Contents

1

Getting Started

1.1 The only thing to remember is the placement of parentheses, and the need to use * for multiplication:

```
1 + 2*(3+4)
```

```
## [1] 15
```

```
4^3 + 3^(2+1)
```

```
## [1] 91
```

```
sqrt((4+3)*(2+1))
```

```
## [1] 4.582576
```

```
( (1+2)/(3+4) )^2
```

```
## [1] 0.1836735
```

1.2 These would be $(2+3)-4$, $2+(3*4)$, $(2/3)/4$, and $2^{(3^4)}$; the last working right to left.

1.3 Translating this to R requires attention to the use of parentheses and using an asterisk for multiplication:

```
(1 + 2*3^4) / (5/6 - 7)
```

```
## [1] -26.43243
```

1.4 We use the 1/2 power as an alternative to the `sqrt` function:

```
(0.25 - 0.2) / (0.2 * (1 - 0.2)/100)^(1/2)
```

```
## [1] 1.25
```

1.5 We don't use `c` below, as it is a very commonly used function in R:

```
a <- 2; b <- 3; d <- 4; e <- 5
a * b * d * e
```

```
## [1] 120
```

1.6 It is 1770.

1.7 It is `2510`. Instead of scanning, this can be automated:

```
require(UsingR)
```

```
## Loading required package: UsingR
## Loading required package: MASS
##
## Attaching package: 'UsingR'
##
## The following object is masked from 'package:ggplot2':
##
##    movies
##
## The following object is masked from 'package:survival':
##
##    cancer
```

```
max(exec.pay)
```

```
## [1] 2510
```

1.8 These values come from:

```
require(UsingR)
mean(exec.pay)

## [1] 59.88945

min(exec.pay)

## [1] 0

max(exec.pay)

## [1] 2510
```

1.9 This is done with:

```
require(UsingR)
mean(exec.pay)

## [1] 59.88945

mean(exec.pay, trim=0.10)

## [1] 29.96894
```

The big difference is due to the fact that the very large salaries that are trimmed have big influence on the average of the data set computed by `mean`.

1.10 The variable names are printed when the data set is displayed. They are `Tree`, `age`, and `circumference`.

1.11 The only trick is to reference the variable appropriately:

```
mean(Orange$age)
```

```
## [1] 922.1429
```

2

Univariate data

2.1 For example:

```
p <- c(2, 3, 5, 7, 11, 13, 17, 19)
```

2.2 The `diff` function returns the distance between fill-ups, so `mean(diff(gas))` is your average mileage per fill-up, and `mean(gas)` is the uninteresting average of the recorded mileage.

2.18 The comparison of strings is done lexicographically. That is, comparisons are done character by character until a tie is broken. The comparison of characters varies due to the locale. This may be decided by ASCII codes—which yields alphabetically ordering—but need not be. See `?locale` for more detail.

2.21 Enter in the data as follows:

```
x <- c(0.57, 0.89, 1.08, 1.12, 1.18, 1.07, 1.17, 1.38, 1.441, 1.72)
names(x) <- 1990:1999
```

Using `diff` gives

```
diff(x)
```

```
##   1991   1992   1993   1994   1995   1996   1997   1998   1999
##  0.320  0.190  0.040  0.060 -0.110  0.100  0.210  0.061  0.279
```

We can see that one year was negative:

```
which(diff(x) < 0)


## 1995
##    5
```

The jump between 1994 and 1995 was negative (there was a work stoppage that year). The percentage difference is found by dividing by `x[-10]` and multiplying by 100. (Recall that `x[-10]` is all but the tenth (10th) number of `x`). The first year's jump was the largest.

```
diff(x)/x[-10] * 100


##      1991      1992      1993      1994      1995      1996
## 56.140351 21.348315  3.703704  5.357143 -9.322034  9.345794
##      1997      1998      1999
## 17.948718  4.420290 19.361554
```

2.34 The graphics can be produced with these commands:

```
require(MASS)
hist(DDT)
boxplot(DDT)
```

The histogram shows the data to be roughly symmetric, with one outlying value, so the mean and median should be similar and the standard deviation about three-quarters the IQR. The median and IQR can be identified on the boxplot giving estimates of 3.2 for the mean and a standard deviation a little less than 0.5. We can check with this command:

```
c(mean=mean(DDT), sd=sd(DDT))


##      mean        sd
## 3.3280000 0.4371531
```

2.36 The histogram has a rather wide range (about 3 times from smallest to largest. Some year had over 93 feet of snow fall!

2.37 First assign names. Then you can access the entries using the respective state abbreviations.

```
names(state.area) <- state.abb
state.area[NJ]
```

```
##   NJ
## 7836
```

```
sum(state.area < state.area[NJ])/50 * 100
```

```
## [1] 8
```

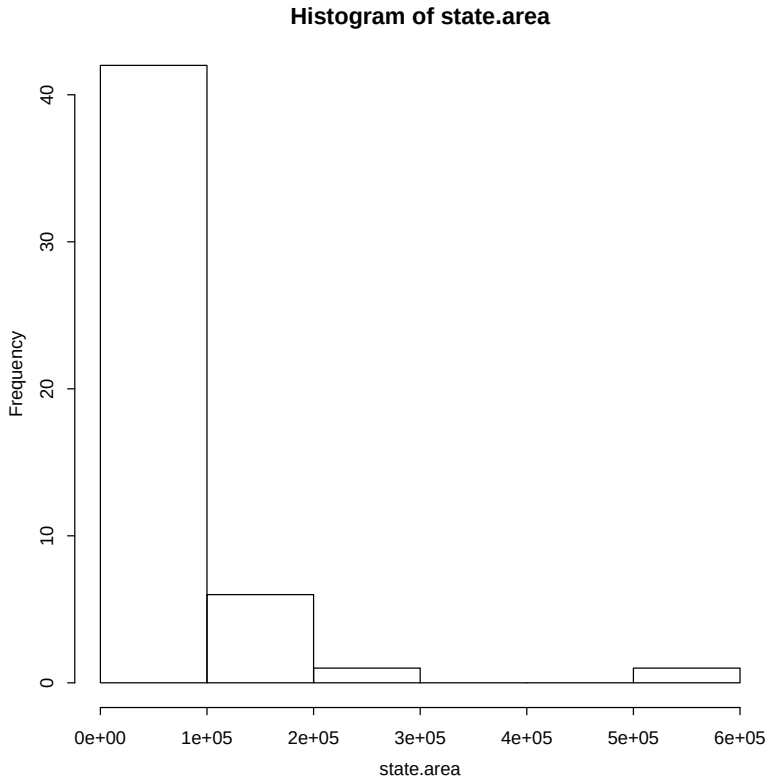
```
sum(state.area < state.area[NY])/50 * 100
```

```
## [1] 40
```

To see that Alaska is the big state, we could look at this histogram then query:

```
hist(state.area)                          # 50,000 cuts of last case
state.area[state.area > 5e5]
```

```
##     AK
## 589757
```



2.39 The definition of the median is incorrect. Can you think of a shape for a distribution when this is actually okay?

2.40 The median is lower for skewed-left distributions. It makes an area look more affordable. For exclusive listings, the mean is often used to make an area seem more expensive.

2.42 These values are found with:

```
sum(rivers < 500) / length(rivers)

## [1] 0.5815603

sum(rivers < mean(rivers)) / length(rivers)

## [1] 0.6666667
```

```
quantile(rivers, 0.75)


## 75%
## 680
```

2.43 First grab the data and check the units (minutes). The top 10% is actually the 0.10 quantile in this case, as shorter times are better.

```
times <- nym.2002$time                  # easier to use
range(times)                            # looks like minutes


## [1] 147.3333 566.7833


sum(times < 3*60)/length(times) * 100   # 2.6% beat 3 hours


## [1] 2.6


quantile(times,c(.10, .25))             # 3:28 to 3:53


##     10%     25%
## 208.695 233.775


quantile(times,c(.90))                  # 5:31


##    90%
## 331.75
```

It is doubtful that the data is symmetric. It is much easier to be relatively slow in a marathon, as it requires little talent and little training—just doggedness.

2.45 We see

```
stem(islands)                           # quite skewed


##
##   The decimal point is 3 digit(s) to the right of the |
##
```

```
##    0 | 00000000000000000000000000000111111222338
##    2 | 07
##    4 | 5
##    6 | 8
##    8 | 4
##   10 | 5
##   12 |
##   14 |
##   16 | 0
```

```
c(mean=mean(islands),
  median=median(islands),
  trimmed=mean(islands,trim=0.25))
```

```
##       mean     median    trimmed
## 1252.72917   41.00000   51.08333
```

The data set is quite skewed due to the seven continents. We wouldn't expect the mean and median to agree, but note that after trimming the mean and median are similar.

2.46 We can find the z-score for Barry Bonds using the name as follows:

```
(OBP[bondsba01]- mean(OBP)) / sd(OBP)
```

```
## bondsba01
##  5.990192
```

This is a decidedly "non-normal" data set, as we wouldn't expect z-scores larger than 3 in that case.

2.53 The histograms are all made in a similar manner to this:

```
hist(hall.fame$HR)
```

The home run distribution is skewed right, the batting average is fairly symmetric, and on-base percentage is also symmetric but may have longer tails than the batting average.

2.54 After a log transform the data looks more symmetric. If you find the median of the transformed data, you can take its exponential to get the median of the untransformed data. Not so with the mean.

2.60 This can be done as follows (using a different name from the built-in `mode` function):

```
Mode <- function(x) {
  tbl <- table(x)
  ind <- which(tbl == max(tbl))
  vals <- names(ind)
  as(vals, class(x)[1])                    # unnecessary!
}
```

Outside of the last line, this is a simple translation of the example give. The last line is not necessary. It simply generalizes the call to `as.numeric` in the example by coercing the output to the class of the input variable.

2.61 From this command we see the answer is 1001-2000 dollars:

```
bumps <- cut(bumpers, c(0, 1000, 2000, 3000, 4000))
table(bumps)
```

```
## bumps
##     (0,1e+03] (1e+03,2e+03] (2e+03,3e+03] (3e+03,4e+03]
##             2             8             7             6
```

2.62 The output of `summary` is a table:

```
summary(Cars93$Cylinders)
```

```
##      3      4      5      6      8 rotary
##      3     49      2     31      7      1
```

This seems a good choice—factors are used to categorize values and a table of counts is a useful summary.

2.63 Applying the idiom to `lorem` we have:

```
chars <- unlist(strsplit(lorem, split=""))
table(chars)
```

```
## chars
##  \n       ,   ;   .   a   A   b   c   C   d   D   e   E   f   F
##  10 589  48   1  73 251   3  38 156   5 102   8 370   3  22   2
##   g   h   i   I   j   l   L   m   M   n   N   o   p   P   q   Q
##  45  17 343   8   4 220   3 142   7 211  13 174  80   6  30   2
##   r   s   S   t   u   U   v   V   x
## 183 272   7 289 289   3  49   5   3
```

Scanning we see that `e` is the most common. To avoid scanning, the function `sort` can be called on the output of `table`:

```
sort(table(chars))
```

```
## chars
##   ;   F   Q   A   E   L   U   x   j   C   V   P   M   S   D   I
##   1   2   2   3   3   3   3   3   4   5   5   6   7   7   8   8
##  \n   N   h   f   q   b   g   ,   v   .   p   d   m   c   o   r
##  10  13  17  22  30  38  45  48  49  73  80 102 142 156 174 183
##   n   l   a   s   t   u   i   e
## 211 220 251 272 289 289 343 370 589
```

2.64 This is done with

```
require(MASS)
dotchart(table(Cars93$Cylinders))
```

```
## Warning in dotchart(table(Cars93$Cylinders)): 'x' is neither a
vector nor a matrix: using as.numeric(x)
```

The graphic shows that 4-cylinder cars were the most popular in 1993. Was this the case in 1974 (cf. `mtcars$cyl`)?

3

Bivariate data

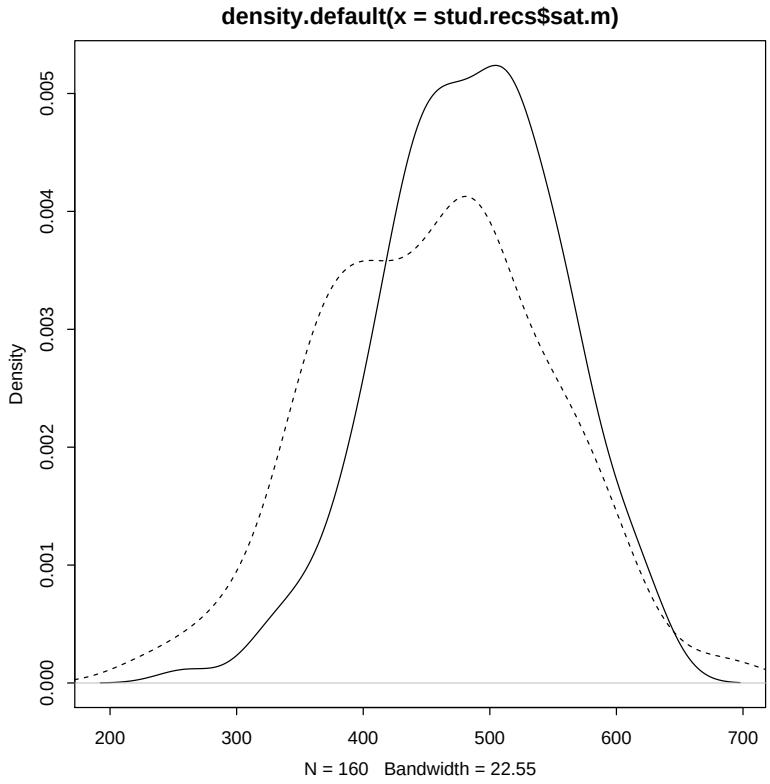
3.10 The boxplots may be produced with a command like:

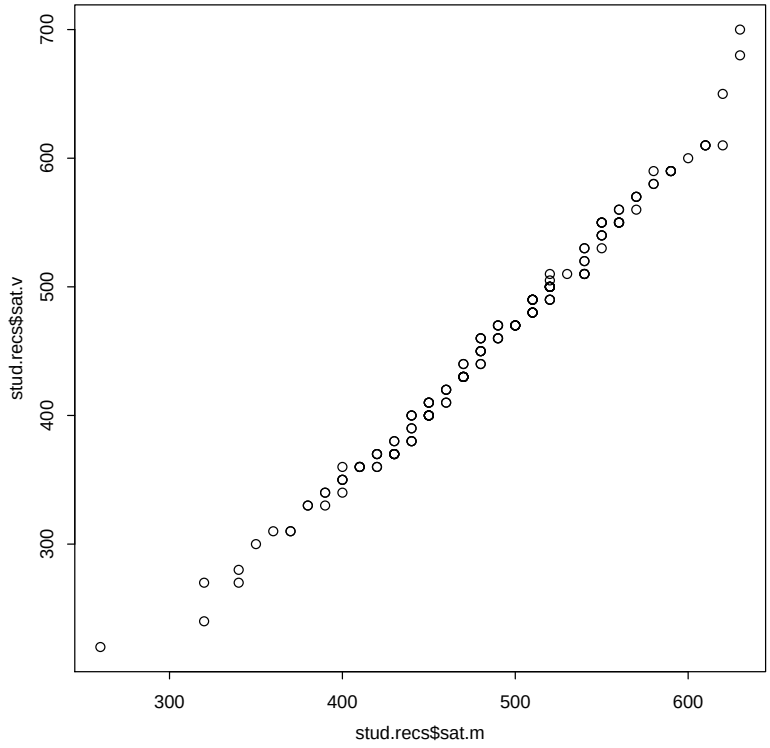
```
boxplot(twins$Foster, twins$Biological, names=c("Foster","Biological"))
```

The centers and spreads appear to be similar.

3.11 If we don't account for potential clipping, the graphs can be produced with

```
plot(density(stud.recs$sat.m))
lines(density(stud.recs$sat.v),lty=2)
qqplot(stud.recs$sat.m, stud.recs$sat.v)
```

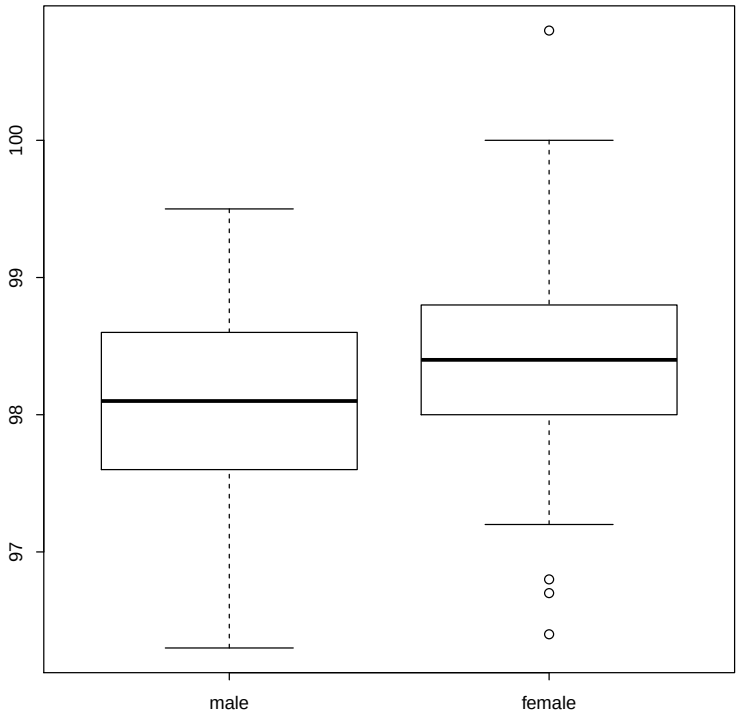

density.default(x = stud.recs$sat.m)
Density
0.000
0.001
0.002
0.003
0.004
0.005
200
300
400
500
600
700
N = 160 Bandwidth = 22.55


stud.recs$sat.v
300
400
500
600
700
300
400
500
600
stud.recs$sat.m

We see from the density plots that the center for `sat.v` appears to be larger. From the q-q plot it looks like the two distributions have similar shapes, as the points align fairly well.

3.12 We can split the data up using the condition `gender==1` as follows:

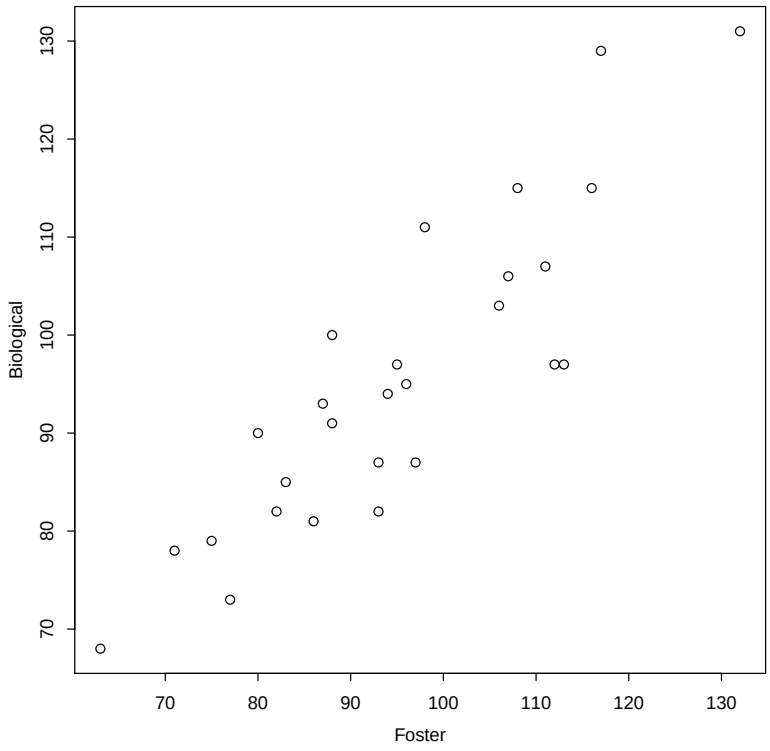
```
temperature <- normtemp$temperature; gender <- normtemp$gender
male <- temperature[gender == 1]
female <- temperature[gender == 2]
boxplot(list(male=male, female=female))
```



From the graphic it appears that the centers of the distributions are different.

3.16 The plot can be made with the commands

```
plot(Biological ~ Foster, data=twins)
```



Based on this, a guess of 0.8 for the Pearson coefficient and a similar value for the Spearman coefficient seem reasonable. A check can be done with

```
cor(twins$Foster, twins$Biological)

## [1] 0.8819877

cor(twins$Foster, twins$Biological, method="spearman")

## [1] 0.8858324
```

3.20 The correlation is found with

```
cor(batting$HR, batting$SO)


## [1] 0.7084697
```

Although the two variables are quite correlated, there is a likely a lurking variable—the number of at bats—that would explain the correlation.

3.22 We fit the model and then make the prediction as follows:

```
res <- lm(abdomen ~ wrist, data = fat)
predict(res, data.frame(wrist=17))


##        1
## 83.75187
```

3.23 The `wtloss` data can be plotted as follows:

```
cor(wtloss$Weight, wtloss$Days)             # close to -1


## [1] -0.9853149


plot(Weight ~ Days, data=wtloss)
res <- lm(Weight ~ Days, data=wtloss)
abline(res)
plot(residuals(res) ~ Days, data=wtloss)    # A trend
```

We see from the scatter plot that a linear model may not explain the data well, despite the correlation coefficient that is close to -1. The residual plot amplifies this observation. This is to be expected as it is initially easy to lose a large amount of weight, but this gets progressively harder as there is less excess weight to shed.

3.26 The calculations are performed as follows:

```
lm(Female ~ Male, data = too.young)


##
## Call:
```

```
## lm(formula = Female ~ Male, data = too.young)
##
## Coefficients:
## (Intercept)         Male
##      5.4720       0.5754
```

```
plot(Female ~ Male, data = too.young)
abline(lm(Female ~ Male, data = too.young))
abline(7,1/2,lwd=2)
```

The result from the data set is a little more conservative than the rule of thumb, except for the teenage years.

3.29 From the scatter plot we see that four temperatures were used. It appears that only one data point is associated with a temperature of 150°C, but in fact there were ten:

```
table(motors$temp)
```

```
##
## 150 170 190 220
##  10  10  10  10
```

It is hard to tell whether the values for this temperature fit a linear model. Assuming they do, we can fit the model with

```
res <- lm(time ~ temp,  data=motors)
res
```

```
##
## Call:
## lm(formula = time ~ temp, data = motors)
##
## Coefficients:
## (Intercept)         temp
##     22999.1       -106.8
```

Then predictions can be made using `predict`:

```
predict(res, newdata=data.frame(temp=210))
```

```
##        1
## 580.5888
```

3.31 We can tabulate the coins then multiply by the amounts to find the total amount of money in the change bin.

```
table(coins$value)
```

```
##
## 0.01 0.05  0.1 0.25
##  203   59   42   67
```

```
table(coins$value) * c(0.01, 0.05, 0.1, 0.25)
```

```
##
##  0.01  0.05   0.1  0.25
##  2.03  2.95  4.20 16.75
```

```
sum(table(coins$value) * c(0.01, 0.05, 0.1, 0.25))
```

```
## [1] 25.93
```

The barplot of all years can be produced with the command

```
barplot(table(coins$year))
```

It shows a generally increasing trend, as it is more likely to contain more recent coins.

To get the coins by decade, a first attempt would be to try

```
table(cut(coins$year, breaks = seq(1920,2010,by=10)))
```

```
##
## (1.92e+03,1.93e+03] (1.93e+03,1.94e+03] (1.94e+03,1.95e+03]
##                   3                   2                   0
```

```
## (1.95e+03,1.96e+03] (1.96e+03,1.97e+03] (1.97e+03,1.98e+03]
##                   3                  16                  43
## (1.98e+03,1.99e+03]    (1.99e+03,2e+03]    (2e+03,2.01e+03]
##                  99                 147                  58
```

But this has ranges like 1921-1930. Using the argument `right=FALSE` will cause the intervals to take the form $[a,b)$, for example, 1920-1929.

Finally, the two way table can be made just by passing the data frame to `table`, as in `table(coins)`. Alternatively, one could reference the variables individually, as with `table(coins$year, coins$value)`. Either way, the marginal distributions are certainly not uniform, but there is very little change in the proportions from year to year. A check of the correlation yields:

```
cor(coins$year, coins$value)
```

```
## [1] 0.0595
```

3.32 The table is made with

```
require(MASS)
table(UScereal$shelf, UScereal$mfr)
```

```
##
##      G  K  N  P  Q  R
##   1  6  4  2  2  0  4
##   2  7  7  0  1  3  0
##   3  9 10  1  6  2  1
```

It appears that `Q` (Quaker Oats) is underrepresented on the shelf 1, and `R` (Ralston Purina) is overrepresented there. Shelf 1 is less likely to be seen by casual consumers, so it has less chance of encouraging impulse buying.

4

Multivariate data

4.8 The variables `Alcohol.consumption`, `Working.hours`, and `Military.spending`.

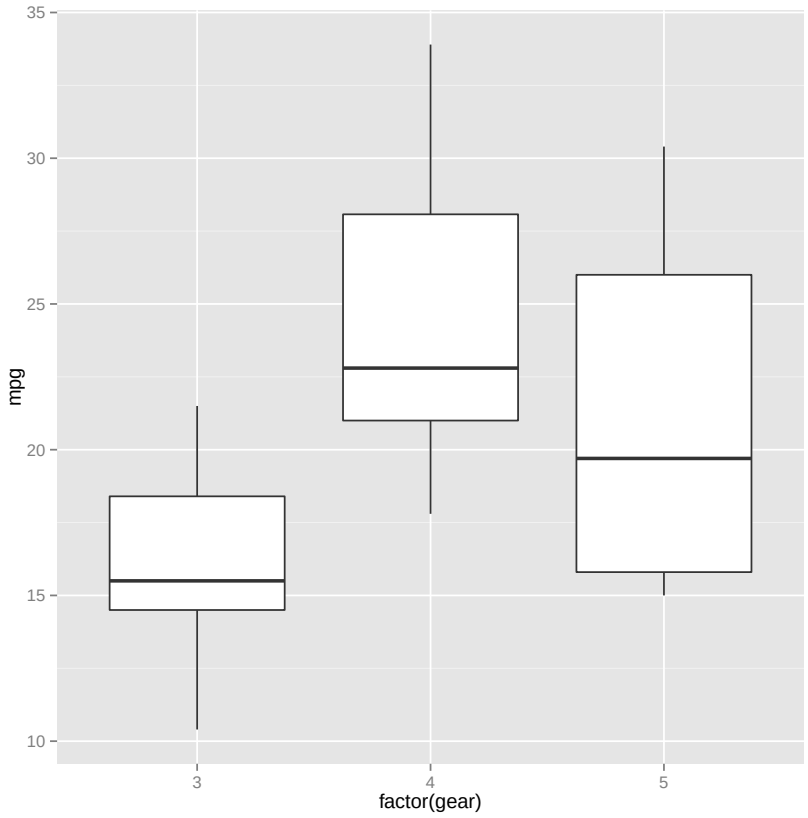
4.19 Subtraction is done left to right, as in $(2 - 3) - 4$, and powers are done right to left: $2^{(3^4}$.

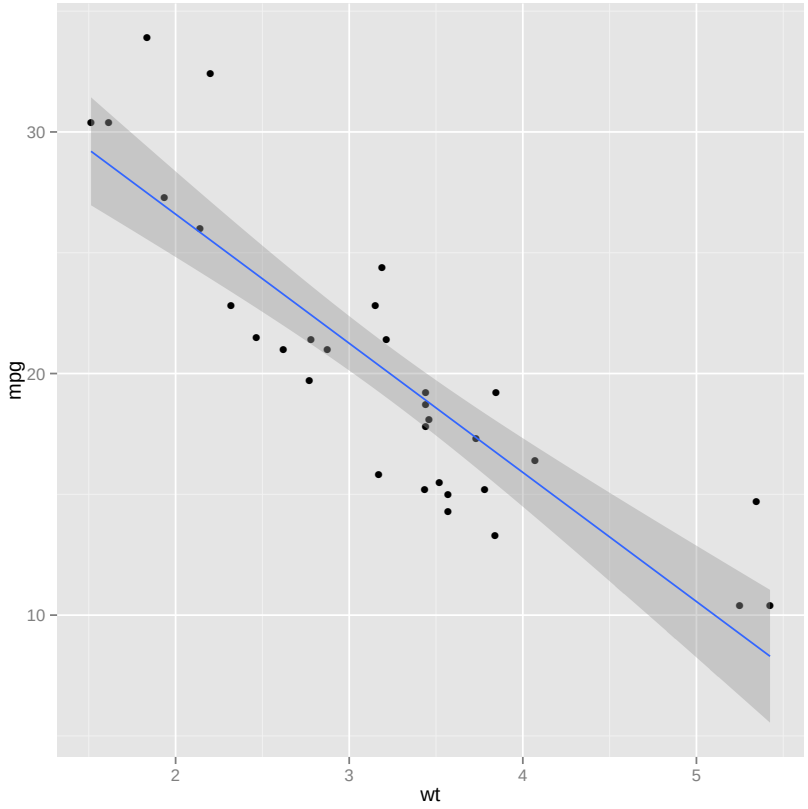
5

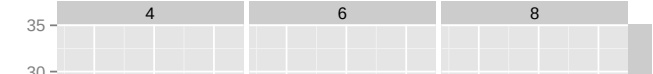
Multivariate graphics

5.7 These are done with:

```
p <- ggplot(mtcars)
p + aes(x=factor(gear), y=mpg) + geom_boxplot() # boxplots
p + aes(x=wt, y=mpg) + geom_point() + geom_smooth(method="lm") # scatter plot
p + aes(x=hp, y=mpg, col=factor(am), pch=factor(am)) + # add color and shape for transmissi
  geom_point() +
  facet_grid(gear ~ cyl)
```


35
30
25
20
15
10
mpg
3
4
5
factor(gear)


30
20
10
mpg
2
3
4
5
wt


4
6
8
35
30

6

Populations

6.3 The frequencies in `Balls` may be used for the probabilities, as shown:

```
with(nba.draft, sample(Team, 1, prob=Balls))


## [1] Memphis
## 13 Levels: Atlanta Chicago Cleveland Denver ... Washington
```

6.9 If we think of each try as independent and identically distributed, then the number of strikes in 12 tries is binomial. Thus, we have

```
dbinom(12, size=12, prob=.3)


## [1] 5.31441e-07
```

6.10 We can answer this using a sum with the following:

```
sum(dbinom(k, size=1e6, prob=1/2))


## [1] 0
```

6.12 This is known as de Mere's problem. Although the expected number of successes in each case is identical, as $24/36 = 4/6$, the probabilities differ. Namely,

```
1 - pbinom(0,4,1/6)           # one or more 6 in 4 rolls
```

```
## [1] 0.5177469
```

```
1 - pbinom(0,24,1/36)         # one or more double sixes
```

```
## [1] 0.4914039
```

The rolling of four dice is better than even, and the rolling of 24 is worse.

6.13 If we roll a pair of dice, the odds of getting a pair of sizes is $p = 1/36$. This can be answered as a binomial question: Let X record the number of double sixes in n rolls, what is the smallest n with $\mathsf{P}(X \geq 1) \geq 1/2$?

As we have `pbinom` answers $\mathsf{P}(X <= k)$, we translate the problem to: what is the smallest n for which $\mathsf{P}(X \leq 0) \leq 1/2$?

```
n <- 18:30                              # some range, may be wrong
names(n) <- n                           # helps, not necessary
pbinom(0, size=n, p=1/36)               # use vectorized n
```

```
##        18        19        20        21        22        23
## 0.6022541 0.5855248 0.5692603 0.5534475 0.5380739 0.5231274
##        24        25        26        27        28        29
## 0.5085961 0.4944685 0.4807332 0.4673795 0.4543968 0.4417746
##        30
## 0.4295031
```

Scanning the answers, we see that $n = 25$, not 18. (This can also be solved directly from $(35/36)^n \leq 1/2$, why?)

6.17 We assume that the scores are normally distributed. A student picked at random has a score that is assumed to be normally distributed with mean 20.6 and standard deviation 5.5. We compute $\mathsf{P}(X > 22)$ with

```
1 - pnorm(22, mean=20.6, sd=5.5)
```

```
## [1] 0.3995371
```

Thus, about 40% of the students passed the exam.

How many more would be expected to fail if the mark were moved? We multiply the probability times the sample size to get

```
1000000 * diff(pnorm(c(22,23), mean=20.6, sd=5.5))
```

```
## [1] 68250.64
```

6.20 The area under the standard normal density between -6 and 6 is characterized by

```
pnorm(6) - pnorm(-6)
```

```
## [1] 1
```

```
1 - (pnorm(6) - pnorm(-6))
```

```
## [1] 1.973175e-09
```

This is about 2 in a billion.

6.22 First we scale the data:

```
x <- father.son$fheight
x <- (x - mean(x))/sd(x)                # or scale(x)[,1]
sum(abs(x) < 1)/length(x)
```

```
## [1] 0.6753247
```

```
sum(abs(x) < 2)/length(x)
```

```
## [1] 0.9619666
```

```
sum(abs(x) < 3)/length(x)
```

```
## [1] 0.9990724
```

6.28 Using the de Moivre-Laplace theorem for the normal approximation gives

```
n <- 100; p <- 1/2; b <- 42
pbinom(b,n,p)

## [1] 0.06660531

pnorm(b+1/2, n*p, sqrt(n*p*(1-p)))

## [1] 0.0668072
```

6.29 The binomial model assumes *i.i.d.* at bats. The answer can be found with the following:

```
n <- 600; p <- .300
phat <- .350; a <-  n*phat
1 - pnorm(a - 1/2, n*p, sqrt(n*p*(1-p)))

## [1] 0.004293556
```

6.31 Let $X = X_1 + X_2 + \cdots + X_{15}$ be the total weight of the 15 occupants. The $\mathsf{P}(X > 3500)$ is equivalent to $\mathsf{P}(\bar{X} > 3500/15)$, which by the central limit theorem is

```
1 - pnorm(3500/15, mean=180, sd=25/sqrt(15))

## [1] 1.110223e-16
```

7

Statistical inference

8

Confidence intervals

8.5 The `prop.test` function gives

```
n <- 100; phat <- 0.45
prop.test(n*phat, n, conf.level = 0.8)


##
##  1-sample proportions test with continuity correction
##
## data:  n * phat out of n, null probability 0.5
## X-squared = 0.81, df = 1, p-value = 0.3681
## alternative hypothesis: true p is not equal to 0.5
## 80 percent confidence interval:
##  0.3827104 0.5190311
## sample estimates:
##    p
## 0.45


prop.test(n*phat, n, conf.level = 0.9)


##
##  1-sample proportions test with continuity correction
##
## data:  n * phat out of n, null probability 0.5
## X-squared = 0.81, df = 1, p-value = 0.3681
## alternative hypothesis: true p is not equal to 0.5
## 90 percent confidence interval:
##  0.3657761 0.5370170
## sample estimates:
##    p
## 0.45
```

8.8 We have a 2 percent margin of error, or

$$.02 = z^* \mathsf{SE}(\widehat{p}).$$

But $z^* = 1.96$ and $\mathsf{SE}(\widehat{p}) = \sqrt{.54 * (1 - .54)}/\sqrt{n}$. Solving gives

```
phat <- .54
zstar <- 1.96
(zstar * sqrt(phat*(1-phat)) / 0.02)^2


## [1] 2385.634
```

8.11 We have $z^* \mathsf{SE}(\widehat{p}) \leq 0.01$. We know z^* as α is given, so we have

$$\sqrt{\frac{\widehat{p}(1-\widehat{p})}{n}} \leq \frac{0.01}{z^*}.$$

Solving for n we get

$$\left(\frac{z^*}{0.01}\right)^2 \widehat{p}(1-\widehat{p}) \leq n.$$

No matter what $\widehat{p}$ is, the expression $\widehat{p}(1-\widehat{p})$ is largest when $\widehat{p} = 1/2$, so we have n is large enough if

$$\left(\frac{z^*}{0.01}\right)^2 \frac{1}{2}\left(1 - \frac{1}{2}\right) \leq n.$$

For our example, we have

```
## for 95% confidence
alpha <- 0.05; zstar <- qnorm(1 - alpha/2)
(zstar/0.01)^2/4


## [1] 9603.647


## for 90% confidence
alpha <- 0.1; zstar <- qnorm(1 - alpha/2)
(zstar/0.01)^2/4


## [1] 6763.859
```

```
## for 80% confidence
alpha <- 0.2; zstar <- qnorm(1 - alpha/2)
(zstar/0.01)^2/4


## [1] 4105.936
```

8.13 To compute, we have

```
xbar <- 9.5; s <- 1; n <- 15
alpha <- 0.1
tstar <- qt(1 - alpha/2, df=n-1)
SE <- s/sqrt(n)
c(xbar - tstar*SE, xbar + tstar*SE)


## [1] 9.045232 9.954768
```

We see that ten days is not in this interval.

8.15 These can be found with

```
t.test(homedata$y1970, conf.level = 0.9)


##
##  One Sample t-test
##
## data:  homedata$y1970
## t = 262.87, df = 6840, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 90 percent confidence interval:
##  70377.72 71264.14
## sample estimates:
## mean of x
##  70820.93


t.test(homedata$y2000, conf.level = 0.9)


##
##  One Sample t-test
##
## data:  homedata$y2000
```

```
## t = 169.79, df = 6840, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 90 percent confidence interval:
##  265769.5 270970.0
## sample estimates:
## mean of x
##  268369.8
```

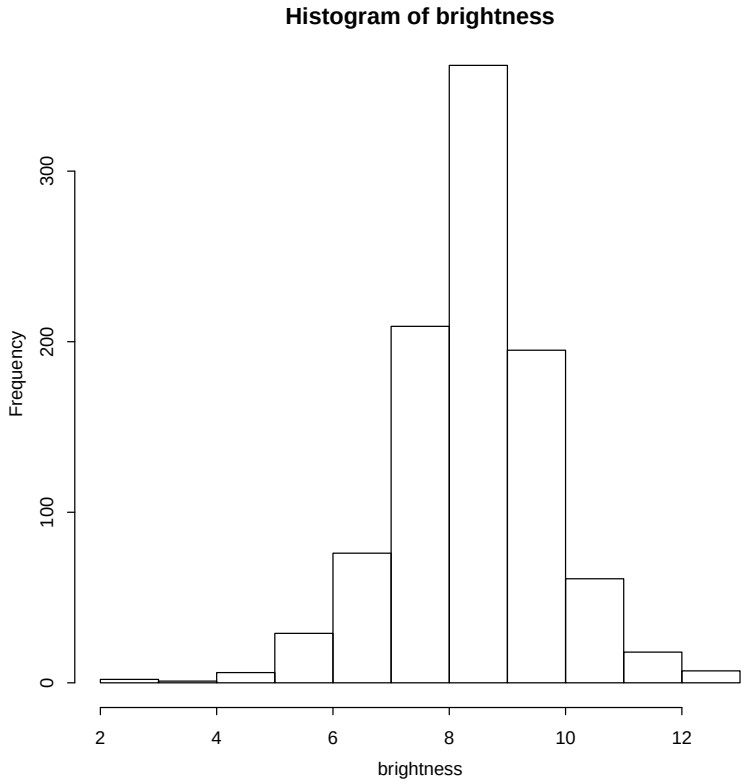
To assess appropriateness, we should ask if either the population appears normal, or n seems large:

```
qqnorm(homedata$y2000)                  # no, skewed
length(homedata$y2000)                  # yes, n=6841
```

8.17 To properly make such a statistical inference, we assume that we can treat the data as a random sample from the population of visible (with the aid of a telescope) stars and then use the sample to make an estimate for the mean brightness of these stars.

First, we make a histogram, to check that the sample appears to come from a normally distributed population. Once this is verified, then `t.test` can be used to find the desired confidence interval.

```
hist(brightness)              # bell-shaped, looks good
t.test(brightness, conf.level = 0.90)$conf.int


## [1] 8.349184 8.486303
## attr(,"conf.level")
## [1] 0.9
```



For reference: in most suburbs the human eye can see stars with a brightness of 5.5 or less of which about 2,800 exist.

8.18 No, it does not. The mean appears to be 98.2°F. (See `http://hypertextbook.com/facts/LenaWong.shtml` for some discussion on this.)

```
t.test(normtemp$temperature,conf.level=.90)


##
##  One Sample t-test
##
## data:  normtemp$temperature
## t = 1527.9, df = 129, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 90 percent confidence interval:
##  98.14269 98.35577
## sample estimates:
## mean of x
```

```
##  98.24923
```

8.22 The following commands will do the simulation:

```
set.seed(10)
```

```
M <- 200; n <- 10
alpha <- 1 - 0.90
res <- replicate(M, {
  x <- rnorm(n, mean=0, sd=2)
  c(z=qnorm(1-alpha/2) * 2/sqrt(n),
    t=qt(1 - alpha/2, df=n-1) * sd(x)/sqrt(n))
})                                          # a matrix
sum(res["z",] < res["t", ]) / M
```

```
## [1] 0.64
```

In this simulation it is 64% of the time. The z one is usually smaller, but not as often as might have been guessed.

8.24 The confidence interval can be found with:

```
place <- nym.2002$place
n <- length(place)
alpha <- 1 - 0.90
(1-alpha)^(1/n)
```

```
## [1] 0.9998946
```

```
max(place)/(1-alpha)^(1/n)
```

```
## [1] 23664.49
```

```
max(place)
```

```
## [1] 23662
```

So we are 90% certain that θ is between 23,662 and 23,664. (The real answer is 23,664.)

8.27 This data is paired off, as it is reasonable to assume that there is a correlation between the IQ scores of twins. We assume that differences in the scores can be viewed as a random sample from a normally distributed population. Then a confidence interval is found as follows:

```
foster <- c(80, 88,75, 113, 95, 82, 97, 94, 132, 108)
biological <- c(90, 91, 79, 97, 97, 82, 87, 94, 131, 115)
plot(foster, biological)
boxplot(foster - biological)
t.test(foster, biological, conf.level=0.90, paired=TRUE)


##
##  Paired t-test
##
## data:  foster and biological
## t = 0.041019, df = 9, p-value = 0.9682
## alternative hypothesis: true difference in means is not equal to 0
## 90 percent confidence interval:
##  -4.368937  4.568937
## sample estimates:
## mean of the differences
##                     0.1
```

The exploratory plots show the correlation between the two sets of scores and the distribution of the differences, indicating that a paired *t*-test is appropriate.

8.29 The data set implies that the population distribution is skewed and not normally distributed or symmetric. Thus `t.test` and `wilcox.test` are not appropriate. However, after a log transform, it appears that `wilcox.test` will be.

```
commutes <- rep(c(21,22,23,24,25,26,29,31,33),
  c(3,2,2,6,2,1,1,2,1))
ans <- wilcox.test(log(commutes), conf.int=TRUE, conf.level=0.9)


## Warning in wilcox.test.default(log(commutes), conf.int = TRUE,
conf.level = 0.9): cannot compute exact p-value with ties
```

```
## Warning in wilcox.test.default(log(commutes), conf.int = TRUE,
conf.level = 0.9): cannot compute exact confidence interval with
ties

confint(ans)

## (3.13, 3.26) with 90 percent confidence

confint(ans, transform=exp)             # also exp(ans$conf.int)

## (22.98, 26.11) with 90 percent confidence
```

Or, that $[23,26]$ is the confidence interval. The warning message is about the ties in the data. Notice that the sampling statistic is based on an assumption of a continuous distribution.[1]

8.30 We might envision the collection of songs on an album as coming from a population of all possible songs the band could have written at the given time. A difference of means would indicate a tendency to write longer songs at one of the given times.

The only trick to finding the confidence interval is referencing the times from *The Joshua Tree*. This needs to be quoted.

```
wilcox.test(u2$October,u2$"The Joshua Tree", conf.int=TRUE)

## Warning in wilcox.test.default(u2$October, u2$"The Joshua Tree",
conf.int = TRUE): cannot compute exact p-value with ties
## Warning in wilcox.test.default(u2$October, u2$"The Joshua Tree",
conf.int = TRUE): cannot compute exact confidence intervals with
ties

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  u2$October and u2$"The Joshua Tree"
## W = 26.5, p-value = 0.02778
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -94.000025  -9.999975
```

[1] The `coin` package has several functions that can work with tied data.

```
## sample estimates:
## difference in location
##              -48.99996
```

The calculated interval is $[-94, -10]$.

8.31 The AGE variable is symmetric, so no transformation is needed.

```
hist(cfb$AGE)                 # symmetric
wilcox.test(cfb$AGE, conf.int=TRUE)


##
##  Wilcoxon signed rank test with continuity correction
##
## data:  cfb$AGE
## V = 500500, p-value < 2.2e-16
## alternative hypothesis: true location is not equal to 0
## 95 percent confidence interval:
##  47.99992 50.00005
## sample estimates:
## (pseudo)median
##       48.99996
```

The INCOME variable is skewed. We do a log-transform first.

```
hist(log(cfb$INCOME +1))
ans <- wilcox.test((log(cfb$INCOME +1)), conf.int=TRUE)
exp(ans$conf.int) - 1


## [1] 35960.81 40470.11
## attr(,"conf.level")
## [1] 0.95
```

9

Significance tests

9.1 The FDA must assume that the drug is safe in the null hypothesis. The alternative would be that it is not safe.

9.2 We tabulate to get the counts.

```
cnts = table(samhda$marijuana)
cnts
```

```
##
##   1   2   9
## 134 463   3
```

We ignore the cases coded with 9, as they presumably are for individuals who chose not to answer. We see that there are 134 who answered "yes" and 463 who answered "no." The hypothesis test is between

$$H_0 : p = 0.5, \quad H_A : p > 0.5.$$

The p-value is given by

```
x <- cnts[1]
n <- cnts[1] + cnts[2]
prop.test(x, n, p=.5, alternative="greater")
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  x out of n, null probability 0.5
## X-squared = 180.21, df = 1, p-value = 1
## alternative hypothesis: true p is greater than 0.5
## 95 percent confidence interval:
##  0.1968507 1.0000000
```

```
## sample estimates:
##         p
## 0.2244556
```

This finds a p-value of 1. Decidely not significant. The data do not support the alternative, rather they suggest a decrease in the percentage of those who have tried marijuana.

9.5 The normal approximation is said to be valid provided np and $n(1-p)$ are 5 or more. In this case $p = .999$ under H_0 and $n = 5{,}760$, leaving $n(1-p)$ just larger than 5. Assuming the approximation produces accurate p-values, we have the one-sided test, yielding

```
out <- prop.test(5731, 5760, p=0.999, alternative="less")
out$p.value


## [1] 1.274574e-21
```

This is a very small p-value, indicating that the data is inconsistent with the null hypothesis.

9.6 The key to solving this is to find the value of z so that $P(\widehat{p} > z) = 0.05$. This is answered using the quantiles of the normal distribution, as $\widehat{p}$ is approximately normally distributed with mean $p = 0.1500$ and standard deviation $\sqrt{p(1-p)/n}$.

```
p <- 0.1500; n <- 150000
out <- qnorm(.95, mean=p,sd=sqrt(p*(1-p)/n))
```

Multiplying by n produces the cutoff:

```
n * out


## [1] 22727.47
```

9.8 The test must be done by hand, as the data is summarized.

```
xbar = 58260; n = 25; sd = 3250
mu = 55000; SE = sd/sqrt(n)
```

```
T = (xbar - mu)/SE
T
```

```
## [1] 5.015385
```

```
pt(T, df=n-1, lower.tail=FALSE)
```

```
## [1] 1.998964e-05
```

The significance test has a small p-value.

9.11 First check normality, then use `t.test` as follows:

```
x <- babies$dht
x <- x[x<99]
t.test(x, mu=68, alternative="greater")
```

```
##
##  One Sample t-test
##
## data:  x
## t = 20.796, df = 743, p-value < 2.2e-16
## alternative hypothesis: true mean is greater than 68
## 95 percent confidence interval:
##  70.02973      Inf
## sample estimates:
## mean of x
##   70.2043
```

9.15 We compute the p-value directly for a one-sided test with a null hypothesis of \$200:

```
xbar <- 650
n <- 8
s <- 100
mu <- 200
obs <- (xbar - mu) / (s/sqrt(n))
pt(obs, df=n-1, lower.tail=FALSE)
```

```
## [1] 2.139251e-06
```

The small p-value indicates the difference is statistically significant.

9.17 The sign test of the hypotheses

$$H_0 : \text{median} = 220{,}000, \quad H_A : \text{median} > 220{,}000$$

is done with test statistic T, the number of data points more than 22. Larger values of T support the alternative. The p-value is then calculated as

```
T <- sum(exec.pay > 22)
n <- length(exec.pay)
T


## [1] 113


pbinom(T - 1, n, .5, lower.tail=FALSE)


## [1] 0.03252251
```

9.18 The `log` function can be used directly, as in

```
wilcox.test(log(exec.pay), mu = log(22), alternative="greater")


##
##  Wilcoxon signed rank test with continuity correction
##
## data:  log(exec.pay)
## V = 10966, p-value = 0.004144
## alternative hypothesis: true location is greater than 3.091042
```

We see a similarly small p-value as in the previous question, indicating that the null hypothesis is not supported.

If you tried to do a histogram to check for symmetry, you may have gotten an error message. This is because of the data values for which the compensation is 0. (The logarithm of 0 is treated as $-\infty$.) To avoid this error, you can exclude this case first as follows:

```
ep <- exec.pay[exec.pay > 0]
hist(log(ep))
```

9.21 Assuming the phones are a random sample from the population of all phones, this can be done with `prop.test`, as follows:

```
x <- c("A"=14, "B"=15)
n <- c("A"=150, "B"=125)
prop.test(x, n, alternative="less")


##
##  2-sample test for equality of proportions with
##  continuity correction
##
## data:  x out of n
## X-squared = 0.27016, df = 1, p-value = 0.3016
## alternative hypothesis: less
## 95 percent confidence interval:
##  -1.00000000  0.04240782
## sample estimates:
##     prop 1     prop 2
## 0.09333333 0.12000000
```

The p value is not small, the answer is "no."

9.28 We have $n_1 = n_2 = 10{,}000$ and $\widehat{p}_1 = 0.216$, $\widehat{p}_2 = .193$. A two-sided significance test is performed with `prop.test` as follows:

```
p <- c(.216,.193)
n <- c(10000,10000)
x <- n * p
out <- prop.test(x,n)
out$p.value < 0.01                      # TRUE


## [1] TRUE
```

The difference is statistically significant. However, if the surveys were only of size 1,000 the difference would not have been statistically significant. As always, differences must be assessed in terms of variability which depends on sample size.

9.29 The assumptions are such that the t-statistic can be used to compute the small p-value. We assume that the population variances are equal.

```
xbar1 <- 79; xbar2 <- 110
n1 <- n2 <- 250
s1 <- 25; s2 <- 20
sp <- sqrt(( (n1-1)*s1^2 + (n2-1)*s2^2 )/(n1+n2-2))
SE <- sp * sqrt(1/n1 + 1/n2)
T <- (xbar1 - xbar2)/SE
2 * pt(T, df = n1+n2-2)       # T is negative, two sided


## [1] 1.224317e-43
```

9.36 The data is not independent among the samples as there are only 205 parents and 930 children represented. Treating it as though the pairs are independently drawn, we can use a paired t-test to get

```
with(Galton, t.test(child,parent, paired=TRUE))


##
##  Paired t-test
##
## data:  child and parent
## t = -2.8789, df = 927, p-value = 0.004082
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.36949983 -0.06993982
## sample estimates:
## mean of the differences
##              -0.2197198
```

The small p-value indicates a difference in the means. The confidence interval suggests that it is between 0.07 and 0.37 inches.

10

Goodness of fit

10.3 The data set gives percentages, but `chisq.test` wants proportions. We first define a quick function to compute proportions, then apply the two tests.

The data set is a data frame, where each row is of interest. Some care must be taken when extracting the probabilities, as `chisq.test` needs a vector not a data frame. Here we use `unlist` to provide that.

```
prob <- mandms["milk chocolate", ]
prob <- unlist(prob)
bagful <- c(15, 34, 7, 19, 29, 24)
names(bagful) = c("blue", "brown", "green", "orange", "red", "yellow")
chisq.test(bagful, p=prob/sum(prob))


##
##  Chi-squared test for given probabilities
##
## data:  bagful
## X-squared = 7.0651, df = 5, p-value = 0.2158
```

We repeat with "Peanut:"

```
prob <- mandms["Peanut", ]
prob <- unlist(prob)
chisq.test(bagful, p=prob/sum(prob))


##
##  Chi-squared test for given probabilities
##
## data:  bagful
## X-squared = 13.328, df = 5, p-value = 0.02049
```

It appears that the bag is milk chocolate not peanut, though repeatedly performing significance tests requires an adjustment to the significance level.

10.7 The first test is done quite quickly using the defaults, as the uniform distribution is the null hypothesis.

```
murder <- c(63 , 53 , 50 , 51 , 55 , 52 , 56)
chisq.test(murder)


##
##  Chi-squared test for given probabilities
##
## data:  murder
## X-squared = 2.1263, df = 6, p-value = 0.9077
```

The p-value is not small. It may be said there is no statistically significant difference in the day.

The second we must do by hand. First note that if we specify p_w, the weekend probability, then p_d, the weekday probability, is $(1 - 2p_w)/5$. There is only 1 degree of freedom. We estimate p_w with the average of the weekend counts:

```
n <- sum(murder)
phatw <- (53 + 65)/(2*n)
phatd <- (1 - 2*phatw)/5
e <- n * c(phatw, rep(phatd,5), phatw)
cs <- sum ( (murder - e)^2/e )
cs


## [1] 0.7099884


1 - pchisq(cs, df=1)


## [1] 0.3994477
```

Again, the p-value is not small.

10.11 The accident data can be entered in using `cbind` as follows:

```
accidents <- cbind(
  none=c(67,42,75,56,57),
  minor=c(10,6,8,4,15),
  major=c(5,5,4,6,1))
rownames(accidents) <- c("<18", "18-25", "26-40", "40-65", "65>")
accidents
```

```
##       none minor major
## <18     67    10     5
## 18-25   42     6     5
## 26-40   75     8     4
## 40-65   56     4     6
## 65>     57    15     1
```

```
chisq.test(accidents)
```

```
## Warning in chisq.test(accidents): Chi-squared approximation may
be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  accidents
## X-squared = 12.586, df = 8, p-value = 0.1269
```

10.12 After massaging the data, a glimpse at a contingency table shows us that there appears to be some dependency between the variables. This is borne out by the tiny p-value returned by `chisq.test`.

```
aq <- airquality[complete.cases(airquality),]
aq <- transform(aq,
  te = cut(Temp, quantile(Temp)),
  oz = cut(Ozone,quantile(Ozone))
)
xtabs(~ te + oz, data=aq)
```

```
##            oz
## te          (1,18] (18,31] (31,62] (62,168]
##   (57,71]       15       9       3        0
##   (71,79]       10      10       7        1
##   (79,84.5]      4       6      11        5
```

```
##   (84.5,97]      0       0       6       22
```

```
chisq.test(xtabs(~ te + oz, data=aq))
```

```
##
##  Pearson's Chi-squared test
##
## data:  xtabs(~te + oz, data = aq)
## X-squared = 76.309, df = 9, p-value = 8.713e-13
```

10.17 The two tests can be carried out with

```
shapiro.test(babies$ht[babies$ht < 99])$p.value
```

```
## [1] 4.893686e-10
```

```
shapiro.test(babies$wt[babies$wt < 999])$p.value
```

```
## [1] 0.001191711
```

In each case the null hypothesis would be rejected.

10.19 The Shapiro-Wilk test accepts a null hypothesis of normally distributed data.

```
shapiro.test(normtemp$temperature)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  normtemp$temperature
## W = 0.98658, p-value = 0.2332
```

10.20 A plot of both the empirical density and the theoretical density of the gamma distribution with parameters chosen by `fitdistr` can be produced as follows:

```
library(MASS)
fitdistr(rivers,"gamma")
```

```
##       shape         rate
##   2.578975570   0.004363394
##  (0.268578387) (0.000488203)
```

```
plot(density(rivers))
curve(dgamma(x,shape=2.578, rate= 0.00436), add=TRUE, lty=2)
```

The gamma shape matches the data, but the tail behavior does not seem that similar. A quantile-quantile plot will show this:

```
qqplot(rivers, rgamma(100, shape=2.578, rate= 0.00436))
```

11

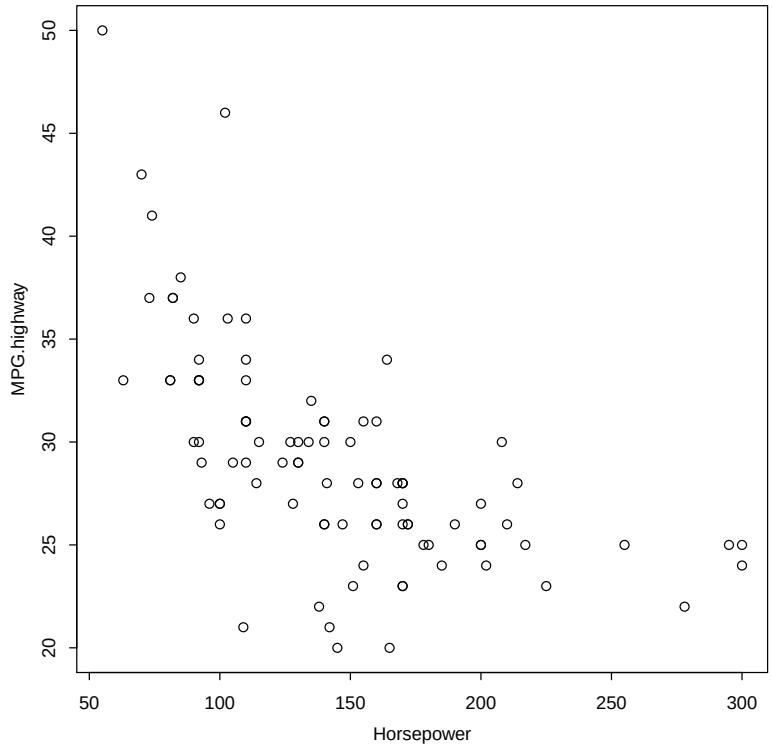
Linear regression

11.1 We begin by loading in the data set and looking at the names.

```
library(MASS)                  # loads data set
```

For the model of highway mileage by horsepower we expect a negative correlation. A scatterplot confirms this.

```
plot(MPG.highway ~ Horsepower, data = Cars93)
res <- lm(MPG.highway ~ Horsepower, data = Cars93)
res


##
## Call:
## lm(formula = MPG.highway ~ Horsepower, data = Cars93)
##
## Coefficients:
## (Intercept)   Horsepower
##    38.14988     -0.06302

predict(res, newdata=data.frame(Horsepower=225))


##        1
## 23.97066
```

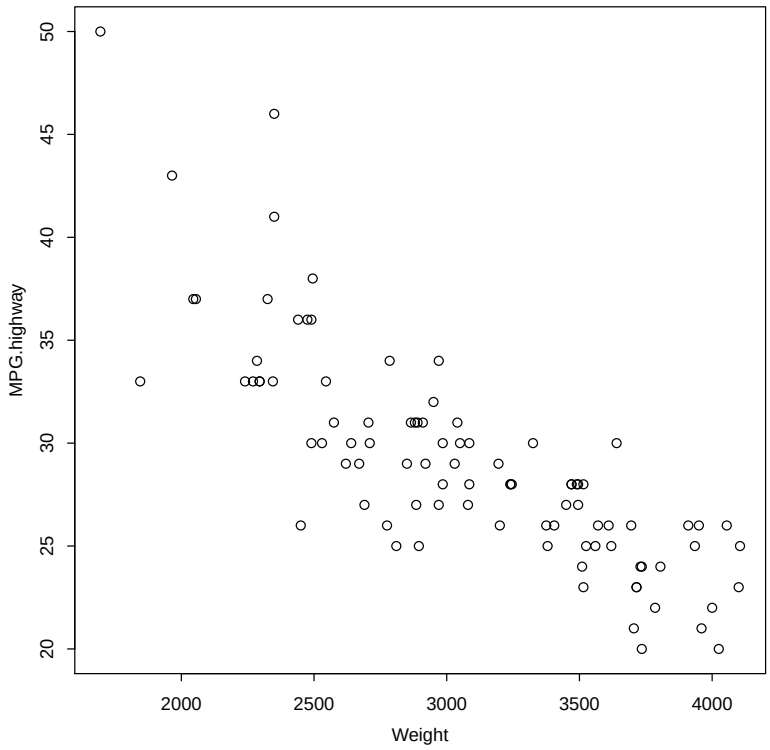


Modeling highway mileage by automobile weight should have a similar negative correlation. Again we confirm and make the requested predictions.

```
f <- MPG.highway ~ Weight
plot(f, data=Cars93)
res <- lm(f, data=Cars93)
res


##
## Call:
## lm(formula = f, data = Cars93)
##
## Coefficients:
## (Intercept)       Weight
##   51.601365    -0.007327
```

```
predict(res, newdata=data.frame(Weight=c(2524, 6400)))
```

```
##         1         2
## 33.107868  4.708186
```

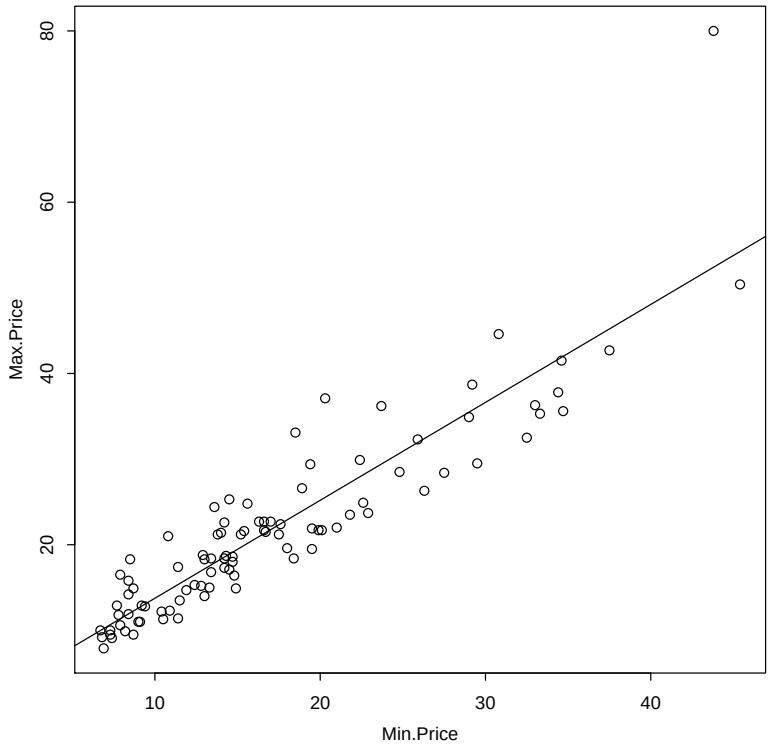


The prediction for the MINI Cooper may be close, but there is no reason to expect the prediction for the HUMMER to be close, as the value of the predictor is outside the range of the data.

The variable `Min.Price` records the value of the stripped-down version of the car, and `Max.Price` records the fully equipped version. We'd expect that `Max.Price` would roughly be a fixed amount more than `Min.Price`, as the differences—the cost of leather seats, a bigger engine, perhaps—are roughly the same for each car. Checking, we have:

```
f <- Max.Price ~ Min.Price
plot(f, data=Cars93)
res <- lm(f, data=Cars93)
```

```
abline(res)
res


##
## Call:
## lm(formula = f, data = Cars93)
##
## Coefficients:
## (Intercept)    Min.Price
##       2.314        1.144
```



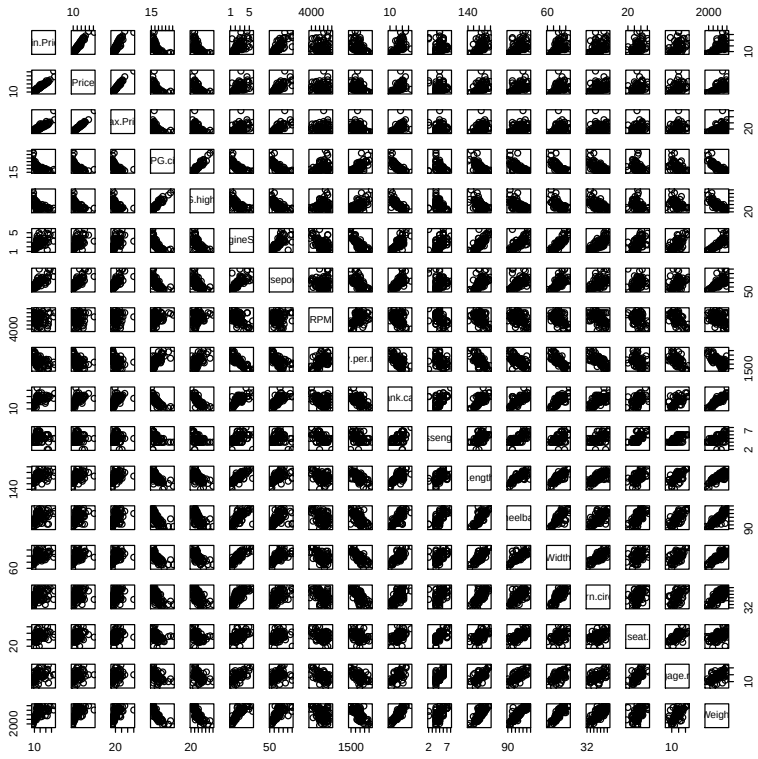
The slope of 1.14 indicates that perhaps add-ons for more expensive cars cost more, but in this case it appears to be due to the one large outlier, as robust regression estimates are much closer to 1:

```
rlm(f, data=Cars93)
```

```
## Call:
## rlm(formula = f, data = Cars93)
## Converged in 7 iterations
##
## Coefficients:
## (Intercept)   Min.Price
##    3.609198    1.028727
##
## Degrees of freedom: 93 total; 91 residual
## Scale estimate: 3.18
```

A scatterplot matrix may show additional linear relationships. These are produced with the `pairs` command, as in `pairs(Cars93)`. Doing so directly produces too many scatterplots. We can trim down the size of the data frame then plot again. Doing so using only the nonfactors can be done as follows:

```
cars <- Cars93[,sapply(Cars93, function(x) !is.factor(x))]
pairs(cars)
```



Looking at the plots produced we see, for example, that variables 1 and 2, 2 and 3, 4 and 5, etc., are linearly related. These variables can be identified from the graphic if the monitor is large enough, or with the command `names(cars)`.

11.6 The p-value is found by computing the t-statistic using $\widehat{\beta}_1$, and $\mathsf{SE}(\widehat{\beta}_1)$ is found from the `summary` function.

```
price <- c(300, 250, 400, 550, 317, 389, 425, 289, 389, 559)
no.bed <- c(3, 3, 4, 5, 4, 3, 6, 3, 4, 5)
res <- lm(price ~ no.bed)
summary(res)


##
## Call:
## lm(formula = price ~ no.bed)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -108.00  -53.95   -5.75   59.77   99.10
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    94.40      97.98   0.963   0.3635
## no.bed         73.10      23.76   3.076   0.0152 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 75.15 on 8 degrees of freedom
## Multiple R-squared:  0.5419,Adjusted R-squared:  0.4846
## F-statistic: 9.462 on 1 and 8 DF,  p-value: 0.01521
```

```
T <- (73.1 - 60)/23.8
pt(T, df=8, lower.tail=FALSE)
```

```
## [1] 0.2985304
```

The large p-value is not significant.

11.9 For illustration, we type the data into a text file with two columns, the first being the elevation. Then we use `read.table` to read it in.

```
x <- read.table(file="tmp.txt")
names(x) <- c("elev", "Temp")
res <- lm(Temp ~ elev, x)
summary(res)
```

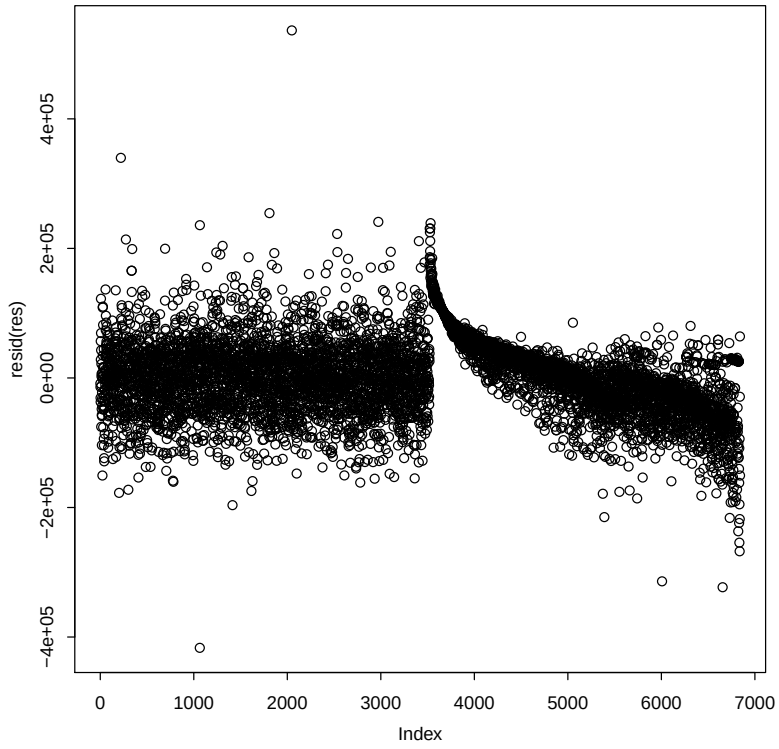
```
##
## Call:
## lm(formula = Temp ~ elev, data = x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0844 -0.4433  0.1369  0.5072  3.1025
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 59.2906619  1.7173294  34.525 3.93e-08 ***
## elev        -0.0051146  0.0009214  -5.551  0.00144 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

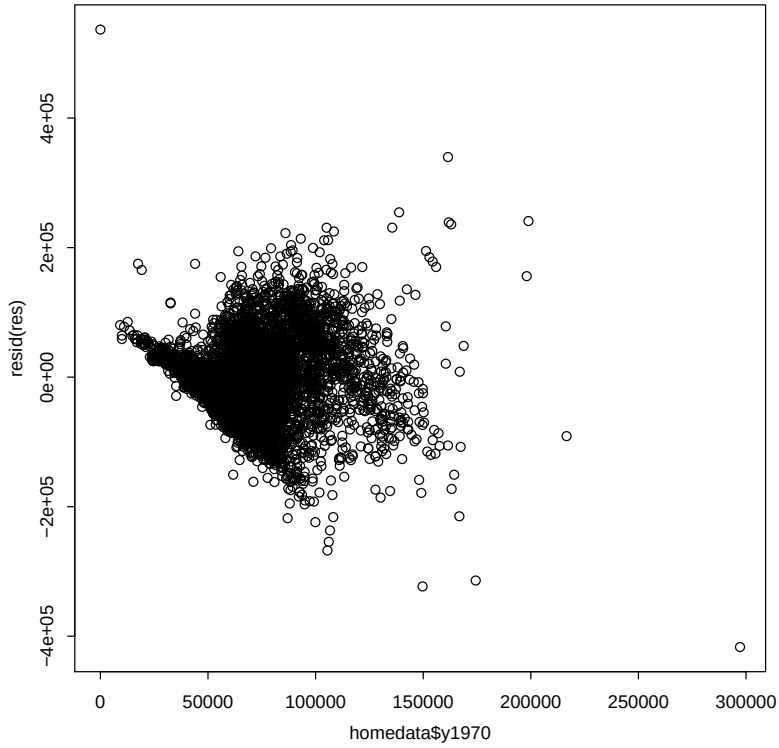
```
##
## Residual standard error: 1.879 on 6 degrees of freedom
## Multiple R-squared:  0.837,Adjusted R-squared:  0.8099
## F-statistic: 30.81 on 1 and 6 DF,  p-value: 0.001445


T <- (-0.005115 - (-0.00534)) / 0.000921
T


## [1] 0.2442997


2*pt(T, df=6, lower.tail=FALSE) # two-sided


## [1] 0.8151384
```

The p-value is not statistically significant.

11.10 The predicted value can be found with `predict`. Though you may not want to believe it, as the model has some issues. The simple plot of the residuals shows values that are scattered around 0, with nothing unusual. However, the second plot using the 1970 values on the x-axis instead of the index, shows that the variance increases with the price of the house.

```
res <- lm(y2000 ~ y1970, data=homedata)
predict(res, newdata=dataframe(y1970=80000))


## Error in predict.lm(res, newdata = dataframe(y1970 = 80000)):
could not find function "dataframe"


predict(res, newdata=data.frame(y1970=80000))


##        1
## 316633.1


plot(resid(res))                  # simple plot
plot(homedata$y1970, resid(res))  # shows spread
```

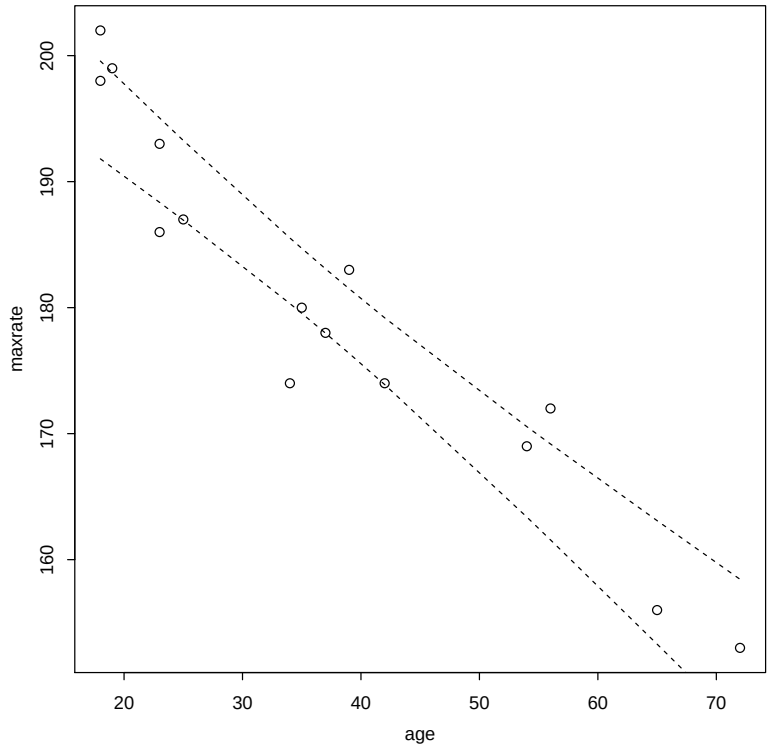

resid(res)
4e+05
2e+05
0e+00
−2e+05
−4e+05
0
1000
2000
3000
4000
5000
6000
7000
Index


resid(res)
4e+05
2e+05
0e+00
−2e+05
−4e+05
0
50000
100000
150000
200000
250000
300000
homedata$y1970

11.13 The linear relationship seems appropriate from a scatterplot. But the plot of the residuals versus the fitted values will show that the variance seems to increase for larger values of the defect size.

11.16 If `res.mhr` stores the model, this can be done as follows:

```
age.sort <- sort(unique(heartrate$age))
pred.res <- predict(res.mhr, newdata=data.frame(age=age.sort), int="conf")
plot(maxrate ~ age, heartrate); abline(res)
lines(age.sort,pred.res[,3], lty=2)
lines(age.sort,pred.res[,2], lty=2)
```



The only change is the abbreviated `int="cont"`.

11.21 No. The p-value is 0.14.

```
init.h <- c(600,700,800,950,1100,1300,1500)
h.d <- c(253, 337, 395, 451, 495, 534, 573)
res.lm3 <- lm(h.d ~ init.h + I(init.h^2) + I(init.h^3))
res.lm4 <- update(res.lm3, . ~ . + I(init.h^4))
anova(res.lm3, res.lm4)

## Analysis of Variance Table
##
## Model 1: h.d ~ init.h + I(init.h^2) + I(init.h^3)
## Model 2: h.d ~ init.h + I(init.h^2) + I(init.h^3) + I(init.h^4)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1      3 48.254
## 2      2 12.732  1    35.522 5.5799  0.142
```

11.25 We can fit the models using `update`:

```
res.1 <- lm(weight ~ age + height, data=kid.weights)
res.2 <- update(res.1, . ~ . + I(height^2))
res.3 <- update(res.2, . ~ . + I(height^3))
res.4 <- update(res.3, . ~ . + I(height^4))
anova(res.1,res.2,res.3,res.4)

## Analysis of Variance Table
##
## Model 1: weight ~ age + height
## Model 2: weight ~ age + height + I(height^2)
## Model 3: weight ~ age + height + I(height^2) + I(height^3)
## Model 4: weight ~ age + height + I(height^2) + I(height^3) + I(height^4)
##   Res.Df   RSS Df Sum of Sq       F    Pr(>F)
## 1    247 33408
## 2    246 30604  1   2803.37 25.3988 9.085e-07 ***
## 3    245 27880  1   2723.95 24.6792 1.274e-06 ***
## 4    244 26931  1    949.08  8.5988  0.003684 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANOVA table shows that for each nested model the new term is statistically significant. The full model is the one selected by this criteria.

11.27 Only the `Weight` variable is selected.

```r
library(MASS)                 # load data set
res = lm(MPG.city ~ EngineSize + Weight + Passengers + Price, data=Cars93)
short_summary(res)
```

```
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 46.3894133  2.0975157 22.1164 < 2.2e-16 ***
## EngineSize   0.1961189  0.5888804  0.3330    0.7399
## Weight      -0.0082072  0.0013429 -6.1115 2.633e-08 ***
## Passengers   0.2696217  0.4249510  0.6345    0.5274
## Price       -0.0358039  0.0491785 -0.7280    0.4685
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
stepAIC(res, trace=0)
```

```
##
## Call:
## lm(formula = MPG.city ~ Weight, data = Cars93)
##
## Coefficients:
## (Intercept)       Weight
##   47.048353    -0.008032
```

12

Analysis of variance

12.1 The data is stored so that the function `oneway.test` is straightforward to use:

```
names(morley)

## [1] "Expt"  "Run"   "Speed"

oneway.test(Speed ~ Expt, data=morley)

##
##  One-way analysis of means (not assuming equal variances)
##
## data:  Speed and Expt
## F = 3.0061, num df = 4.000, denom df = 47.044, p-value =
## 0.02738
```

12.3 A boxplot will show that the income data, like most income data, is heavily skewed.

```
boxplot(income ~ race, data=female.inc)
```

An analysis of variance using `oneway.test` would be inappropriate. However, the three population distributions appear to be roughly the same shape. The Kruskal–Wallis test then is appropriate.

```
kruskal.test(income ~ race, data=female.inc)

##
##  Kruskal-Wallis rank sum test
```

```
##
## data:  income by race
## Kruskal-Wallis chi-squared = 11.794, df = 2, p-value =
## 0.002748
```

The small p-value is not consistent with the assumption of equal mean wages for all three races represented in the data.

12.6 Before using either function, the data is put into the form of a numeric variable to record the values and a factor to record the laboratory.

```
lab1 <- c(4.13, 4.07, 4.04, 4.07, 4.05)
lab2 <- c(3.86, 3.85, 4.08, 4.11, 4.08)
lab3 <- c(4.00, 4.02, 4.01, 4.01, 4.04)
lab4 <- c(3.88, 3.89, 3.91, 3.96, 3.92)
chems <- stack(data.frame(lab1,lab2,lab3,lab4))
boxplot(values ~ ind, data=chems)       # var.equal unlikely
oneway.test(values ~ ind, data=chems, var.equal=FALSE)


##
##  One-way analysis of means (not assuming equal variances)
##
## data:  values and ind
## F = 18.715, num df = 3.0000, denom df = 7.9144, p-value =
## 0.0005927
```

The null hypothesis is that each lab has the same mean. The small p-value suggests that the data is not consistent with this assumption.

12.10 We enter the data and then use `stack` to put it in the proper format:

```
x <- c(63, 64, 95, 64, 60, 85)
y <- c(58, 56, 51, 84, 77)
z <- c(85, 79, 59, 89, 80, 71, 43)
d <- stack(list("test 1"=x, "test 2"=y, "test 3"=z))
plot(values ~ ind, data=d, xlab = "test", ylab="grade")
```

The boxplots show that the assumption of independent samples from a common population, which perhaps is shifted, is appropriate.

The Kruskal–Wallis test returns

```
kruskal.test(values ~ ind, data=d)


##
##  Kruskal-Wallis rank sum test
##
## data:  values by ind
## Kruskal-Wallis chi-squared = 1.7753, df = 2, p-value =
## 0.4116
```

This large p-value indicates no reason to doubt the null hypothesis of equally difficult exams.

12.11 The commands to perform this analysis are

```
short_summary(lm(attendance ~ league, data=MLBattend))


##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1716321      36259 47.3345  < 2e-16 ***
## leagueNL      127296      52093  2.4436  0.01475 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the mean difference of 127,296 fans is significant with a p-value of 0.015.

12.13 The data can be entered as follows:

```
type1 <- c(303, 293, 296, 299, 298)
type2 <- c(322, 326, 315, 318, 320, 320)
type3 <- c(309, 327, 317, 315)
wear <- list(type1=type1, type2=type2, type3=type3)
wear.st <- stack(wear)
```

The use of `stack` stores the data in two variables: the numeric information in `values` and a factor indicating the type in `ind`. The p-value from `oneway.test` can be returned succinctly with

```
oneway.test(values ~ ind, data = wear.st)$p.value


## [1] 0.0001521839
```

Using `lm`, this p-value is returned by the extractor function `summary` in the section labeled `F-statistic`.

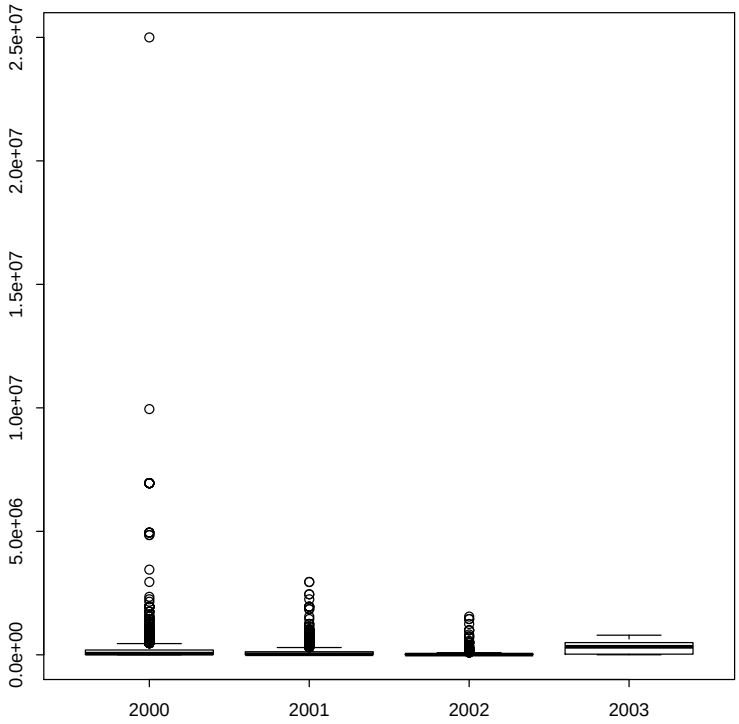
```
summary(lm(values ~ ind, data = wear.st))


##
## Call:
## lm(formula = values ~ ind, data = wear.st)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0000 -2.0833 -0.1667  1.5167 10.0000
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  297.800      2.205 135.077  < 2e-16 ***
## indtype2      22.367      2.985   7.493  7.3e-06 ***
## indtype3      19.200      3.307   5.806  8.4e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.93 on 12 degrees of freedom
## Multiple R-squared:  0.838,Adjusted R-squared:  0.811
## F-statistic: 31.03 on 2 and 12 DF,  p-value: 1.81e-05
```

Why the difference? The F-test assumes equal variances, whereas the default for `oneway.test` assumes unequal variances. Changing the default produces equivalent answers.

```
oneway.test(values ~ ind, data = wear.st, var.equal=TRUE)$p.value


## [1] 1.810398e-05
```

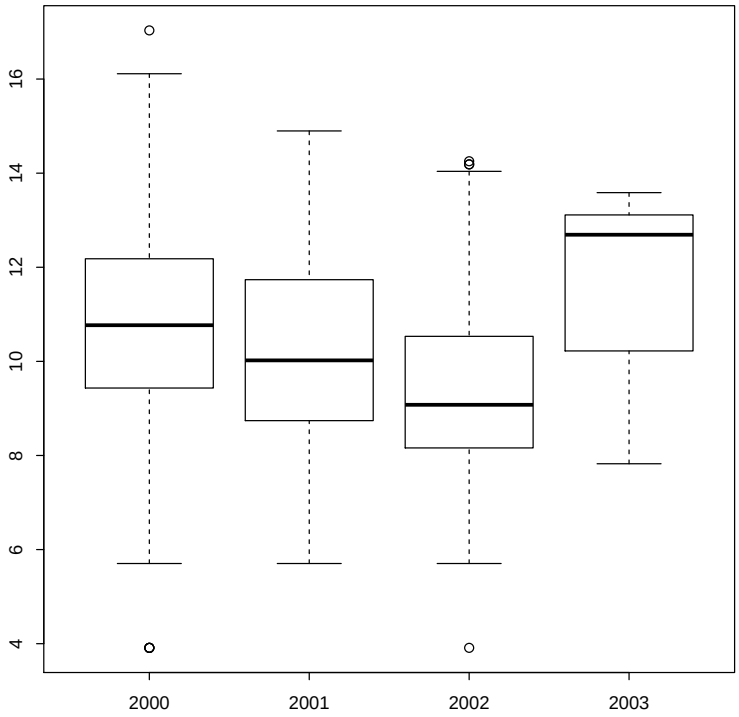
12.16 The boxplot of `amount` broken up by year shows skewed data.

```
boxplot(amount ~ factor(year), data=npdb)
```



Once a logarithm is taken, the data appears to come from a symmetric distribution.

```
boxplot(log(amount) ~ factor(year), data=npdb)
```



A one-way analysis of variance can be performed using `lm` as follows:

```
res <- lm(log(amount) ~ factor(year), subset= year < 2003, npdb)
short_summary(res)
```

```
##                   Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)      10.707768   0.027640 387.4048 < 2.2e-16 ***
## factor(year)2001 -0.487228   0.051887  -9.3902 < 2.2e-16 ***
## factor(year)2002 -1.285061   0.095546 -13.4497 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value for the F-test is tiny, indicating that the null hypothesis is not likely to have yielded this data.

12.17 We need to compare the following

```
short_summary(lm(Speed ~ factor(Expt),data=morley))

##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept)    909.000     16.599 54.7619 < 2.2e-16 ***
## factor(Expt)2  -53.000     23.475 -2.2577 0.0262509 *
## factor(Expt)3  -64.000     23.475 -2.7263 0.0076266 **
## factor(Expt)4  -88.500     23.475 -3.7700 0.0002835 ***
## factor(Expt)5  -77.500     23.475 -3.3014 0.0013561 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

to the output of `TukeyHSD`:

```
TukeyHSD(aov(Speed ~ factor(Expt), data=morley))

##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Speed ~ factor(Expt), data = morley)
##
## $`factor(Expt)`
##      diff        lwr        upr     p adj
## 2-1 -53.0 -118.28006  12.280058 0.1679880
## 3-1 -64.0 -129.28006   1.280058 0.0574625
## 4-1 -88.5 -153.78006 -23.219942 0.0025733
## 5-1 -77.5 -142.78006 -12.219942 0.0115793
## 3-2 -11.0  -76.28006  54.280058 0.9899661
## 4-2 -35.5 -100.78006  29.780058 0.5571665
## 5-2 -24.5  -89.78006  40.780058 0.8343360
## 4-3 -24.5  -89.78006  40.780058 0.8343360
## 5-3 -13.5  -78.78006  51.780058 0.9784065
## 5-4  11.0  -54.28006  76.280058 0.9899661
```

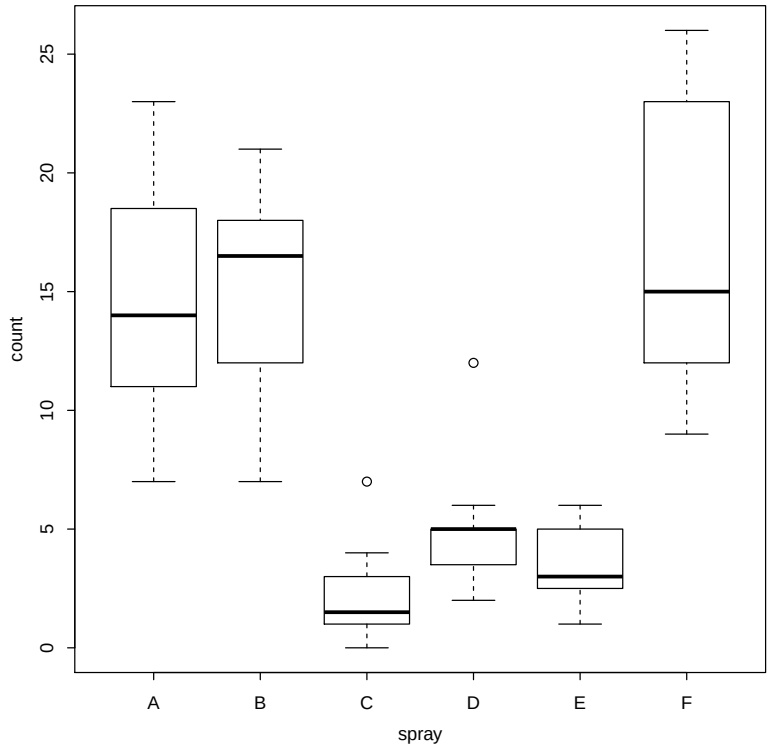
The latter flags `4-1`, `5-1` and `5-4`. The marginal tests find that all the means are different from the reference level `1`.

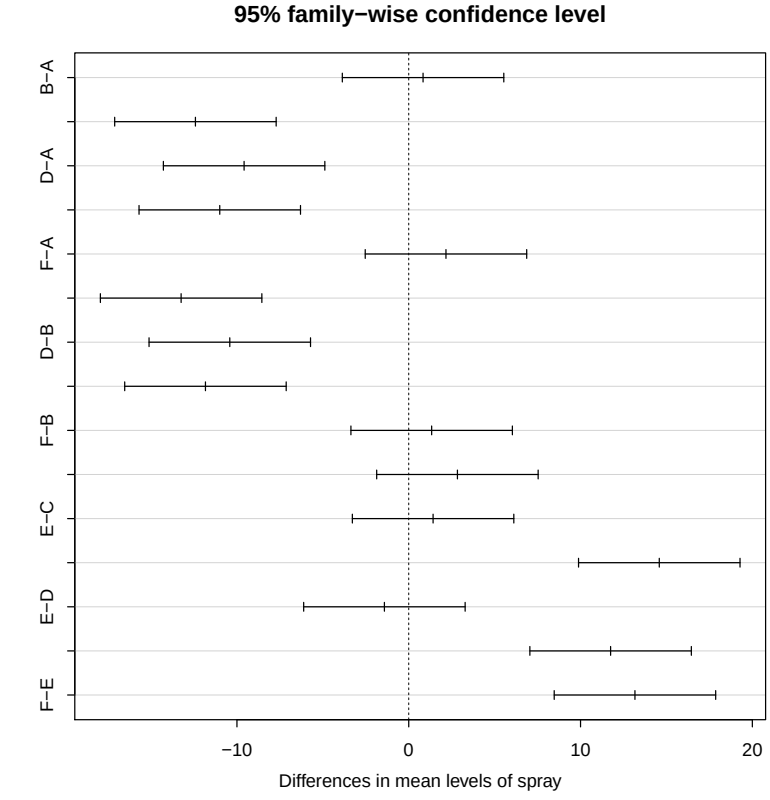
12.19 The following commands will show that `B` and `F` are:

```
plot(count ~ spray, data=InsectSprays)
res.aov <- aov(count ~ spray, data=InsectSprays)
summary(res.aov)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## spray        5   2669   533.8    34.7 <2e-16 ***
## Residuals   66   1015    15.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(TukeyHSD(res.aov))
```


count
0
5
10
15
20
25
A
B
C
D
E
F
spray


95% family−wise confidence level
B−A
D−A
F−A
D−B
F−B
E−C
E−D
F−E
−10
0
10
20
Differences in mean levels of spray

12.20 The ANCOVA is performed as follows:

```
res <- lm(time ~ age + gender, data=nym.2002)
short_summary(res)

##              Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 241.45016    6.25031 38.6301 < 2.2e-16 ***
## age           1.22592    0.15483  7.9178 6.419e-15 ***
## genderMale  -29.39666    3.63343 -8.0906 1.716e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The difference is estimated to be a half-hour in total time.

12.23 We fit the ANCOVA model first:

```
kid.weights$BMI = with(kid.weights, (weight/2.2) / (height*2.54/100)^2)
res.full = lm(BMI ~ age + gender, kid.weights)
res.age  = lm(BMI ~ age, kid.weights)
res.gender  = lm(BMI ~ gender, kid.weights)
anova(res.full, res.age)

## Analysis of Variance Table
##
## Model 1: BMI ~ age + gender
## Model 2: BMI ~ age
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1    247 12911
## 2    248 12911 -1  -0.33189 0.0063 0.9366

anova(res.full, res.gender)

## Analysis of Variance Table
##
## Model 1: BMI ~ age + gender
## Model 2: BMI ~ gender
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1    247 12911
## 2    248 12960 -1   -48.791 0.9334 0.3349
```

We see that neither variable is significant by `stepAIC`:

```
require(MASS)
stepAIC(res)


## Start:  AIC=7087.79
## wt ~ gestation + wt1 + ht + factor(smoke)
##
##                 Df Sum of Sq    RSS    AIC
## - wt1            1     474.4 377907 7087.3
## <none>                       377433 7087.8
## - gestation      1     851.6 378284 7088.6
## - ht             1    6755.5 384188 7107.7
## - factor(smoke)  4   25053.9 402487 7159.2
##
## Step:  AIC=7087.34
## wt ~ gestation + ht + factor(smoke)
##
##                 Df Sum of Sq    RSS    AIC
## <none>                       377907 7087.3
## - gestation      1     831.3 378739 7088.1
## - ht             1    7473.5 385381 7109.5
## - factor(smoke)  4   25128.5 403036 7158.9
##
## Call:
## lm(formula = wt ~ gestation + ht + factor(smoke), data = babies)
##
## Coefficients:
##    (Intercept)       gestation              ht  factor(smoke)1
##       89.37866         0.01097         0.46988        -8.93897
## factor(smoke)2  factor(smoke)3  factor(smoke)9
##        0.55280         1.07946         4.01091
```

12.26 The data can be entered and models fit as follows:

```
likability <- c(-1, 4, 0, -1, 4, 1, 6, 2, 7, 1, 2, 2, 7, 5, 2, 3, 6, 1)
web <- gl(2, 9, 18, c("N", "Y"))
tv <- gl(3, 6, 18, labels=c(0, "1-2", "3+"))

res.web  <- lm(likability ~ web)
res.tv   <- lm(likability ~ tv)
res.both <- lm(likability ~ tv + web)
```

To see whether the web itself has an effect we can look at the output of `summary`:

```
summary(res.web)
```

```
##
## Call:
## lm(formula = likability ~ web)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4444 -2.0278 -0.8333  1.7222  4.5556
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.4444     0.8731    2.80   0.0129 *
## webY          0.7778     1.2348    0.63   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.619 on 16 degrees of freedom
## Multiple R-squared:  0.0242,Adjusted R-squared:  -0.03679
## F-statistic: 0.3968 on 1 and 16 DF,  p-value: 0.5377
```

Web advertising is not effective in this assessment. However, controlling first for television exposure we have

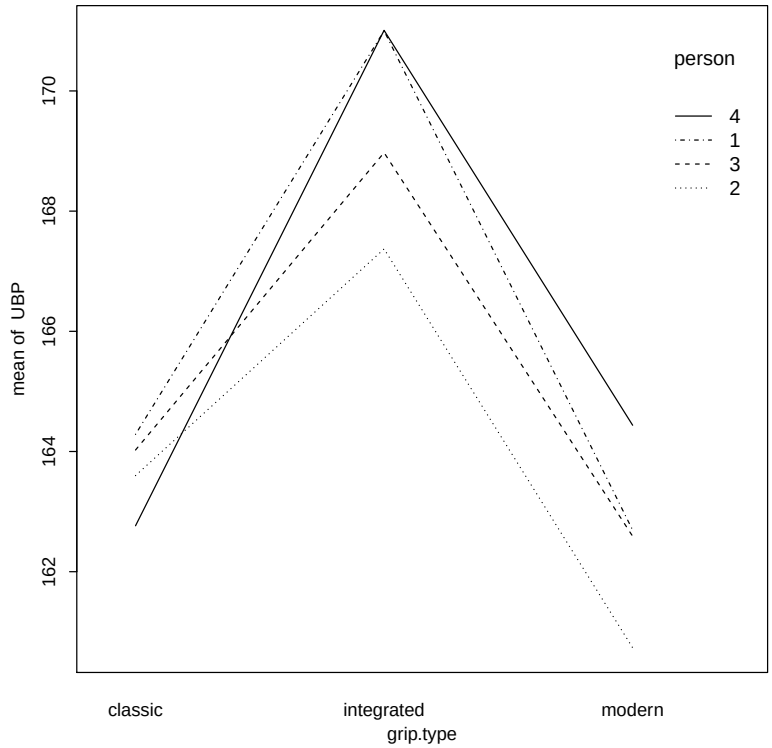
```
anova(res.tv, res.both)
```

```
## Analysis of Variance Table
##
## Model 1: likability ~ tv
## Model 2: likability ~ tv + web
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     15 86.167
## 2     14 69.500  1    16.667 3.3573 0.08826 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

That is, the effect of web advertising seems more likely. It may be worthwhile to perform this test on a bigger sample to try to detect any differences. It is difficult to detect differences with so few observations per category.

12.27 This can be carried out as follows:

```
with(grip, interaction.plot(grip.type,person,UBP))
res.int   <- lm(UBP ~ person * grip.type, data=grip)
res.noint <- lm(UBP ~ person + grip.type, data=grip)
res.per   <- lm(UBP ~ person,             data=grip)
res.grip  <- lm(UBP ~ grip.type,          data=grip)
res.none  <- lm(UBP ~ 1,                  data=grip)
```



Now, we check to see what is statistically significant:

```
anova(res.noint,res.int)

## Analysis of Variance Table
##
## Model 1: UBP ~ person + grip.type
## Model 2: UBP ~ person * grip.type
```

```
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     30 509.01
## 2     24 483.86  6     25.15 0.2079 0.9709
```

This agrees with the interaction plot. No evidence of an interaction is present.

```
anova(res.none,res.per)
```

```
## Analysis of Variance Table
##
## Model 1: UBP ~ 1
## Model 2: UBP ~ person
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     35 875.55
## 2     32 848.19  3     27.35 0.3439 0.7937
```

No evidence that the difference varies among subjects (recall that this is simulated data).

```
anova(res.none,res.grip)
```

```
## Analysis of Variance Table
##
## Model 1: UBP ~ 1
## Model 2: UBP ~ grip.type
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     35 875.55
## 2     33 536.36  2    339.18 10.434 0.0003079 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The effect of the grip seems significant.

Finally, we see that `stepAIC` returns the same conclusion.

```
library(MASS, verbose=FALSE)
stepAIC(res.int)
```

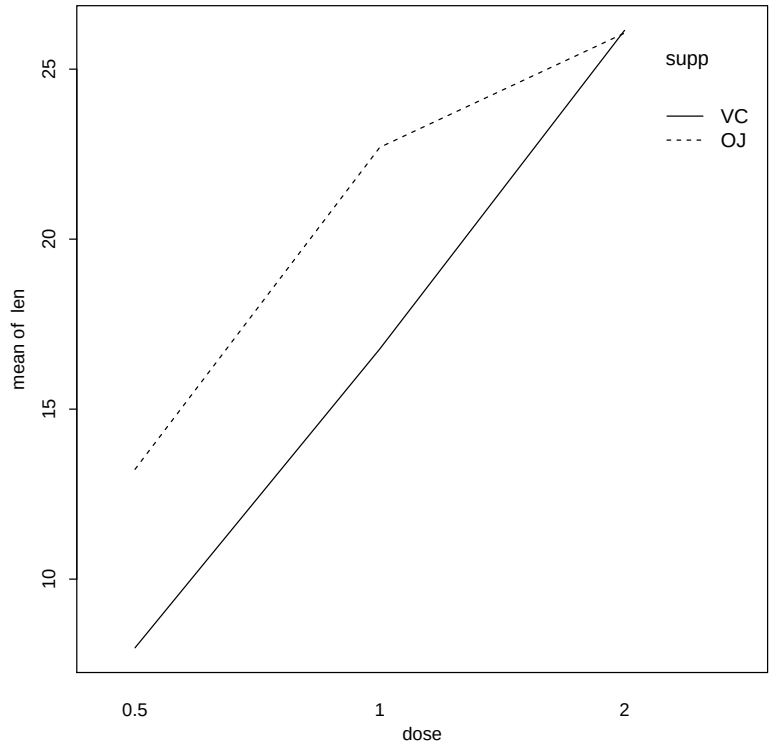
```
## Start:  AIC=117.54
## UBP ~ person * grip.type
##
```

```
##                    Df Sum of Sq    RSS    AIC
## - person:grip.type  6     25.15 509.01 107.36
## <none>                          483.86 117.54
##
## Step:  AIC=107.36
## UBP ~ person + grip.type
##
##             Df Sum of Sq    RSS    AIC
## - person     3     27.35 536.36 103.25
## <none>                   509.01 107.36
## - grip.type  2    339.18 848.19 121.75
##
## Step:  AIC=103.25
## UBP ~ grip.type
##
##             Df Sum of Sq    RSS    AIC
## <none>                   536.36 103.25
## - grip.type  2    339.18 875.55 116.89
##
## Call:
## lm(formula = UBP ~ grip.type, data = grip)
##
## Coefficients:
##         (Intercept)  grip.typeintegrated      grip.typemodern
##             163.669                5.919               -1.055
```

12.29 We proceed as follows:

```
with(ToothGrowth, interaction.plot(dose, supp, len))
res.full <- lm(len ~ supp + dose, data=ToothGrowth)
res.add  <- lm(len ~ supp * dose, data=ToothGrowth)
anova(res.full, res.add)
```

```
## Analysis of Variance Table
##
## Model 1: len ~ supp + dose
## Model 2: len ~ supp * dose
##   Res.Df     RSS Df Sum of Sq      F  Pr(>F)
## 1     57 1022.56
## 2     56  933.63  1     88.92 5.3335 0.02463 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```


supp
VC
OJ
mean of len
10
15
20
25
0.5
1
2
dose

13

Extensions of linear model

13.1 The logistic regression is carried out as follows:

```
res <- glm(enjoyed ~ gender + age, data=tastesgreat,
           family=binomial)
summary(res)
```

```
##
## Call:
## glm(formula = enjoyed ~ gender + age, family = binomial, data = tastesgreat)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.84192  -0.88512  -0.06624   0.74655   2.55961
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.18443    3.09644  -2.643  0.00821 **
## genderMale   2.42241    0.95590   2.534  0.01127 *
## age          0.16491    0.06519   2.530  0.01142 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 55.452  on 39  degrees of freedom
## Residual deviance: 38.981  on 37  degrees of freedom
## AIC: 44.981
##
## Number of Fisher Scoring iterations: 5
```

This shows that both are flagged as significant.

13.8 This model is actually done in the help page for the data set. We can fit the model by running the command

```
library(MASS)                # load in help page
example(wtloss)


##
## wtloss> wtloss.fm <- nls(Weight ~ b0 + b1*2^(-Days/th),
## wtloss+     data = wtloss, start = list(b0=90, b1=95, th=120))
##
## wtloss> wtloss.fm
## Nonlinear regression model
##   model: Weight ~ b0 + b1 * 2^(-Days/th)
##    data: wtloss
##     b0     b1     th
##  81.37 102.68 141.91
##  residual sum-of-squares: 39.24
##
## Number of iterations to convergence: 3
## Achieved convergence tolerance: 4.324e-06
##
## wtloss> plot(wtloss)
##
## wtloss> with(wtloss, lines(Days, fitted(wtloss.fm)))


wtloss.fm


## Nonlinear regression model
##   model: Weight ~ b0 + b1 * 2^(-Days/th)
##    data: wtloss
##     b0     b1     th
##  81.37 102.68 141.91
##  residual sum-of-squares: 39.24
##
## Number of iterations to convergence: 3
## Achieved convergence tolerance: 4.324e-06


plot(Weight ~ Days, wtloss)   # make scatterplot
lines(predict(wtloss.fm) ~ Days, data = wtloss)
```

The value of `b0` is the predicted long-term weight. If we want the predicted amount after 365 days we have

```
predict(wtloss.fm, newdata=data.frame(Days=365))


## [1] 98.64172
```

(Of course, predictions beyond the range of the data are not a good idea.)

13.9 We can fit both using the same starting values and compare:

```
l  <- function(t, a, b, k, t0) (a + b*t) * (1 - exp(-k*(t-t0)))
l1 <- function(t, a, k, t0) l(t, a, 0, k, t0)
res.l <- nls(length ~ l(age,a,b,k,t0), data=reddrum,
             start=c(a=32,b=.25,k=.5,t0=0))
res.l1 <- nls(length ~ l1(age,a,k,t0), data=reddrum,
              start <- c(a=32,k=.5,t0=0))
AIC(res.l)


## [1] 308.6806


AIC(res.l1)


## [1] 378.1829
```

So the more complicated model is better by this criteria.

(The point of the paper that this data came from is to show that both of these models pale in comparison to the more complicated one they presented, which takes into account seasonal growth and a decrease in growth rate due to age.)

13.10 First we make a new year variable, then we fit the model. We use the new value of the car to estimate N.

```
year <- with(midsize, 2004-Year)
f <- function(x, N, r) N * exp(-r*x)
with(midsize, nls(Accord ~ f(year,N,r), start=c(N=Accord[1], r=1/5)))


## Nonlinear regression model
##   model: Accord ~ f(year, N, r)
##    data: parent.frame()
##         N         r
```

```
## 1.670e+04 1.403e-01
##  residual sum-of-squares: 1311037
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 1.602e-06
```

```
with(midsize, nls(Camry ~ f(year,N,r),  start=c(N=Camry[1],  r=1/5)))
```

```
## Nonlinear regression model
##   model: Camry ~ f(year, N, r)
##    data: parent.frame()
##         N         r
## 1.895e+04 1.578e-01
##  residual sum-of-squares: 2288709
##
## Number of iterations to convergence: 5
## Achieved convergence tolerance: 1.087e-06
```

```
with(midsize, nls(Taurus ~ f(year,N,r), start=c(N=Taurus[1], r=1/5)))
```

```
## Nonlinear regression model
##   model: Taurus ~ f(year, N, r)
##    data: parent.frame()
##         N         r
## 1.768e+04 2.602e-01
##  residual sum-of-squares: 23029827
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 5.472e-06
```

The Accord has the smallest decay rate, and the Taurus the largest.