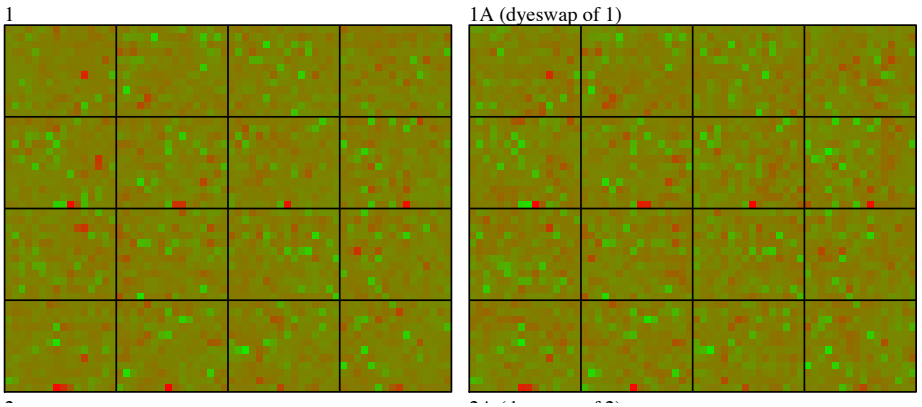


Figure 2.17 This false color image shows the intensity of the post signal (red), relative to the pre signal (green), for the first two of six half-slides ('panels') in a two channel microarray gene expression experiment. Use of one dye-swap pair per slide was designed to allow adjustment for any systematic red-green bias.

*Microarrays and alternatives — technical note

In the experimental procedure and subsequent processing that led to the plots shown in Figure 2.17, the slides are first printed with probes, with one probe per spot, each designed to check for the expression of one gene. The two samples carry labeling with separate fluorescent dyes so that when later a spot "lights up" under a scanner, the relative intensities of the two dye frequencies will provide a measure of differences in the signal intensity.

After labeling the separate samples, mixing them, and wiping the mixture over the slide or half-slide, and various laboratory processing steps, a scanner was used to determine, for each spot, the intensities generated from the two samples. Various corrections are then necessary, leading finally to the calculation of logarithms of intensity ratios. Essentially, it is logarithms of intensity ratios that are shown in Figure 2.17.

For further information on the statistical analysis of microarray data, see Smyth (2004). With suitable pre-processing of the data, the methods carry over to the analysis of RNA-Seq data. See Law et al. (2014). For background on the coral data, see Grasso et al. (2008).

The false discovery rate (FDR)

The object `DAAG::coralPval` has 3072 p-values from the gene expression data represented in Figure 2.17. The following calculates, for several different thresholds `pcrit` $= p_{crit}$, the total number of genes detected as differentially expressed with threshold as the threshold:

```
coralPval <- DAAG::coralPval
pcrit <- c(0.05, 0.02, 0.01, 0.001)
under <- sapply(pcrit, function(x)sum(coralPval<=x))
```

The numbers expected under the null hypothesis, in each case, are:

```
expected <− pcrit∗length(coralPval)
```

These numbers can be set out in a table that allows a comparison of the implications of choosing one threshold rather than another.

```
fdrtab <− data.frame(Threshold=pcrit, Expected=expected,
Discoveries=under, FDR=round(expected/under, 4))
print(xtable::xtable(fdrtab), include.rownames=FALSE, hline.after=FALSE)
```

Threshold	Expected	Discoveries	FDR
0.05	153.60	1310	0.12
0.02	61.44	1068	0.06
0.01	30.72	900	0.03
0.00	3.07	491	0.01

The column headed FDR is just the number of detections ('discoveries') expected under the null hypothesis, divided by the actual number detected. Although often described as an adjusted p-value, the result of the adjustment is not a p-value, but an estimate of the false discovery rate. For the false discovery rate to equal 0.05, the unadjusted p-value threshold should be set somewhere between 0.01 and 0.02.

The Benjamini-Hochberg method for adjusting p-values relies, in essence, on the argument just given. Rather than finding an unadjusted p-value threshold, it is however more straightforward to work directly with adjusted values, calculated as will now be described. After sorting the p-values from smallest to largest, the calculation is:

$$p_{adj[i]} = \frac{m}{i} p_i\,; \quad i = 1, 2, \ldots m$$

A further tweak is to set each $p_{adj[i]}$ to the smallest value, if any, that appears later in the sequence. This ensures that $p_{adj[i]}$ is a monotonic function of p_i. (Also, any value that is greater than 1.0 is set to 1.) The function `p.adjust()` (*stats* package in base R), can be used (specify `method="BH"`) to do the adjustments, thus:

```
fdr <− p.adjust(coralPval, method="BH")
```

Here are numbers that fall under thresholds 0.05, 0.04, 0.02, and 0.01:

```
fdrcrit <− c(0.05, 0.04, 0.02, 0.01)
under <− sapply(fdrcrit, function(x)sum(coralPval≤x))
setNames(under, paste(fdrcrit))
```

```
0.05 0.04 0.02 0.01
1310 1234 1068  900
```

The FDR for a cutoff of 0.05 is a composite value, with some genes that fall under this threshold having a FDR much greater than 0.05, and many more having an FDR that is much less. The discussion that now follows shows how this composite FDR can be broken apart. Take p_{45} as the false discovery rate for genes in the range $0.04 <$ fdrcrit ≤ 0.05. Then the 1310 genes with fdrcrit ≤ 0.05 are comprised thus:

- 1310 - 1234 = 76 genes with an average FDR of p_{45}
- 1234 genes with an average FDR of 0.04

Then

$$p_{45} \times 76 + 0.04 \times 1234 = 0.05 \times 1310$$

Solving for p_{45} yields, rounded to two decimal places

$$p_{45} = 0.21$$

As with use of the $p \leq 0.05$ criterion for a single p-value, it is tempting to place greater weight than is warranted on an FDR statistic that falls just under 0.05.

The average estimated false discovery rate for genes with $0.01 <$ fdrcrit ≤ 0.02, is:

$$\frac{0.02 \times 1068 - 0.01 \times 900}{1068 - 900} = 0.07$$

This line of argument can be combined with the assumption of a smooth change in the FDR to provide a local false discovery rate estimate. These bear the same relationship to the false discovery rate that a density, for the relevant distribution, bears to the corresponding p-value, i.e., to an area in the tail or tails of the distribution. The function `locfdr::locfdr()` is designed to provide, as well as estimates of local FDRs, an estimate of the proportion of p-values that correspond to cases where the null hypothesis is true. Estimates of the proportion of nulls may vary widely, depending on the method used.

As noted, the false discovery rate estimates are not p-values in the conventional sense. They give a frequency based probability that a detected difference is a real difference – information that p-values do not provide. Why insist on working with p-values when there is a better alternative? When m is large, the estimate provided has high statistical accuracy.

The `p.adjust()` FDR estimate remains valid in a wide range of contexts where p-values are positively correlated. Also available is `method="BY"`, designed for contexts where there may be quite general dependence structures. This gives a very conservative adjustment. Other available adjustment methods (see `?p.adjust)` are more in the style of p-values. .

The number 3072 of p-values is small relative to much other expression array data. The experimental data that will be considered in Section 9.5, from an experiment with RNA-Seq data, yielded 18658 p-values for each comparison of interest.

2.7.4 Data with a two-way structure (two factors)

Consider now data from an experiment that compared wild type (`wt`) and genetically modified rice plants (`ANU843`), each with three different chemical treatments. A first factor relates to whether `F10` or `NH4Cl` or `NH4NO3` is applied. A second factor relates to whether the plant is wild type (`wt`) or `ANU843`.

There are 72 sets of results, i.e., two types (`variety`) × three chemical treatments (`fert`) × 6 replicates, with the setup repeated across each of two blocks (`Block`). Figures 2.18A and 2.18B show alternative perspectives on these data.

Figure 2.18B shows a large difference between `ANU843` and wild type (`wt`) for

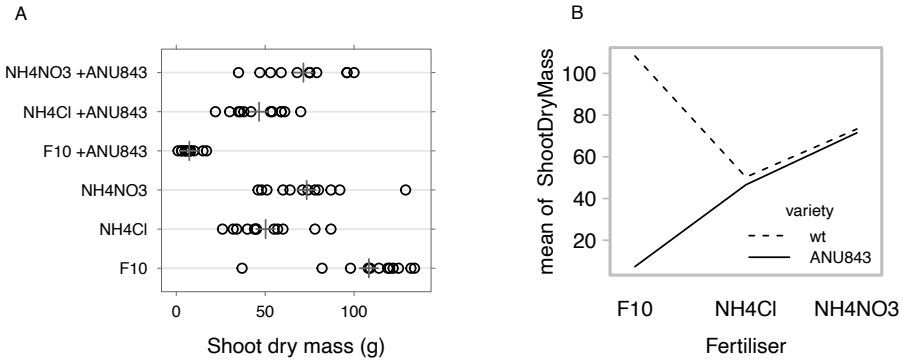


Figure 2.18 Both panels are for rice shoot dry mass data. Panel A shows a one-way strip plot, with different strips for different treatment regimes. Treatment means are shown with a large +. The interaction plot in Panel B shows how the effect of fertilizer (the first factor) changes with variety (the second factor). Data relate to Perrine et al. (2001).

the F10 treatment. For the other treatments, there is no detectable difference. A two-way analysis will show a large interaction.[7]

Note, finally, that the treatments were arranged in two blocks. In general, this has implications for the analysis. This example will be discussed again in Chapter 4, where block effects will be taken into account.

2.7.5 Presentation issues

The discussion so far has treated all comparisons as of equal interest. Often they are not. There are several possibilities:

- Interest may be in comparing treatments with a control, with comparisons between treatments of lesser interest.
- Interest may be in comparing treatments with one another, with any controls used as a check that the order of magnitude of the treatment effect is pretty much what was expected.
- There may be several groups of treatments, with the chief interest in comparisons between the different groups.

Any of these situations should lead to specifying in advance the specific treatment comparisons that are of interest.

Often, however, scientists prefer to regard all treatments as of equal interest. Results may be presented in a graph that displays, for each factor level, the mean and its associated standard error. Alternatives to displaying bars that show the standard error may be to show a 95% confidence interval for the mean, or to show the standard deviation. Displaying or quoting the standard deviation may be appropriate when the interest is, not in comparing level means, but in obtaining an idea of the extent to which the different levels are clearly separated. In any case:

[7]
```
## Simplified version of code, Panel B only
with(rice, interaction.plot(fert, variety, ShootDryMass, xlab="Fertiliser"))
```

Table 2.5 *Each tester made two firmness tests on each of five fruit.*

Fruit	Tester	Firmness	Mean
1	1	6.8, 7.3	*7.05*
2	1	7.2, 7.3	*7.25*
3	1	7.4, 7.3	*7.35*
4	1	6.8, 7.6	*7.2*
5	1	7.2, 6.5	*6.85*
6	2	7.7, 7.7	*7.7*
7	2	7.4, 7.0	*7.2*
8	2	7.2, 7.6	*7.4*
9	2	6.7, 6.7	*6.7*
10	2	7.2, 6.8	*7.0*

- For graphical presentation, use a layout that reflects the data structure, i.e., a one-way layout for a one-way data structure, and a two-way layout for a two-way data structure.
- Explain clearly how error bars should be interpreted – ± SE limits, ± 95% confidence interval, ± SED limits, or whatever. Or if the intention is to indicate the variation in observed values, the SD (standard deviation) may be appropriate.
- Where there is more than one source of variation, explain what source(s) of 'error' is/are represented. It is pointless and potentially misleading to present information on a source of error that is of little or no interest, e.g., on analytical error when the relevant error for the treatment comparisons that are of interest arises from fruit to fruit or tree to tree variation.

2.8 Data with a nested variation structure

Ten apples are taken from a box. A randomization procedure assigns five to one tester, and the other five to another tester. Each tester makes two firmness tests on each of their five fruit. Firmness is measured by the pressure needed to push the flat end of a piece of rod through the surface of the fruit. Table 2.5 gives the results, in N/m^2.

For comparing the testers, we have five experimental units for each tester, not ten. One way to do a t-test is to take means for each fruit. We then have five values (means, italicized) for one treatment, that we can compare with the five values for the other treatment.

If the data structure is ignored, and ten values for one tester are compared with ten values for the other tester (the pretense is that we have ten experimental units for each tester), the analysis will suggest that the treatment means are more accurate than is really the case. It is likely to underestimate the standard error of the treatment difference.

2.8.1 Degrees of freedom considerations

For comparison of two means when the sample sizes n_1 and n_2 are small, it is important to have as many degrees of freedom as possible for the denominator of

the t-test. A small bias in a calculated SED may be a reasonable trade-off for extra degrees of freedom.

The same considerations arise in the one-way analysis of variance, and we pursue the issue in that context. It is illuminating to plot out, side by side, say 10 SEDs based on randomly generated normal variates, first for a comparison based on 2 d.f., then 10 SEDs for a comparison based on 4 d.f., etc.

A formal statistical test is thus unlikely, unless the sample is large, to detect differences in variance that may have a large effect on the result of the test. It is therefore necessary to rely on judgment. Both past experience with similar data and subject area knowledge may be important. In comparing two treatments that are qualitatively similar, differences in the population variance may be unlikely, unless the difference in means is at least of the same order of magnitude as the individual means. If the means are of similar magnitude, then it is reasonable to expect that the variances will be similar, though this is by no means inevitable,

If the treatments are qualitatively different, then differences in variance may be expected. Thus in weed control experiments there will be few weeds in all plots where there is effective weed control, and thus little variation. In control plots, or for plots given ineffective treatments, there may be huge variation.

If there do seem to be differences in variance, it may be possible to model the variance as a function of the mean. It may be possible to apply a variance-stabilizing transformation. Or the variance may be a smooth function of the mean. Otherwise, if there are just one or two degrees of freedom per mean, use a pooled estimate of variance unless the assumption of equal variance seems clearly unacceptable.

2.8.2 General multi-way analysis of variance designs

Generalization to multi-way analysis of variance raises a variety of new issues. If each combination of factor levels has the same number of observations, and if there is no structure in the *error* (or *noise*), the extension is straightforward. The extension is less straightforward when one or both of these conditions are not met. For unbalanced data from designs with a simple error structure, it is necessary to use the `lm()` (linear model) function. The functions `nlme::lme()` and `lme4::lmer()` are both able to handle problems where there is structure in the error term. Data from unbalanced as well as from balanced designs can be handled. Chapter 7 will take up the discussion of models of this type.

2.9 Bayesian estimation – further commentary and approaches

The account of Bayesian methods given in this text should be enough to give a sense of the broad difference in perspective that they offer, relative to the role that p-values have in frequentist approaches. As well as providing for the calculation of Bayes Factors and/or posterior probability distributions for one- and two-sample t-tests, the *BayesFactor* package has provision for tests that relate to linear models more generally. Further possibilities include tests for proportions (assuming observations are independent with the same probability), contingency tables (under

the independence assumption relevant to the sampling design), and for correlations (assuming normality for y given x). Readers are encouraged to work through, as a minimum, the first two of the vignettes that come with the *BayesFactor* package. Calculation of Bayes Factors in other contexts, and the use of other forms of summary statistics, will in general require resort to the Markov Chain Monte Carlo (MCMC) simulation approach that will be demonstrated in Subsection 2.9.3. To get details of the wide range of R packages that implement Bayesian methods, check the CRAN task view for Bayesian inference.[8]

*2.9.1 *Bayesian estimation with normal priors and likelihood*

A relatively simple example is that of a normal likelihood (as considered in Subsection 1.4.6) where the unobserved true mean is now also assumed to have a normal distribution, now with mean μ_0 and variance σ_0^2. The posterior density of the mean is then normal with

$$\text{mean} = \frac{n\bar{y} + \mu_0\,\sigma^2/\sigma_0^2}{n + \sigma^2/\sigma_0^2}; \text{variance} = \frac{\sigma^2}{n + \sigma^2/\sigma_0^2}.$$

This assumes that σ^2 is known; the sample variance can be used as an estimate. An alternative is to put a prior distribution on this parameter as well.

In problems where the model has many parameters, direct evaluation of the relevant posterior distributions for parameters of interest is commonly computationally intractable. Fortunately, a simulation technique called Markov Chain Monte Carlo (or MCMC) will usually give effective approximations to these posterior distributions. Calculations must run for long enough that the posterior distribution reaches a steady state that is independent of the starting values of parameters.

Exercise 16a at the end of the chapter demonstrates the simulation of a finite state Markov chain. Subsection 2.9.3 will use Bayesian MCMC for straight line regression.

2.9.2 Further comments on Bayes Factors

The Sellke et al., 2001 upper limit

For $p < e^{-1}$, Sellke et al. (2001) give an upper bound, applying to a wide class of priors, on the Bayes Factor that corresponds to a given p-value. The posterior odds in favor of the alternative are no greater than:

$$\frac{R}{-eplog(p)} \tag{2.4}$$

For $p = 0.05$, this upper bound is $2.46R$, while for $p = 0.01$ it is $7.99R$.

The result applies to a class of frequently used priors for the effect size that include the normal and t-distributions, with the mean centered at the (point) null hypothesis. The requirement is that for any fixed tail probability p_0 the distribution of $p/p_0|p < p_0$ is non-increasing as the test statistic increases in absolute value.

[8] `https://cloud.r-project.org/web/views/Bayesian.html`

Held and Ott (2018) discuss other bounds that are available, including bounds that account for sample size. In their account, the bound given is a lower bound in favor of the null, which better reflects what priors that are centered at the null are designed to do.

*A note on the Bayesian Information Criterion

For models where the number of observations is large relative to the number of parameters estimated, an argument can be made for using the the Bayesian Information Criterion function `BIC()` statistic that was introduced in Subsection 1.7.1 as a starting point for calculating a Bayes Factor. Given BIC statistics `BIC1` and `BIC2`, the factor favoring the second model over the first is taken to be:

$$\exp(\frac{\text{BIC2} - \text{BIC1}}{2})$$

This is the Bayes Factor obtained when the difference between the BIC2 and the BIC1 model is assumed to have a "unit information" prior. In the case of one- and two-sample comparisons, the prior is a normal distribution that is centered on the mean of the data, with variance for the difference δ that estimated for a single observation. See, e.g., Neath and Cavanaugh (2012). Such a prior is unreasonably favorable to the alternative when the sample size is small. Note that because the mean is centered on the mean of the data, the conditions under which the Sellke et al. (2001) upper limit apply are violated.

The following uses a result from Rouder et al. (2009) that, for two-sided p-values from a one-sample t-test, gives the corresponding BIC based Bayes Factor. The Bayes Factor that is derived using `BayesFactor::ttest.tstat()` as in shown in Figure 1.24 is given for comparison.

```
pval <- c(.05,.01,.001); np <- length(pval)
Nval <- c(4,6,10,20,40,80,160); nlen <- length(Nval)
## Difference in BIC statistics, interpreted as Bayes factor
t2BFbic <- function(p,N){t <- qt(p/2, df=N-1, lower.tail=FALSE)
                         exp((N*log(1+t^2/(N-1))-log(N))/2)}
bicVal <- outer(pval, Nval, t2BFbic)
## Bayes factor, calculated using BayesFactor::ttest.tstat()
t2BF <- function(p, N){t <- qt(p/2, df=N-1, lower.tail=FALSE)
          BayesFactor:: ttest.tstat (t=t, n1=N, simple=TRUE, rscale = "medium")}
BFval <- matrix(nrow=np, ncol=nlen)
for(i in 1:np)for(j in 1:nlen) BFval[i,j] <- t2BF(pval[i], Nval[j])
cfVal <- rbind(BFval, bicVal)[c (1,4,2,5,3,6),]
dimnames(cfVal) <- list(
  paste(rep(pval,rep(2,np)), rep(c("- from ttest.tstat ", "- from BIC"),np)),
        paste0(c("n=",rep("",nlen-1)),Nval))
round(cfVal,1)
```

```
                              n=4     6    10   20   40   80  160
0.05 - from ttest.tstat       2.4   2.1   1.8  1.4  1.1  0.8  0.6
0.05 - from BIC               9.6   5.1   3.0  1.8  1.2  0.8  0.5
0.01 - from ttest.tstat       7.0   6.9   6.2  5.1  4.1  3.1  2.3
0.01 - from BIC              76.5  31.4  15.3  8.0  5.0  3.3  2.3
0.001 - from ttest.tstat     32.3  40.0  40.9 36.6 30.5 24.2 18.4
0.001 - from BIC           1606.1 464.0 175.7 77.1 43.7 27.8 18.7
```

For $p = 0.001$ and $n = 4$, the ratio of 1606:1 is much larger than the 999:1 ratio that results from wrongly interpreting the null hypothesis 1 in 1000 value as giving a relative probability. With large samples, the data overwhelms the prior. The estimates are then very similar to those returned by `BayesFactor::ttest.tstat()` with its Cauchy prior that is centered on the point null.

These results reinforce the point that the "Bayes Factor" that is calculated from BIC statistics will overly favor the alternative when sample sizes are small. Bear in mind that the small sample context is the one where other model assumptions are of most consequence.

The article `https://www.rpubs.com/lindeloev/bayes_factors` describes and compares results from different models used to calculate Bayes factors in several relevant R packages.

2.9.3 Bayesian regression estimation using the* MCMCpack *package*

Subsection 2.9 discussed ideas of Bayesian estimation, drawing attention to the use of the Markov Chain Monte Carlo (MCMC) simulation technique to generates successive parameter estimates. The simulation process must be allowed to *burn in*, i.e., run for long enough that the posterior distribution reaches a steady state that is independent of the starting values of parameters. The function `MCMCpack::MCMCregress()`, with a similar syntax to `lm()`, can be used for the calculations. The default is to assume independent uniform priors for the regression coefficients, to allow the simulation to run for 10 000 iterations, and to take the first 1000 iterations as burn-in.

The example that now follows fits a regression model to the `roller` data of Subsection 1.5.2. It is designed for illustrative purposes, for a regression fit where most data analysts would consider this level of sophistication overdone. Code and accompanying output are:

```
suppressPackageStartupMessages(library(MCMCpack))
roller.mcmc <- MCMCregress(depression ~ weight, data=DAAG::roller)
summary(roller.mcmc)
```

```
Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

             Mean     SD Naive SE Time-series SE
(Intercept) -2.00  5.486  0.05486        0.05316
weight       2.65  0.812  0.00812        0.00765
sigma2      60.47 40.610  0.40610        0.52264

2. Quantiles for each variable:

              2.5%   25%   50%   75%  97.5%
```

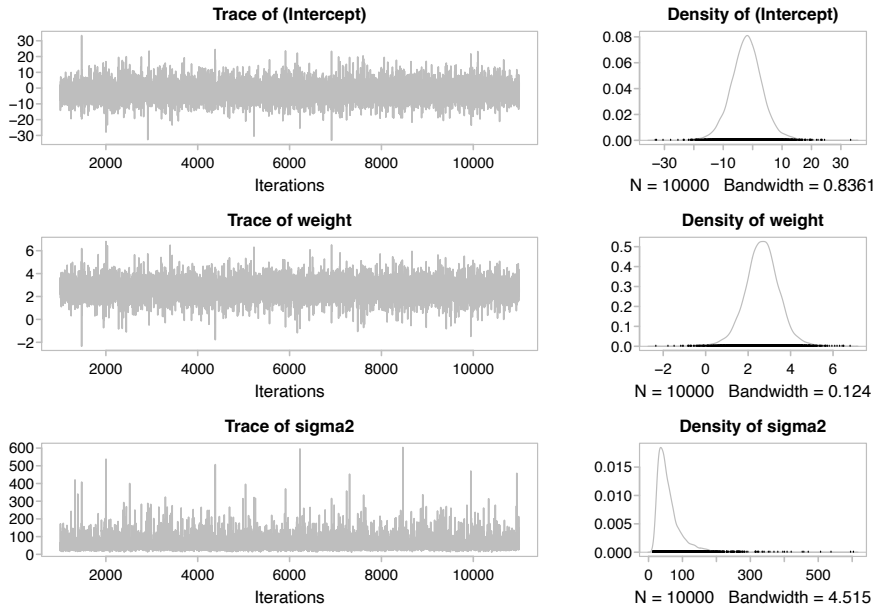


Figure 2.19 Diagnostic plots for the Bayesian analysis that used `MCMCregress()`.

```
(Intercept) -12.80  -5.38  -1.99   1.29    9.25
weight        1.01   2.17   2.66   3.16    4.26
sigma2       21.01  35.40  49.39  71.42  166.23
```

Because estimates from the previous iteration are the starting values for the current iteration, the sequence of estimates is Markovian and there is a lag 1 partial autocorrelation. The time series SE in the final column is designed to adjust for this partial autocorrelation. (Specifically, it is assumed that they follow an autoregressive process of order 1.) The standard error is inflated to take account of the partial autocorrelation between successive estimates. Notice that the coefficient estimates are very similar to those obtained in Subsection 1.5.2 using `lm()`, while the SEs (both sets) are slightly larger.

Figure 2.19 uses a plot method, for objects of class `mcmc` that can be used as a check on whether an adequate number of iterations were allowed for burn-in. The layout has been changed somewhat from the default. The code is:

```
mat <- matrix(c(1:6), byrow=TRUE, ncol=2)
# panels are 1, then 2, ... 6. Layout=dim(mat), i.e., 3 by 2
layout(mat, widths=rep(c(2,1),3), heights=rep(1,6))
# NB: widths & heights are relative
plot(roller.mcmc, auto.layout=FALSE, ask=FALSE, col="gray40")
# The method is plot.mcmc()
```

These plots are unremarkable. For this very simple model, burn-in occurs quickly, and none of the plots show any indication of a trend. The posterior distributions of the model coefficients all look plausibly normal.

The *coda* package, on which *MCMCpack* depends, has several other functions that give diagnostic information that may be helpful in interpreting the MCMC results. See `help(package="coda")`.

2.10 Recap

- The aim should be an insightful and coherent account of the data, placing it in the context of what is already known. Ensure that the statistical analysis assists this larger purpose. Ensure that the analysis and graphs reflect any important structure in the data.
- In group comparisons, present means, standard errors, and numbers for each group. Results from formal significance tests have secondary usefulness.
- The use of many significance tests readily leads to data summaries that lack coherence and insight. Look for coherent forms of analysis that are effective in summing up what can be learned from the data.
- Reserve multiple range tests for unstructured comparisons.
- Think about the science behind the data. What analysis (or analyses) will best reflect that science?

Statistical models have both deterministic (*signal*) and random error (*noise*) components. In simpler cases, as in the present chapter, the model takes the form:

$$\text{observation} = \text{signal} + \text{noise}.$$

The hope is that the fitted value will recapture most of the signal, and that the residual will contain mostly noise. Unfortunately, as the relative contribution of the noise increases,

- it becomes harder to distinguish between signal and noise,
- it becomes harder to decide between competing models.

Model assumptions, such as normality, independence, and constancy of the variance, should be checked to the extent possible. Plots of residuals are important diagnostic tools. The function `plot()`, with an `lm` model object as argument, gives a basic set of diagnostic plots, as in Figure 2.7.) A separate check is needed, where data have a time or other such sequence that may affect results, for sequential correlation. A course check can be performed by applying the function `acf()` to the vector of residuals. See Subsection 6.1.2.

Model coefficients give the values by which the values in the respective columns of the model matrix must be multiplied. These are then summed over all columns. Later chapters will use the model matrix formulation to fit models where fitted values are the sum of linear combinations of nonlinear terms.

Keep in mind that the usual product-moment correlation measures linear association. Wherever possible, use the richer and more insightful regression framework.

Alternatives to straight line regression with x and y as measured are:

- Transform x and/or y.
- Use polynomial regression.
- Fit a smoothing curve.

Regress y on x, or x on y?

The line for the regression of y on x is different from the line for the regression of x on y. The difference between the two lines is most marked when the correlation is small.

2.11 Further reading

Finding the right statistical model is an important part of statistical problem solving. Chatfield (2003) has helpful comments. Clarke (1968) has a useful discussion of the use of models in archaeology. See, also, the very different points of view of Breiman and Cox (as discussant) in Breiman (2001). Our stance is much closer to Cox than to Breiman.

Miller (1986) has extensive comment on consequences of failure of assumptions, and on how to handle such failures. Smith (2014) is a wide-ranging compendium of examples of errors in data interpretation. Discussion stays at an elementary, mostly non-mathematical, level. Johnson (1995) comments critically on the limitations of widely used nonparametric methods.

Bayesian methodology is used, in this text, primarily as a commentary on what can be achieved using frequentist approaches. Chapters 4, 6 and 7 of Bolker (2008) give a brief summary of Bayesian methodology, including Bayesian modeling. Spiegelhalter, Myles, et al. (2000) is a helpful 130 page overview of practical issues for the use of Bayesian methods. See also Doorn et al. (2021).

2.12 Exercises

1. In a study that examined the use of acupuncture to treat migraine headaches, consenting patients on a waiting list for treatment for migraine were randomly assigned in a 2:1:1 ratio to acupuncture treatment, a 'sham' acupuncture treatment in which needles were inserted at non-acupuncture points, and waiting-list patients whose only treatment was self-administered (Linde, Streng, et al., 2005). (The 'sham' acupuncture treatment was described to trial participants as an acupuncture treatment that did not follow the principles of Chinese medicine.) The two tables that follow summarize results:

   a. Numbers of patients who experienced a more than 50% reduction in headaches over a four-week period, relative to a pre-randomization baseline were:

	Acupuncture	Sham acupuncture	Waiting list
$\geq$ 50% reduction	74	43	11
$<$ 50% reduction	71	38	65

   b. Patients who received the acupuncture and sham acupuncture treatments were asked to guess their treatment. Results were:

	Acupuncture	Sham acupuncture
Chinese	82	30
Other	17	26
Don't know	30	16

Analyze the two tables. What, in each case, are the conclusions that should be drawn? Comment on implications for patient treatment and further research.

2. The table `UCBAdmissions` was discussed in Section 1.3.1. The following gives a table that adds the 2×2 tables of admission data over all departments:

```
## UCBAdmissions is in the datasets package
## For each combination of margins 1 and 2, calculate the sum
UCBtotal <- apply(UCBAdmissions, c(1,2), sum)
```

   a. Compare the information in the table `UCBtotal` with the result from applying the function `mantelhaen.test()` (see `?mantelhaen.test`) to the dataset `UCBAdmissions`. Comment on the difference.
   b. The Mantel-Haenzel test is valid only if the male-to–female odds ratio for admission is similar across departments. The following code calculates the relevant odds ratios:

```
apply(UCBAdmissions, 3, function(x) (x[1,1]*x[2,2])/(x[1,2]*x[2,1]))
```

   Is the odds ratio consistent across departments? Which department(s) stand(s) out as different? What is the nature of the difference?

3. The following fictitious data is designed to illustrate issues for combining data across tables.

Table A:

	Engineering			Sociology			Total	
	Male	Female		Male	Female		Male	Female
Admit	30	10	Admit	15	30	Admit	45	40
Deny	30	10	Deny	5	10	Deny	35	20

Table B:

	Engineering			Sociology			Total	
	Male	Female		Male	Female		Male	Female
Admit	30	20	Admit	10	20	Admit	40	40
Deny	30	10	Deny	5	25	Deny	35	35

To enter the data for Table A, type:

```
tabA <- array(c(30,30,10,10,15,5,30,10), dim=c(2,2,2))
```

and similarly for Table B. The third dimension in each table is faculty, as required for using faculty as a stratification variable for the Mantel–Haenzel test. From the help page for `mantelhaen.test()`, extract and enter the code for the function `woolf()`. Apply the function `woolf()`, followed by the function `mantelhaen.test()`, to the data of each of Tables A and B. Explain, in words, the meaning of each of the outputs. Then apply the Mantel–Haenzel test to each of these tables.

4. In a sequence of 1000 experiments, let $P = 0.8$ (power) be the probability that an effect of interest will be detected at $p \leq \alpha$, where $\alpha = 0.05$. How many of the

1000 experiments can be expected to show an apparent effect at the $\alpha = 0.05$ cutoff level? Use Equation 1.4 in Subsection 1.6.4 to estimate the PPV, i.e., what proportion is expected to be real? where the number of genuine cases is 200, 300, ..., 900, in each case out of 1000.

5. *For constructing bootstrap confidence intervals for the correlation coefficient, the Fisher z-transformation of the correlation gives, under bivariate normality assumptions for the (x, y) combinations from which the correlation was calculated, a statistic with an approximately normal sampling distribution. The following lines of R code obtain a bootstrap confidence interval for the z-transformed correlation between `chest` and `belly` in the `possum` data frame. The final step applies the inverse of the z-transformation to the confidence interval to return it to the original scale. Run the code and compare the resulting interval with the one computed without transformation. Is the z-transform necessary here?

```
z.transform <- function(r) .5*log((1+r)/(1-r))
z.inverse <- function(z) (exp(2*z)-1)/(exp(2*z)+1)
  possum.fun <- function(data, indices) {
    chest <- data$chest[indices]
    belly <- data$belly[indices]
    z.transform(cor(belly, chest))}
possum.boot <- boot(possum, possum.fun, R=999)
z.inverse(boot.ci(possum.boot, type="perc")$percent[4:5])
# The 4th and 5th elements of the percent list element
# hold the interval endpoints. See ?boot.ci
```

6. Use the function `rexp()` to simulate 100 random observations from an exponential distribution with rate 1. Use the bootstrap (with 99 999 replications) to estimate the standard error of the median. Repeat several times. Compare with the result that would be obtained using the normal approximation, i.e., $\sqrt{\pi/(2n)}$.
7. Low doses of the insecticide toxaphene may cause weight gain in rats (Chu et al., 1988). A sample of 20 rats are given toxaphene in their diet, while a control group of 8 rats are not given toxaphene. Assume further that weight gain among the treated rats is normally distributed with a mean of 60g and standard deviation 30g, while weight gain among the control rats is normally distributed with a mean of 10g and a standard deviation of 50g. Using simulation, compare confidence intervals for the difference in mean weight gain, using the pooled variance estimate and the Welch approximation. Which type of interval is correct more often?

   Repeat the simulation experiment under the assumption that the standard deviations are 40g for both samples. Is one of the methods now giving systematically larger confidence intervals? Which type of interval do you consider best?
8. *Experiment with the `DAAG::pair65` example and plot various views of the likelihood function, either as a surface using the `persp()` function or as one-dimensional profiles using the `curve()` function. Is there a single maximizer? Where does it occur?
9. *Suppose the mean reaction time to a particular stimulus has been estimated in several previous studies, and appears to be approximately normally distributed

with mean 0.35 seconds with standard deviation 0.1 seconds. On the basis of 10 new observations, the mean reaction time is estimated to be 0.45 seconds with an estimated standard deviation of 0.15 seconds. Based on the sample information, what is the likelihood estimator for the true mean reaction time? What is the Bayes' estimate of the mean reaction time?

10. Use the robust regression function `MASS::rlm()` to fit lines to the data in `elastic1` and `elastic2`. Compare the results with those from use of `lm()`. Compare regression coefficients, standard errors of coefficients, and plots of residuals against fitted values.
11. In the dataset `pressure` (*datasets*), examine the dependence of pressure on temperature. Try:

    ```
    with(pressure, MASS::boxcox(pressure ~ I(1/(temperature+273))))
    ```

    What transformation does this suggest?
    [Theory suggests that the logarithm of the vapor pressure should be approximately inversely proportional to the absolute temperature. Search on the internet for "Claudius-Clapeyron equation", or look in a suitable reference text.]
12. *Use the function `car::powerTransform()` to determine transformations for, for both variables, for use in connection with Exercise 11. (Be sure to work with absolute temperature.)
    a. Examine diagnostics for the regression fit that results following the suggested transformations. In particular, examine the plot of residuals against temperature. Comment on the plot. What are its implications for further investigation of these data?
    b. Use `summary()` with the output from `car::powerTransform()`. Are the results consistent with the Claudius-Clapeyron equation? [Note that Subsection 3.3.3 supplements the discussion of `powerTransform()` in Subsection 2.5.6.]
13. Fit the double binomial distribution to the `qra::malesINfirst12` data from Subsection 2.3.1 that gave numbers of male and female children in large families in Saxony in the nineteenth century.
    a. Compare the worm plots. Is there any consistent difference in the patterns?
    b. Use `AIC` or `GAIC` to compare the fits obtained using `gamlss::gamlss()`. Repeat with the argument `c=TRUE` that is available for the *gamlss* AIC method. Why does this make almost no difference? What is n in this context?
14. *The following function returns the coefficient of the estimated linear functional relationship between `x` and `y`:

    ```
    "funRel" <-
    function(x=leafshape$logpet, y=leafshape$loglen, scale=c(1,1)){
      ## Find principal components rotation; see Subsection 9.1.2
      ## Here (unlike 9.1.2) the interest is in the final component
      xy.prc <- prcomp(cbind(x,y), scale=scale)
      b <- xy.prc$rotation[,2]/scale
      c(bxy = -b[1]/b[2])      # slope of functional equation line
    }
    ## Try the following:
    leafshape <- DAAG::leafshape
    ```

```
funRel(scale=c(1,1))    # Take x and y errors as equally important
## Note that all lines pass through (mean(x), mean(y))
```

a. Write $b_{y.x}$ for the slope of the regression line of y on x and $b_{x.y}$ for the slope of the regression line of x on y. For each of the three settings `scale=c(1,10)`, `scale=c(1,1)`, `scale=c(10,1)`, of the argument `scale`, note where the values of the functional coefficient lie in the range between $b_{y.x}$ and $b_{x.y}^{-1}$.
b. Repeat this for each of the data frames `softbacks`, `elastic2`, and (with the variables `logpet` and `loglen`) `leafshape17`.
c. Explain the effect of changing the settings of the argument `scale`.

15. *A Markov chain is a data sequence which has a special kind of dependence. For example, a fair coin is tossed repetitively by a player who begins with \$2. If 'heads' appear, the player receives one dollar; otherwise, she pays one dollar. The game stops when the player has either \$0 or \$5. The amount of money that the player has before any coin flip can be recorded – this is a Markov chain. A possible sequence of plays is as follows:

Player's fortune:	2	1	2	3	4	3	2	3	2	3	2	1	0
Coin Toss result:	T	H	H	H	T	T	H	T	H	T	T	T	

Note that all we need to know in order to determine the player's fortune at any time is the fortune at the previous time as well as the coin flip result at the current time. The probability of an increase in the fortune is 0.5 and the probability of a decrease in the fortune is 0.5. The transition probabilities can be summarized in a transition matrix:

```
     0   1   2   3   4   5
0  1.0 0.0 0.0 0.0 0.0 0.0
1  0.5 0.0 0.5 0.0 0.0 0.0
2  0.0 0.5 0.0 0.5 0.0 0.0
3  0.0 0.0 0.5 0.0 0.5 0.0
4  0.0 0.0 0.0 0.5 0.0 0.5
5  0.0 0.0 0.0 0.0 0.0 1.0
```

The $(i+1, j+1)$ entry of this matrix is the probability of making a change from the value i to the value j. Here, the possible values of i and j are $0, 1, 2, \ldots, 5$. According to the matrix, there is a probability of 0 of making a transition from \$2 to \$4 in one play, since the (2,4) element is 0; the probability of moving from \$2 to \$1 in one transition is 0.5, since the (2,1) element is 0.5.

The following function can be used to simulate N values of a Markov chain sequence, with transition matrix P:

```
Markov <- function(N=15, initial.value=1, transition=P, stopval=NULL)
  {X <- numeric(N)
   X[1] <- initial.value + 1  # States 0:(n-1); subscripts 1:n
   n <- nrow(transition)
   for (i in 2:N){
    X[i] <- sample(1:n, size=1, prob=transition[X[i-1], ])
    if(length(stopval)>0)if(X[i] %in% (stopval+1)){X <- X[1:i]; break}}
  X - 1
}
## Set `stopval=c(0,5)` to stop when  the player's fortune is $0 or $5
```

Simulate 15 values of the coin flip game, starting with an initial value of $2. Repeat the simulation several times.

16. *A Markov chain for the weather in a particular season of the year has the transition matrix, from one day to the next:

```
       Sun Cloud Rain
Sun    0.6   0.2  0.2
Cloud  0.2   0.4  0.4
Rain   0.4   0.3  0.3
```

The (i, j) entry of this matrix is the probability of making a change from the value i to the value j, where `Sun` is 1, `Cloud` is 2, and `Rain` is 3. It can be shown, using linear algebra, that in the long run this Markov chain will visit the states according to the *stationary* distribution:

Sun	Cloud	Rain
0.429	0.286	0.286

A result called the *ergodic* theorem allows us to estimate this distribution by simulating the Markov chain for a long enough time.

a. Simulate 1000 values, and calculate the proportion of times the chain visits each of the states. Compare the proportions given by the simulation with above theoretical proportions.

b. Here is code that uses the function `zoo::rollmean()` to calculate rolling averages of the proportions over a number of simulations and plot the result:

```
plotmarkov <-
  function(n=1000, width=101, start=0, transition=Pb, npanels=5){
    xc2 <- Markov(n, initial.value=start, transition)
    mav0 <- zoo::rollmean(as.integer(xc2==0), k=width)
    mav1 <- zoo::rollmean(as.integer(xc2==1), k=width)
    npanel <- cut(1:length(mav0), breaks=seq(from=1, to=length(mav0),
                  length=npanels+1), include.lowest=TRUE)
    df <- data.frame(av0=mav0, av1=mav1, x=1:length(mav0), gp=npanel)
    print(xyplot(av0+av1 ~ x | gp, data=df, layout=c(1,npanels), type="l",
          par.strip.text =list(cex=0.65), auto.key=list(columns=2),
          scales=list(x=list(relation="free"))))
}
```

Try varying the number of simulations and the width of window. How wide a window is needed to get a good sense of the stationary distribution? This series settles down rather quickly to its stationary distribution (it "burns in" quite quickly). A reasonable width of window is however needed to give an accurate indication of the stationary distribution.

17. The Monty Hall problem (Franco-Watkins et al., 2003) arose from a 1970's era television game show in which a contestant is provided with an opportunity to win an expensive prize. The prize is hidden behind one of three doors, and the contestant is asked to choose one of the doors. Upon making the choice, the game show host chooses one of the other two doors to be opened, according to a rule that ensures that the door hiding the prize is not opened. If neither door hides the prize, then the door to be opened is chosen at random. The contestant

is then allowed the option of choosing from among the remaining two unopened doors; in other words, they may switch to a new door or remain with their first choice. The final door chosen is then opened, either revealing the prize or revealing nothing.

a. Prior to starting the game, what should be the contestant's assessment of the probability that the prize is behind door 1?
b. Suppose the contestant first chooses door 1. Calculate the probability that the host will choose to open door 3, under the hypotheses:

$$H_1: \text{ the prize is hidden behind door 1}$$

$$H_2: \text{ the prize is hidden behind door 2}$$

$$H_3: \text{ the prize is hidden behind door 3}$$

c. Suppose the host opens door 3 (revealing no prize). Which of the three hypotheses, H_1, H_2 and H_3 maximizes the likelihood? In other words, under which of them is the probability that the host chooses door 3 highest?
d. By calculating the ratio of the likelihood under H_2 to the likelihood under H_1, show that the Bayes factor that door 2 hides the prize is 2.0.
e. Use Bayes' Theorem to show that the probability of H_1, given the host's choice of door 3, is $1/3$, and the probability of H_2, under that same choice is $2/3$. Alternatively, use the Bayes Factor calculated above, together with the prior probabilities to arrive at these posterior probabilities of H_1 and H_2.