A BRIEF INTRODUCTION TO R

This note is meant as a brief introduction to R. Our discussion of R will not cover every tool and every procedure that are discussed in our text. While our coverage is not comprehensive, it should help you get started with R.

Several excellent introductory books on how to use R have been written. We can recommend Crawley, M. J.: The R Book (Wiley, 2007); Everitt, B. S. and Hothorn, T.: A Handbook of Statistical Analyses Using R (Chapman and Hall, 2006), and Venables, W. N. and Smith, D. M.: An Introduction to R (Network Theory, 2002). In addition, you should consult the very detailed description on the R web site.

R is a free software which is available through the internet; it can be downloaded from http://cran.us.r-project.org/. It is very similar to the commercial package S-Plus. R is a language and an environment for statistical computing and graphics. It can be used with Windows 95 or later versions, with a variety of Unix and Linux platforms, and with Apple Macintosh (OS versions later than 8.6).

Here we explain how to run R under Windows. We assume that you have downloaded the most recent version of R from one of the websites. Running R will open up an R window (RGui) and within it an "R Console" window with its prompt ">". R commands may be issued at this point. We will omit the prompt in our subsequent discussion.

R has an extensive help facility. You can get information from the Help function on the top right of your R window. You can access general information; for example, Help > Html help will give you general information on how to get started. Help > R functions will give you information on specific R programs. Try Help > R functions > hist to get information on how to draw a histogram. Alternatively, you can enter help(hist) or help(log) for getting information on the log transformation at the prompt in the R Console; hitting the enter button will execute the command.

R has an extensive set of program packages. It is very likely that the most common packages (such as base, stats, graphics) are loaded automatically when you install the software. If they are not, or if you want to install other more advanced packages, go to Packages > Install package(s) (you may want to go to a website that is close to you) and click on the packages that you want to install (for example qcc for control charts). Then go to Packages > Load package, and load that package. After that the added package should be ready to be used.

Basic commands

R is case-sensitive, so x and X refer to different variables. R operates on named data structures. Data can be entered at the terminal or can be read from an external file. Entering the elements of a vector x – consisting of the four numbers 2, 4, 5, and 7 – one uses the R command

```
x <- c(2,4,5,7)  or  x = c(2,4,5,7)
```

This is an assignment statement using the function c(). Notice that the assignment operator "<-" (which is the same as the "=" operator) consists of the two characters < ("less than") and - ("minus") and points to the object receiving the value of the expression. For simplicity we use "=". Here and in the following we use the font **Ariel** to indicate what is entered by the user.

Let us consider the data in Exercise 1.3-8 of Section 1.3. There we have thickness measurements of n = 150 paint cans. We enter the data into the object "thick".

```
thick=c(29,36,39,34,34,29,29,28,32,31,34,34,39,38,37,
35,37,33,38,41,30,29,31,38,29,34,31,37,39,36,
30,35,33,40,36,28,28,31,34,30,32,36,38,38,35,
35,30,37,35,31,35,30,35,38,35,38,34,35,35,31,
34,35,33,30,34,40,35,34,33,35,34,35,38,35,30,
35,30,35,29,37,40,31,38,35,31,35,36,30,33,32,
35,34,35,30,36,35,35,31,38,36,32,36,36,32,36,
36,37,32,34,34,29,34,33,37,35,36,36,35,37,37,
36,30,35,33,31,35,30,29,38,35,35,36,30,34,36)
```

`print(thick)`	prints out the data
`thick[12]`	is the 12^{th} observation in the list.
`thickordered=order(thick)`	orders the data set from smallest to the largest.
`thickorder[12]`	is the observation with rank 12

Once we have entered the data, we can perform various operations on the data. We can obtain summary statistics, construct a histogram, dot plot, box plot, and so on.

`summary(thick)`	calculates summary statistics (min, first quartile, median, mean, third quartile, max)
`quantile(thick,probs=c(0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9))`	calculates the percentiles of order 0.1, 0.2, …, 0.9

`a1=mean(thick)`
`a2=median(thick)`
`a3=sd(thick)`
`a4=var(thick)`
	These commands calculate and store in the object to the left of the equality, the mean, median, standard deviation, and variance of the 150 observations.

`a4  return`	prints out the variance
`boxplot(thick)`	creates the box and whisker diagram
`hist(thick)`	creates the histogram for further options, look at the help command for hist
`stem(thick)`	creates the stem-and-leaf display
`dotPlot(thick)`	creates the dotplot. Note that the package BHH2 is needed.

Let us consider the data in Table 1.5-1, that lists the weight (x) and the fuel efficiency (y in gallons per 100 miles) of n = 10 cars. The third column represents the name of the car.

X=Weight	Y=GPM	Car
3.4	5.5	AMC Concord
3.8	5.9	Chevy Caprice
4.1	6.5	Ford Country
2.2	3.3	Chevette
2.6	3.6	Toyota Corona
2.9	4.6	Ford Mustang
2.0	2.9	Mazda GLC
2.7	3.6	AMC Sprint
1.9	3.1	VW Rabbit
3.4	4.9	Buick Century

Entering data

Method 1: Reading in each variable individually

```
ygpm=c(5.5,5.9,6.5,3.3,3.6,4.6,2.9,3.6,3.1,4.9)        gallons per 100 miles
xweight=c(3.4,3.8,4.1,2.2,2.6,2.9,2.0,2.7,1.9,3.4)     weight in 1000 pounds
ygpm    return      prints out ygpm
xweight return      prints out xweight
```

Method 2: Reading in the data from the text file that is scored on our website/your computer:

The data sets in this book can be read from external text files. The commands given below show you how to read the text files that had been prepared for this book. Assume that text files are stored on your computer on drive G. Note that the command "header = T" indicates that we have labeled the columns. With the option "fill = TRUE", blank fields are implicitly added (and indicated by na) in case the rows have unequal length.

Chapter 1:

```
data=read.table("G:/Section1.2Exercise1.2-1Thermostat.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.2Temperatures.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Exercise1.3-2MeltingPoint.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Exercise1.3-3Lead1976.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Exercise1.3-8Thickness.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Exercise1.3-13Hurricane.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Exercise1.3-14BatchYield.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Exercise1.3-18Survey.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3Table1.3-1Strength.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3TestScoresN=58.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.3TestScoresN=44.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.4Exercise1.4-3Jaffe.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.4Exercise1.4-6TwoProcesses.txt", header = T, fill = TRUE)
```

```
data=read.table("G:/Section1.4Table1.4-1MisfeedingLeads.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.4LakeNeusiedl.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.4Lead1976&1977.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Exercise1.5-4Fisher.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Exercise1.5-5Tukey.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Exercise1.5-6AirPollution.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Exercise1.5-7Time.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Exercise1.5-8ACT.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Exercise1.5-9Salary.txt", header = T, fill = TRUE)
data=read.table("G:/Section1.5Table1.5-1Cars.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter1Project1MetalCutting.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter1Project2IowaFatalities.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter1Project2IowaVMT.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter1Project3Trucks.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter1Project9NHTemp.txt", header = T, fill = TRUE)
data=read.table("G:/Section3.1Exercise3.1-2Hours.txt", header = T, fill = TRUE)
data=read.table("G:/Section3.1Exercise3.1-3Wind.txt", header = T, fill = TRUE)
data=read.table("G:/Section3.6Exercise3.6-1Observations.txt", header = T, fill = TRUE)
data=read.table("G:/Section3.6Exercise3.6-11Surgery.txt", header = T, fill = TRUE)
```

Chapter 4

```
data=read.table("G:/Chapter4Project7ZieglerStudy1.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter4Project10Bootstrap.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter4Project11Permutation.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter4Project14Trucks.txt", header = T, fill = TRUE)
```

Chapter 5

```
data=read.table("G:/Section5.1Exercise5.1-1Grant.txt", header = T, fill = TRUE)
data=read.table("G:/Section5.1Exercise5.1-2Astro.txt", header = T, fill = TRUE)
data=read.table("G:/Section5.1Exercise5.1-10Cartons.txt", header = T, fill = TRUE)
data=read.table("G:/Section5.1Table5.1-1Chart.txt", header = T, fill = TRUE)
data=read.table("G:/Section5.2Table5.2-Capability.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter5Project1Breeding.txt", header = T, fill = TRUE)
```

Chapter 6

```
data=read.table("G:/Section6.1Exercise6.1-2Cuckoo.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.1Exercise6.1-3Strength.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.1Exercise6.1-6Strength.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.1Exercise6.1-10GPA.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.1Exercise6.1-11Salary.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.1Table6.1-2Deflection.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.2Exercise6.2-4Bakery.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.2Exercise6.2-9Youden1.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.2Exercise6.2-10Youden2.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.3Exercise6.3-1WearTester.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.3Exercise6.3-2Resistors.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.3Exercise6.3-3Productivity.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.3Exercise6.3-4Reaction.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.3Exercise6.3-6Sodium.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.3Table6.3-1Strength.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.4Exercise6.4-2Marigold.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.4Exercise6.4-3Tire.txt", header = T, fill = TRUE)
```

```
data=read.table("G:/Section6.4Exercise6.4-4Cheese.txt", header = T, fill = TRUE)
data=read.table("G:/Section6.4Table6.4-5Yield.txt", header = T, fill = TRUE)
```

Chapter 7

```
data=read.table("G:/Section7.1Exercise7.1-3Sales.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.1Exercise7.1-4Brightness.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.1Exercise7.1-5Break.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.1Exercise7.1-9StressTest.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.1Table7.1-5Popcorn.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.2Exercise7.2-1Pigment.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.2Exercise7.2-4SteelBeam.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.2Table7.2-1Rod.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-4Conversion.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-5Smoothness.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-6Loss.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-7Impurity.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-9Yield.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-10Conversion.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Exercise7.3-11Meredith.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.3Table7.3-5Fabric.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.4Exercise7.4-8Color.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.4Exercise7.4-9Viscosity.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.4Table7.4-2FractFact1.txt", header = T, fill = TRUE)
data=read.table("G:/Section7.4Table7.4-3FractFact2.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter7Project6MotherJones.txt", header = T, fill = TRUE)
```

Chapter 8

```
data=read.table("G:/Section8.1Exercise8.1-1Bets.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.1Exercise8.1-2Yield.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.1Table8.1-1Cars.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Exercise8.3-2Anscombe.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Exercise8.3-5Aerosol.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Exercise8.3-6Cars.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Exercise8.3-8Enrollment.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Exercise8.3-9Eggs.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Exercise8.3-11WeldStrength.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.3Table8.3-1Steam.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.4Exercise8.4-4CarsNew.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.4Exercise8.4-7TrainStoppingDistances.txt", header = T,fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-1GrowthRate.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-2CloudPoint.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-5Soil.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-6GPA.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-7Traction.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-8Oxygen.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-10Yield.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-11ToolLife.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Exercise8.5-15Trees.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.5Table8.5-6Durability.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.6Exercise8.6-3Yield.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.6Exercise8.6-4Reaction.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.6Exercise8.6-5PercYield.txt", header = T, fill = TRUE)
data=read.table("G:/Section8.6Exercise8.6-7Synthesis.txt", header = T, fill = TRUE)
```

```
data=read.table("G:/Chapter8Project2Wine.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter8Project4Neusiedl.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter8Project5Beans.txt", header = T, fill = TRUE)
data=read.table("G:/Chapter8Project8Assay.txt", header = T, fill = TRUE)
```

The first line of the file Section1.5Table1.5-1Cars.txt specifies a name for each variable in the data frame. The subsequent lines include the values for each variable. To read an entire data frame, we use the command

```
cars=read.table("G:/Section1.5Table1.5-1Cars.txt", header = T, fill = TRUE)
```

this creates the data frame (matrix) cars, with three columns. The command header = T indicates that we have labeled the columns.

cars (and return) Lists the contents of the data frame

You can assign objects as follows. Define the first column of the data matrix "cars" as xweight, the second column as ygpm, and the third column as label.

```
xweight=cars[,1]
ygpm=cars[,2]
label=cars[,3]
```

plot(xweight,ygpm)	constructs the scatterplot of ygpm against xweight
plot(ygpm~xweight)	constructs the scatterplot of ygpm against xweight look at "Help > plot" if you want more flexibility with this graph, such as adding nice titles and labels. You can also connect the data points.
corr(xweight,ygpm)	correlation coefficient

Missing data:

Missing data in R are coded as "na" (for not available). Consider the file Chapter8Project2Wine, as an example.

```
data=read.table("G:/Chapter8Project2Wine.txt", header = T, fill = TRUE)
```

The prices for the 1954 and 1956 vintage are missing. Commands will skip over the rows that contain the missing value. For example the summary command calculates the statistics for price (in data[,5]) from the 27 available rows in that column. Summary statistics on the other columns (such as rain in data[,3]) use all available rows for that column (that is, not just those that have information on all columns). The plot command (plot(data[,5]~data[,3]) of price with the missing data on rain with all data available) plots the information from the 27 available pairs).

The command "na.omit(data)" will omit all rows that contain a missing observation in at least one column.

Simple linear regression

For carrying out the regression, use the "lm" command.

Model 1: Regression of gallons per 100 miles on weight

```
model1=lm(ygpm~xweight)      specifies the simple linear regression
summary(model1)      provides the least squares estimates, their standard errors, the
                     Rsquare, residual standard error (the square root of MSE)
anova(model1)        supplies the analysis of variance (ANOVA) table.
plot(xweight,ygpm)   plots ygmp (the y-coordinate) against xweight (the x-coordinate).
abline(model1)       adds the fitted line to the scatter plot
fit1=fitted(model1)  stores the fitted values in fit1
res1=resid(model1)   stores the residuals in res1

n=length(ygpm)                stores the length of the vector as n (here n = 10)
a=matrix(nrow=n,ncol=4)       defines a matrix with 10 rows and 4 columns
a[,1]=xweight
a[,2]=ygpm
a[,3]=fit1
a[,4]=res1
a  return          prints out the matrix a
```

Model 2: Regression of miles per gallon on weight

```
ympg=100/ygpm                creates the variable miles per gallon
model2=lm(ympg~xweight)      fits the linear regression of miles/gallon on weight
summary(model2)
anova(model2)
plot(xweight,ympg)   plots ympg (the y-coordinate) against xweight (the x-coordinate).
abline(model2)       fitted line is added to the scatter plot
fit2=fitted(model2)  stores the fitted values in fit2
res2=resid(model2)   stores the residuals in res2
```

Model 3: Regression of miles per gallon on weight and weight2 (quadratic model)

```
xweight2=xweight*xweight     creates the square of weight
model3=lm(ympg~xweight+xweight2)
summary(model3)
anova(model3)
fit3=fitted(model3)          stores the fitted values in fit3
res3=resid(model3)           stores the residuals in res3
plot(res3~fit3)              residuals versus fitted values
plot(res3~xweight)           residuals versus regressor variable
```

Working with probabilities and generating random variables

Consider a normal distribution with mean $\mu = 2$ and standard deviation $\sigma = 3$. You obtain the density of the distribution at a certain value *x*, $f(x)$, from dnorm(*x*,3,2), the cumulative probability $P(X \le x)$ from pnorm(*x*,3,2), the 80^{th} percentile from qnorm(0.8,3,2), and you can generate 100 realizations from that distribution with rnorm(100,3,2).

It is fairly obvious how to extend this to other distributions. For example dt(*x*,10) for the t-distribution; dgamma(*x*,alpha,1/beta) for the gamma distribution, dbinom(*x*,n,p) for the binomial distribution, dpoisson(*x*,lambda) for the Poisson distribution, and so on.

For example, with *x* = 1.5

```
dnorm(1.5,3,2)
[1] 0.1505687
```
$f(x = 1.5)$

```
pnorm(1.5,3,2)
[1] 0.2266274
```
$P(X \le 1.5)$

```
qnorm(0.9,3,2)
[1] 5.563103
```
80^{th} percentile

```
gengamma=rgamma(100,2,0.2)
mean(gengamma)
[1] 9.105005
```

Normal probability plot (quantile – quantile plot)

```
gennormal=rnorm(100,3,2)
qqnorm(gennormal)
```
[Note the difference to Minitab: the axes are reversed]

Time-sequence plot

Consider the temperature anomalies from 1600 to 1979 in file Section1.2Temperatures.txt.

```
data=read.table("G:/Section1.2Temperatures.txt", header = T, fill = TRUE)
meas=data[,2]
time=data[,1]
plot.ts(meas)
plot.ts(time,meas,xy.lines=TRUE)
```

Confidence intervals and hypotheses testing

One sample t-test:

```
t.test(data,alternative=c("two.sided"),mu=2,conf.level=0.90)
```

Two sample t-test:

```
t.test(data1,data2,alternative=c("two.sided"),mu=0,conf.level=0.90)
t.test(data1,data2,alternative=c("less"),mu=0,var.equal=TRUE,conf.level=0.90)
```

Paired t-test:

```
t.test(data1,data2,alternative=c("two.sided"),mu=0, paired=TRUE,conf.level=0.90)
```

Testing equality of 2 variances:

```
var.test(data1,data2,ratio=1,alternative=c("two.sided","less","greater"),conf.level = 0.95)
```

Chi-square test of independence:

```
count=matrix(c(15,5,18,22,7,23),nrow=2)
```

Data are from Table 4.7-2. Note the frequencies are read column-wise

chisq.test(count)	performs the chi-square test of independence
chisq.test(count)$expected	prints out expected values

Control charts

Need to load the qcc library for control charts

qcc(qcc)	Quality Control Charts
qcc.groups(qcc)	Grouping data based on a sample indicator

ANOVA methods (for the analyses in chapters 6 and 7)

xfac=factor(x) creates a categorical variable

```
x=c("tr","tr","hj","hj")
xfac=factor(x)
xfac
[1] tr tr hj hj
Levels: hj tr
```

For fitting the ANOVA-type models (one-way lay out, two-way lay out, nested hierarchical design, factorial experiments, Latin squares) we use the "lm" command. Use the help function to learn about all features of this very flexible and extremely useful function.

After reading in the data, we need to declare the categorical variable as a factor.

Experiment with a single factor:

Consider the file Section6.1Table6.1-2Deflection.txt.

```
data=read.table("G:/Section6.1Table6.1-2Deflection.txt", header = T, fill = TRUE)
meas=data[,1]
beam=data[,2]
facbeam=factor(beam)
m1=lm(meas~facbeam)
anova(m1)
summary(m1)
fit=fitted(m1)          stores the fitted values
res=resid(m1)           stores the residuals
m11=lm(meas~facbeam-1)  fits the model without an intercept
anova(m11)              same as anova(m1)
summary(m11)            means, as it omits the interecept
```

You can also use the "aov" command:

```
m12=aov(meas~facbeam)
anova(m12)
summary(m12)
tukeyHSD(m12, ordered = TRUE)      for the calculation of Tukey's multiple comparisons
plot(TukeyHSD(m12))
```

You can also use the oneway.test command:

```
oneway.test(meas~facbeam, var.equal = FALSE)
```

Experiment with two factors:

For an experiment with two factors, consider the popcorn example in file Section7.1Table7.1-5Popcorn.txt.

```
data=read.table("G:/Section7.1Table7.1-5Popcorn.txt", header = T, fill = TRUE)
yield=data[,1]
facpopper=factor(data[,2])
facbrand=factor(data[,3])
m2=lm(yield~facpopper+facbrand)     fits a model with main effects for the factors
                                    popper and brand, but without the
                                    interaction
anova(m2)                           various outputs; see Help for detail
summary(m2)

m3=lm(yield~facpopper*facbrand)     fits a model with main effects for the factors
                                    popper and brand, and an interaction
anova(m3)
summary(m3)
```

Nested factors and hierarchical design:

For an experiment with nested factors (hierarchical design), consider the popcorn example in file Section7.2Table7.2-1Rod.txt.

```
data=read.table("G:/Section7.2Table7.2-1Rod.txt", header = T, fill = TRUE)
meas=data[,3]
facrod=factor(data[,1])
facforging=factor(data[,2])
m4=lm(meas~facrod/facforging)
anova(m4)
summary(m4)
```

Factorial designs:

For the 2^4 factorial experiment in Section7.3Table7.3-5Fabric.txt,

```
data=read.table("G:/Section7.3Table7.3-5Fabric.txt", header = T, fill = TRUE)
x1=factor(data[,1])
x2=factor(data[,2])
x3=factor(data[,3])
x4=factor(data[,4])
meas=data[,5]
m5=lm(meas~x1*x2*x3*x4)          Fits all main effects and interactions
summary(m5)
anova(m5)

m6=lm(meas~x1+x2+x3+x4+x1:x2+x1:x3+x1:x4+x2:x3+x2:x4+x3:x4)
                                 Fits all main effects and 2-factor interactions only
summary(m6)
anova(m6)
```

At this time the R support for the generation and analysis of fractional factorial designs is weak – certainly worse than the features in Minitab. However, packages are added continuously, and we suggest that you check the most recent help files.

Random effects can be handled with the R function "lme" in the package nlme. This is not discussed here as the material is more advanced.

Multiple regression

In Chapters 8 we consider multiple linear regression models. These models can be fit quite easily with R. Suppose we have data in the vectors y, x1, x2 and x3. We can fit a multiple linear regression of y on x1, x2, and x3 by using the command

```
mregfit=lm(y~x1+x2+x3)
```

Information about the model is in the fitted model object "mregfit". Note that an intercept term is included by default. One can restrict the intercept to be zero through

```
mulregfit=lm(y~x1+x2+x3-1)
```

The above commands can be fine-tuned according to specific requirements. Many other commands are available to perform various statistical analyses and plots (such as residual analysis, leverages, Cook's D, various residual plots).

summary(mregfit)	Basic regression commands to get the output
anova(mregfit)	ANOVA
fitted(mregfit)	fitted values
resid(mregfit)	residuals