Tutorial: Using the R Environment for Statistical Computing
An example with the Mercer & Hall wheat yield dataset

D G Rossiter
Department of Earth Systems Analysis
International Institute for Geo-information Science & Earth Observation (ITC)
Enschede (NL)

July 18, 2009

Contents

1 Introduction

This tutorial introduces the **R environment for statistical computing and visualisation** [8, 19] and its dialect of the **S language**. It is organized as a systematic analysis of a simple dataset: the Mercer & Hall wheat yield uniformity trial (Appendix A). After completing the tutorial you should:

- know the basics of the R environment;
- be able to use R at a beginning to intermediate level;
- follow a systematic approach to analyze a simple dataset.

The tutorial is organized as a set of **tasks** followed by **questions** to check your understanding; **answers** are at the end of each section. If you are ambitious, there are also some challenges: tasks and questions with no solution provided, that require the integration of skills learned in the section.

Note: For an explanation of the R project, including how to **obtain and install** the software and documentation, see Rossiter [26]. This also contains an extensive discussion of the **S language**, **R graphics**, and many statistical methods, as well as a **bibliography** of texts and references that use R.

2 R basics

Before entering into the sample data analysis (§3), we first explain how to interact with R, and the basics of the S language.

Task 1 : Start R. •

How you do this is system-specific. Under Microsoft WIndows, open the executable program `RGui.exe`; under Mac OS/X, open the application program `R.app`. Rossiter [26, §3] explains how to run the ITC network installation of R.

After starting R, you will be looking at a **console** where you interact with R: giving commands and seeing numerical results; graphs are displayed in their own windows. You perform most actions in R by typing **commands** in response to a **command prompt**, which usually looks like this:

```
>
```

The `>` is a **prompt symbol** displayed by R, **not** typed by you. This is R's way of telling you it's waiting for you to enter a command.

Type your command[1] and press the `Enter` or `Return` keys; R will **execute** (carry out) your command.

Sometimes the command will result in numerical output listed on the console, other times in a graph displayed in a separate window, other times R will just do what you requested without any feedback.

[1] or cut-and-paste from a document such as this one

If your entry is not a complete R command, R will prompt you to complete it with the **continuation prompt symbol**:

```
+
```

R will accept the command once it is **syntactically complete**; in particular any parentheses must balance. Once the command is complete, R will execute it.

Several commands can be given on the same line, separated by `;`. A command may be interrupted by pressing the `Esc` key.

To illustrate this interaction, we draw a sample of **random numbers** from the **uniform probability distribution**; this is an example of **simulation**.

> **Note:** The code in these exercises was tested with Sweave [10, 11] on R version 2.9.0 (2009-04-17), `sp` package Version: 0.9-37, `gstat` package Version: 0.9-60, and `lattice` package Version: 0.17-25 running on Mac OS X 10.5.6. So, the text and graphical output you see here was automatically generated and incorporated into LaTeX by running actual code through R and its packages. Then the LaTeX document was compiled into the PDF version you are now reading. Your output may be slightly different on different versions and on different platforms.

Task 2 : Draw 12 random numbers uniformly distributed from -1 to 1, rounded to two decimal places, and sort them from smallest to largest. •

In this tutorial we show the **R code** like this, with the prompt `>` and then then command:

```
> sort(round(runif(12, -1, 1), 2))
```

Note that the prompt `>` is *not* part of the command typed by the user; it is presented by the R console.

We show the **output** printed by R like this:

```
 [1] -0.87 -0.65 -0.57 -0.49 -0.30 -0.25 -0.22 -0.19  0.13  0.17  0.28
[12]  0.81
```

The numbers in brackets, like `[1]`, refer to the position in the output vector. In the example above, the 12th element is `0.81`.

This first example already illustrates several features of R:

1. It includes a large number of **functions** (here, `runif` to generate **r**andom numbers from the **unif**orm distribution; `round` to **round** them to a specified precision; and `sort` to **sort** them);
2. These functions have **arguments** that specify the exact behaviour of the function. For example, `round` has two arguments: the first is the object to be rounded (here, the vector returned by the `runif` function) and the second is the number of decimal places (here, `2`);

3. Many functions are **vectorized**: they can work on vectors (and usually matrices) as well as scalars. Here the `round` function is modifying the results of the `runif` function, which is a 12-element vector;
4. Values **returned** by a function can be immediately used as an argument to another function. Here the results of `runif` is the vector to be rounded by the `round` function; and these are then used by the `sort` function. To understand a complex expression, read it from the inside out.
5. R has a rich set of functions for **simulation** of **random processes**.

Q1 : *Your results will be different from the ones printed in this note; why?* *Jump to A1* •

To see how this works, we can do the same operation step-by-step.

1. Draw the random sample, and save it in a **local variable** in the **workspace** using the `<-` (assignment) operator; we also list it on the console with the `print` function:

```
> sample <- runif(12, -1, 1)
> print(sample)
```

```
 [1]  0.86261  0.14612 -0.16739 -0.87683  0.89672 -0.83444 -0.43606
 [8]  0.39042 -0.71586  0.65931 -0.17241  0.55953
```

2. Round it to two decimal places, storing it in the same variable (i.e. replacing the original sample):

```
> sample <- round(sample, 2)
> sample
```

```
 [1]  0.86  0.15 -0.17 -0.88  0.90 -0.83 -0.44  0.39 -0.72  0.66 -0.17
[12]  0.56
```

3. Sort it and print the results:

```
> (sample <- sort(sample))
```

```
 [1] -0.88 -0.83 -0.72 -0.44 -0.17 -0.17  0.15  0.39  0.56  0.66  0.86
[12]  0.90
```

This example also shows three ways of printing R output on the console:

- By using the `print` function with the object name as argument;
- By simply typing the object name; this calls the `print` function;
- By enclosing any expression in parenthesis `( ... )`; this forces another evaluation, which prints its results.

R has an immense repertoire of **statistical methods**; let's see two of the most basic.

Task 3 : Compute the theoretical and empirical mean and variance of a sample of 20 observations from a uniformly-distributed random variable in the range $(0 \ldots 10)$, and compare them. •

The theoretical mean and variance of a uniformly-distributed random variable are [2, §3.3]:

$$\begin{aligned} \mu &= (b + a)/2 \\ \sigma^2 &= (b - a)^2/12 \end{aligned}$$

where a and b are the lower and upper endpoints, respectively, of the uniform interval.

First the theoretical values for the mean and variance. Although we could compute these by hand, it's instructive to see how R can be used as an **interactive calculator** with the usual operators such as `+`, `-`, `*`, `/`, and `^` (for exponentiation):

```
> (10 + 0)/2

[1] 5

> (10 - 0)^2/12

[1] 8.3333
```

Now draw a 20-element sample and compute the sample mean and variance, using the `mean` and `var` functions:

```
> sample <- runif(20, min = 0, max = 10)
> mean(sample)

[1] 4.6969

> var(sample)

[1] 8.0719
```

Q2 : *How close did your sample come to the theoretical value?* Jump to A2 •

We are done with the local variable `sample`, so we **remove** it from the workspace with the `rm` ("**re**move") function; we can check the contents of the workspace with the `ls` ("**l**i**s**t") function:

```
> ls()

[1] "sample"

> rm(sample)
> ls()

character(0)
```

On-line help If you know a function or function's name, you can get **help** on it with the `help` function:

```
> help(round)
```

This can also be written more simply as `?round`.

Q3 : *Use the `help` function to find out the three arguments to the `runif` function. What are these? Are they all required? Does the order matter?* *Jump to A3* •

Arguments to functions We can experiment a bit to see the effect of changing the arguments:

```
> runif(1)
```

```
[1] 0.73248
```

```
> sort(runif(12))
```

```
 [1] 0.18394 0.18574 0.21591 0.26612 0.27124 0.46461 0.51027 0.68645
 [9] 0.73841 0.80623 0.82120 0.93067
```

```
> sort(runif(12, 0, 5))
```

```
 [1] 0.20403 0.64116 1.16720 1.38740 1.47611 1.73486 1.97464 2.39523
 [9] 2.58629 2.67808 2.70262 4.17477
```

```
> sort(runif(12, min = 0, max = 5))
```

```
 [1] 0.18940 0.43644 0.76711 1.70539 1.73055 2.42770 2.46312 2.54696
 [9] 2.55902 3.66288 3.88315 4.95848
```

```
> sort(runif(max = 5, n = 12, min = 0))
```

```
 [1] 0.22801 0.67073 0.88072 1.52585 1.60693 2.72898 2.79125 3.18844
 [9] 3.69058 4.39535 4.49391 4.95517
```

Searching for a function If you don't know a function name, but you know what you want to accomplish, you can search for an appropriate function with the `help.search` function:

```
> help.search("principal component")
```

This will show packages and functions relevant to the topic:

```
stats::biplot.princomp   Biplot for Principal Components
stats::prcomp            Principal Components Analysis
stats::princomp          Principal Components Analysis
stats::summary.princomp  Summary method for Principal Components Analysis
```

Then you can ask for more information on one of these, e.g.:

```
> help(prcomp)
```

```
prcomp              package:stats                    R Documentation

Principal Components Analysis

Description:

Performs a principal components analysis on the given data matrix
and returns the results as an object of class 'prcomp'.

Usage: ...
```

2.1 Leaving R

At this point you should leave R and re-start it, to see how that's done.

Before leaving R, you may want to **save your console log** as a text file to document what you did, and the results, or for later re-use. You can edit this file in any plain-text editor. In the Windows GUI you can use the `File| Save to file...` menu command.

To leave R, use the `q` ("quit") function (in any OS). On Windows you can quit the program in the conventional ways: select the `File | Exit` menu item in the Windows GUI, or click the "Close" icon at the top of the main window.

```
> q()
```

You will be asked if you want to **save your workspace** in the current directory; generally you will want to do this[2]. The next time you start R in the same directory, the saved workspace will be automatically loaded.

In this case we haven't created anything useful for data analysis, so you should quit without saving the workspace.

2.2 Answers

A1 : *Random number generation gives a different result each time.*[3]. *Return to Q1* •

A2 : *This depends on your sample; see the results in the text for an example. Return to Q2* •

A3 : *There are three possible arguments: the number of samples* `n`*, the minimum value* `min` *and the maximum* `max`*. The last two are not required and **default** to* 0 *and* 1*, respectively. If arguments are **named** directly, they can be put in any order. If not, they have to follow the default order.* *Return to Q3* •

[2] By default this file is named `.RData`

[3] To start a simulation at the same point (e.g. for testing) use the `set.seed` function

3 Loading and examining a data set

The remainder of this tutorial uses the Mercer & Hall wheat yield data set, which is described in Appendix A. Please read this now.

There are many ways to get data into R [26, §6]; one of the simplest is to create a **comma-separated values** ("CSV") file in a text editor[4]. For this example we have prepared file `mhw.csv` which is supplied with this note.

Task 4 : From the operating system, open the text file `mhw.csv` with a plain-text editor such as WordPad and examine its structure.

Note: *do not* examine it in Excel; this automatically splits the file into spreadsheet columns, obscuring its structure as a text file. •

The first four lines of the file should look like this:

```
"r","c","grain","straw"
1,1,3.63,6.37
2,1,4.07,6.24
3,1,4.51,7.05
```

Q4 : *What does the first line represent? What do the other lines represent, and what is their structure?* *Jump to A4* •

3.1 Reading a CSV file into an R object

Now we read the dataset into R.

Task 5 : Start R. •

Task 6 : If necessary, make sure R is pointed to the same **working directory** where you have stored `mhw.csv`. You can use the `getwd` function to check this, and `setwd` to change it; in the Windows GUI you can use the `File|Change directory...` menu command. •

```
> getwd()
```

Once the directory is changed, the contents of the file can be displayed with the `file.show` function:

```
> file.show("mhw.csv")

"r","c","grain","straw"
1,1,3.63,6.37
2,1,4.07,6.24
3,1,4.51,7.05
4,1,3.9,6.91
...
```

[4] A CSV file can also be prepared as a spreadsheet and exported to CSV format.

A CSV file can be read into R with the `read.csv` function and **assigned** to an object in the **workspace** using the `<-` **operator** (which can also be written as `=`):

```
> mhw <- read.csv("mhw.csv")
```

Q5 : *Why is nothing printed after this command?* *Jump to A5* •

3.2 Examining a dataset

The first thing to do with any dataset is to examine its **structure** with the `str` function.

```
> str(mhw)

'data.frame':        500 obs. of  4 variables:
 $ r    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ c    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ grain: num  3.63 4.07 4.51 3.9 3.63 3.16 3.18 3.42 3.97 3.4 ...
 $ straw: num  6.37 6.24 7.05 6.91 5.93 5.59 5.32 5.52 6.03 5.66 ...
```

Q6 : *How many observations (cases) are there in this frame? How many fields (variables)? What are the field names?* *Jump to A6* •

We can extract the names for each field (matrix column) with the `names` function; this is equivalent to `colnames`:

```
> names(mhw)

[1] "r"     "c"     "grain" "straw"

> colnames(mhw)

[1] "r"     "c"     "grain" "straw"
```

Every object in R belongs to a **class**, which R uses to decide how to carry out commands.

Q7 : *What is the* ***class*** *of this object?* *Jump to A7* •

We can examine the class with the `class` function:

```
> class(mhw)

[1] "data.frame"
```

A **data frame** is used to hold most data sets. The **matrix rows** are the **observations** or **cases**; the **matrix columns** are the named **fields** or **variables**. Both matrix rows and columns have **names**.

Fields in the data frame are commonly referred to by their matrix column name, using the syntax `frame$variable`, which can be read as “extract the field named `variable` from the data frame named `frame`.

Task 7 : Summarize the grain and straw yields. •

```
> summary(mhw$grain)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.73    3.64    3.94    3.95    4.27    5.16

> summary(mhw$straw)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4.10    5.88    6.36    6.51    7.17    8.85
```

A data frame is also a **matrix**; we can see this by examining its **dimensions** with the `dim` function and extracting elements.

The two dimensions are the numbers of matrix rows and columns:

```
> dim(mhw)

[1] 500   4
```

Q8 : *Which matrix dimension corresponds to the observations and which to the fields?* *Jump to A8* •

Matrix rows, columns, and individual cells in the matrix can be extracted with the `[ ..]` operator; this is just like standard matrix notation in mathematics:

```
> mhw[1, ]

  r c grain straw
1 1 1  3.63  6.37

> length(mhw[, 3])

[1] 500

> summary(mhw[, 3])

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.73    3.64    3.94    3.95    4.27    5.16

> mhw[1, 3]

[1] 3.63
```

Matrix rows and columns can also be accessed by their **names**; here is the grain yield of the first plot:

```
> mhw[1, "grain"]

[1] 3.63
```

Q9 : *What is the grain yield of plot 64? Where is this located in the (experimental) field?* *Jump to A9* •

```
> mhw[64, "grain"]
```

```
[1] 4.04
```

```
> mhw[64, c("r", "c")]
```

```
   r c
64 4 4
```

Note the use of the `c` ("catenate", build a chain) function to build a **list** of two names.

An entire field (variable) can be accessed either by matrix column number or name (considering the object to be a matrix) or variable name (considering the object to be a data frame); the output can be limited to the first and last lines only by using the `head` and `tail` functions. By default they show the six first or last values; this can be over-ridden with the optional `n` argument.

```
> head(mhw[, 3])
```

```
[1] 3.63 4.07 4.51 3.90 3.63 3.16
```

```
> tail(mhw[, "grain"], n = 10)
```

```
 [1] 3.29 3.83 4.33 3.93 3.38 3.63 4.06 3.67 4.19 3.36
```

```
> head(mhw$grain)
```

```
[1] 3.63 4.07 4.51 3.90 3.63 3.16
```

The `order` function is somewhat like the `sort` function shown above, but rather than return the actual values, it returns their position in the array. This position can then be used to extract other information from the dataframe.

Task 8 : Display the information for the plots with the five lowest straw yields. •

To restrict the results to only five, we again use the `head` function.

```
> head(sort(mhw$straw), n = 5)
```

```
[1] 4.10 4.28 4.53 4.56 4.57
```

```
> head(order(mhw$straw), n = 5)
```

```
[1] 470 467 441 447 427
```

```
> head(mhw[order(mhw$straw), ], n = 5)
```

```
     r  c grain straw
470 10 24  2.84  4.10
467  7 24  2.78  4.28
441  1 23  2.97  4.53
447  7 23  3.44  4.56
427  7 22  3.05  4.57
```

Q10 : *What are the values shown in the first command, using `sort`? In the second, using `order`? Why must we use the results of `order` to extract the records in the data frame?* *Jump to A10* •

Records can also be selected with **logical criteria**, for example with **numeric comparison operators**.

Task 9 : Identify the plot with the highest and lowest grain yields. •

The `max` (maximum value in a vector) and `min` (minimum value in a vector) functions are applied to the grain yield field, and this is used as a row selector, along with the `==` "numerical equality" comparison operator:

```
> max(mhw$grain)

[1] 5.16

> min(mhw$grain)

[1] 2.73

> mhw[mhw$grain == max(mhw$grain), ]

    r c grain straw
79 19 4  5.16  8.78

> mhw[mhw$grain == min(mhw$grain), ]

     r  c grain straw
338 18 17  2.73  4.77
```

Note: Other numeric comparison operators are `<`, `>`, `<=`, `>=` and `!=` (not equal).

Q11 : *Why is there nothing between the comma '`,`' and right bracket '`]`'?* *Jump to A11* •

Challenge: Extract all the grain yields from the most easterly (highest-numered) column of field plots, along with the straw yields and field plot row number. Sort them from highest to lowest yields[5], also displaying the row numbers and straw yields. Does there seem to be any trend by field plot row? How closely are the decreasing grain yields matched by straw yields?

3.3 Saving a dataset in R format

Once a dataset has been read into R and possibly modified (for example, by assigning field names, changing the class of some fields, or computing new fields) it can be saved in R's internal format, using the `save` function. The

[5] Hint: see the help for the `order` function to find its optional argument to sort the rows in decreasing order of grain yield; or see the `rev` function.

dataset can then be read into the workspace in a future session with the `load` function.

Task 10 : Save the `mhw` object in R format. •

It is conventional to give files with R objects the `.RData` extension.

```
> save(mhw, file = "mhw.RData")
```

3.4 Answers

A4 : *The first line is a* ***header*** *with the* ***variable names****, in this case* `r`, `c`, `grain` *and* `straw`*. The following lines each represent one* ***plot****; there are four variables recorded for each plot, i.e. its row and column number in the field, and its grain and straw yield.* *Return to Q4* •

A5 : *Commands that store their results in an object (using the* `=` *or* `<-` *operators) do their work silently; if you want to see the results enclose the command in parentheses* `( ... )` *or just type the object name at the command prompt.* *Return to Q5* •

A6 : *There are 500 observations (cases), and for each 4 variables:* `r`, `c`, `grain` *and* `straw`*.* *Return to Q6* •

A7 : *It is in class* `data.frame`*.* *Return to Q7* •

A8 : *Matrix rows are observations, matrix columns are fields.* *Return to Q8* •

A9 : *Grain yield 4.04; this is located at field row 4, field column 4* *Return to Q9* •

A10 : *The* `sort` *function shows the actual values of straw yield;* `order` *shows in which records in the data frame these are found. The record numbers are the key into the dataframe.* *Return to Q10* •

A11 : *So that all fields (matrix columns) are selected.* *Return to Q11* •

4 Exploratory graphics

Before beginning a data analysis, it is helpful to **visualise** the dataset. This is generally the first phase of **exploratory data analysis** (EDA) [29].

R is an excellent environment for visualisation; it can produce simple plots but also plots of great sophistication, information and beauty. We look first at single variables and then at the relation between two variables.

4.1 Univariate exploratory graphics

Task 11 : Visualise the **frequency distribution** of grain yield with a stem plot. •

A **stem-and-leaf** plot, displayed by the `stem` function, shows the numerical values themselves, to some precision:

```
> stem(mhw$grain)

  The decimal point is 1 digit(s) to the left of the |

  27 | 38
  28 | 45
  29 | 279
  30 | 144555557899
  31 | 4446678999
  32 | 2345589999
  33 | 002455666677789999
  34 | 00112233444444566777777888999
  35 | 0111233444455566667778999
  36 | 0001111133333444445666666777778889999
  37 | 0001111112222223334444455555666666777789999
  38 | 0011222223334444455566667777999999
  39 | 0111111112222233333444444555666666777777777999
  40 | 01112233334455566666666777777788888899999999
  41 | 00011111223334455557777779999
  42 | 000011111112223333444444666777777788999999
  43 | 011122333356666667777788889999999
  44 | 0011111222234445566667777899
  45 | 01122222234445667888899
  46 | 1344446678899
  47 | 3356677
  48 | 466
  49 | 12349
  50 | 279
  51 | 3336
```

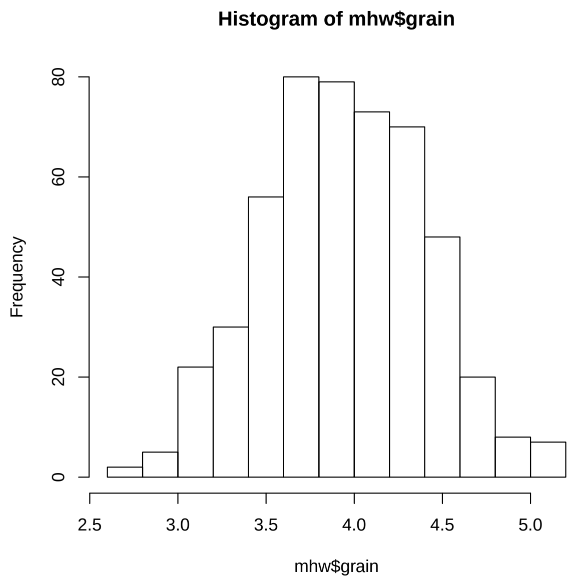
Q12 : *According to the stem-and-leaf plot, what are the approximate values of the minimum and maximum grain yields?* Jump to A12 •

Q13 : *What is the advantage of the stem plot over the histogram?* Jump to A13 •

Task 12 : Visualise the **frequency distribution** of grain yield with a frequency histogram. •

A histogram, displayed by the `hist` function, shows the distribution:

```
> hist(mhw$grain)
```



Saving graphic output

You can **save the graphics** window to any common graphics format. In the Windows GUI, bring the graphics window to the front (e.g. click on its title bar), select menu command `File | Save as ...` and then one of the formats.

Q14 : *What are the two axes of the default histogram?* Jump to A14 •

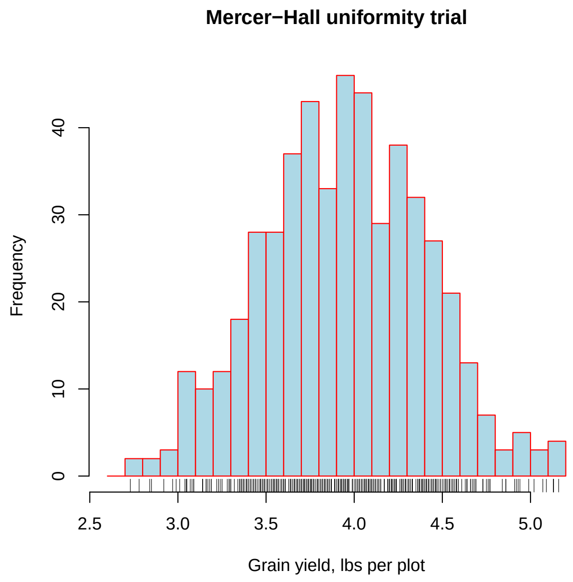
Q15 : *By examining the histogram, how many observations had grain yield below 3 lbs per plot?* Jump to A15 •

4.1.1 Enhancing the histogram*

R graphics, including histograms, can be enhanced from their quite plain default appearance. Here we change the break points with the `breaks` argument, the colour of the bars with the `col` graphics argument, the colour of the border with the `border` graphics argument, and supply a title with the `main` graphics argument.

We then use the `rug` function to add a "rug" plot along the x-axis to show the actual observations. This is an example of a graphics function that **adds** to an existing plot; whereas `hist` creates a **new** plot. Which does which? Consult the help.

```
> hist(mhw$grain, breaks = seq(2.6, 5.2, by = 0.1), col = "lightblue",
+     border = "red", main = "Mercer-Hall uniformity trial",
+     xlab = "Grain yield, lbs per plot")
> rug(mhw$grain)
```



Note the use of the `seq` ("sequence") function to make a list of break points:

```
> seq(2.6, 5.2, by = 0.1)
```

```
 [1] 2.6 2.7 2.8 2.9 3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1
[17] 4.2 4.3 4.4 4.5 4.6 4.7 4.8 4.9 5.0 5.1 5.2
```

In this example, the colours are from a list of known names. For more information on these names, and other ways to specify colours, see Appendix B.

4.1.2 Kernel density*

A **kernel density**, computed with `density` function, fits an empirical curve to a sample supposed to be drawn from a univariate probability distribution [31, §5.6]. It can be used to give a visual impression of the distribution or to **smooth** an empirical distribution.

In the context of EDA, the kernal density can suggest:

- whether the empirical distribution is **unimodal** or **multimodal**;
- in the case of a unimodal distribution, the **theoretical probability density function** from which it may have been drawn.

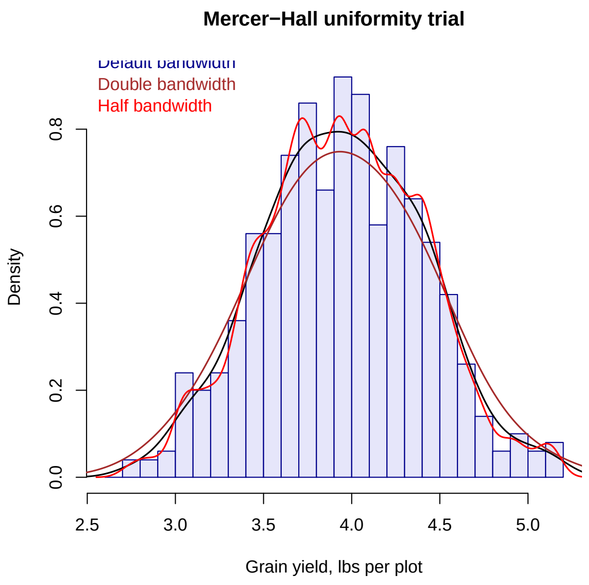
The kernel density is controlled by the `kernel` and `adjust` optional arguments to the `density` function. The default values of `"gaussian"` and `1` select a smoothing bandwidth based on the number of observations and a theoretical normal density.

A special case of the density is a histogram expressed as densities rather than frequencies; this is selected with the optional `freq` ("frequency") argument to the `hist` function set to `FALSE`. The total area under the histogram is then by definition `1`.

The `lines` function can be used to add the empirical density computed by `density` to a density histogram plotted with `hist`. Another interesting view is the kernel density with a rug plot to show the actual values of the sample.

Task 13 : Display a histogram of the grain yields as a density (proportion of the total), with the default kernel density superimposed, along with a double and half bandwidth kernel density. •

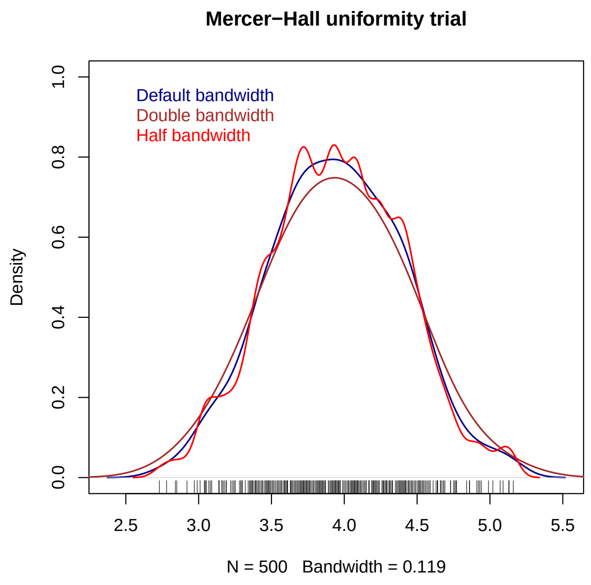
```
> hist(mhw$grain, breaks = seq(2.6, 5.2, by = 0.1), col = "lavender",
+     border = "darkblue", main = "Mercer-Hall uniformity trial",
+     freq = F, xlab = "Grain yield, lbs per plot")
> lines(density(mhw$grain), lwd = 1.5)
> lines(density(mhw$grain, adj = 2), lwd = 1.5, col = "brown")
> lines(density(mhw$grain, adj = 0.5), lwd = 1.5, col = "red")
> text(2.5, 0.95, "Default bandwidth", col = "darkblue",
+     pos = 4)
> text(2.5, 0.9, "Double bandwidth", col = "brown", pos = 4)
> text(2.5, 0.85, "Half bandwidth", col = "red", pos = 4)
```



Task 14 : Repeat, but superimpose the kernel density on a rug plot. •

This time the first plot must be of the density, because `rug` only adds to an existing plot.

```
> plot(density(mhw$grain), lwd = 1.5, ylim = c(0, 1), col = "darkblue",
+     main = "Mercer-Hall uniformity trial")
> rug(mhw$grain, main = "Mercer-Hall uniformity trial",
+     xlab = "Grain yield, lbs per plot")
> lines(density(mhw$grain, adj = 2), lwd = 1.5, col = "brown")
> lines(density(mhw$grain, adj = 0.5), lwd = 1.5, col = "red")
> text(2.5, 0.95, "Default bandwidth", col = "darkblue",
+     pos = 4)
> text(2.5, 0.9, "Double bandwidth", col = "brown", pos = 4)
> text(2.5, 0.85, "Half bandwidth", col = "red", pos = 4)
```



Q16 : *Which bandwidths give rougher or smoother curves? What does the curve for the default bandwidth suggest about the underlying distribution?* *Jump to A16* •

4.1.3 Another histogram enhancement: colour-coding relative frequency*

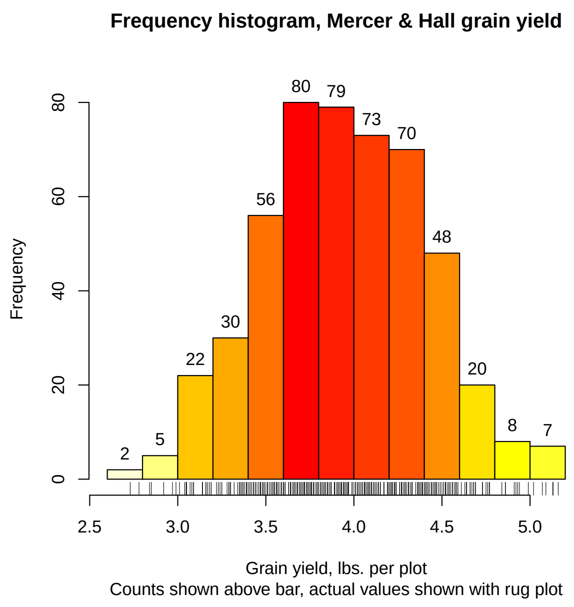
Task 15 : Display a histogram of the grain yield with break points every 0.2 lbs., with the count in each histogram bin printed on the appropriate bar. Shade the bars according to their count, in a colour ramp with low counts whiter and high counts redder. •

The solution to this task depends on the fact that the `hist` function not only plots a histogram graph, it can also **return** an object which can be **assigned** to an object in the workspace; we can then examine the object to find the counts, breakpoints etc. We first compute the histogram but don't plot it (`plot=F` argument), then draw it with the `plot` command, specifying a colour ramp, which uses the computed counts, and a title. Then the `text` command adds text to the plot at `(x, y)` positions computed from the class mid-points and counts; the `pos=3` argument puts the text on top of the bar.

```
> h <- hist(mhw$grain, breaks = seq(2.6, 5.2, by = 0.2),
+     plot = F)
> str(h)
```

```
List of 7
 $ breaks     : num [1:14] 2.6 2.8 3 3.2 3.4 3.6 3.8 4 4.2 4.4 ...
 $ counts     : int [1:13] 2 5 22 30 56 80 79 73 70 48 ...
 $ intensities: num [1:13] 0.02 0.05 0.22 0.3 0.56 ...
 $ density    : num [1:13] 0.02 0.05 0.22 0.3 0.56 ...
 $ mids       : num [1:13] 2.7 2.9 3.1 3.3 3.5 3.7 3.9 4.1 4.3 4.5 ...
 $ xname      : chr "mhw$grain"
 $ equidist   : logi TRUE
 - attr(*, "class")= chr "histogram"
```

```
> plot(h, col = heat.colors(length(h$mids))[length(h$count) -
+     rank(h$count) + 1], ylim = c(0, max(h$count) + 5),
+     main = "Frequency histogram, Mercer & Hall grain yield",
+     sub = "Counts shown above bar, actual values shown with rug plot",
+     xlab = "Grain yield, lbs. per plot")
> rug(mhw$grain)
> text(h$mids, h$count, h$count, pos = 3)
> rm(h)
```



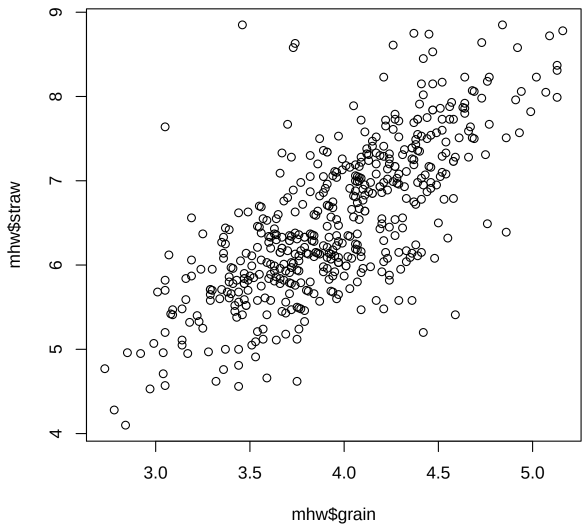
4.2 Bivariate exploratory graphics

When several variables have been collected, it is natural to compare them.

Task 16 : Display a **scatterplot** of straw vs. grain yield. •

We again use `plot`, but in this case there are two variables, so a scatterplot is produced. That is, `plot` is an example of a **generic** function: its behaviour changes according to the **class** of object it is asked to work on.

```
> plot(mhw$grain, mhw$straw)
```

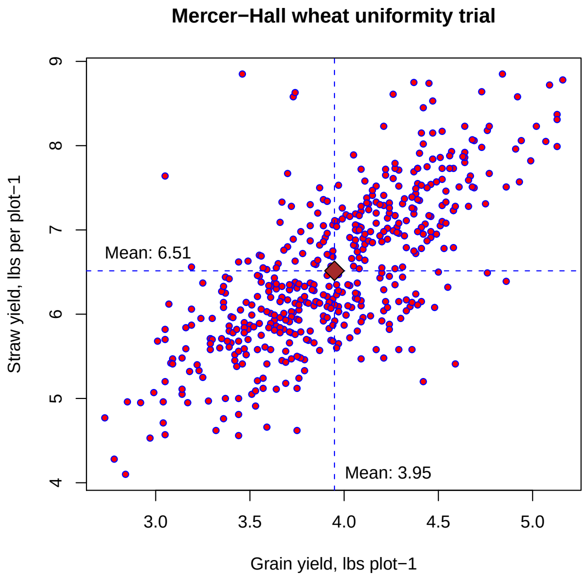


Q17 : *What is the relation between grain and straw yield?* *Jump to A17* •

Of course, this plot can be enhanced. As a simple example:

- We specify the plotting character with the `pch` graphics argument,
- its colours with the `col` (outline) and `bg` (fill) graphics arguments,
- its size with the `cex` "character expansion" graphics argument,
- the axis labels with the `xlab` and `ylab` graphics arguments;
- We add a title with the `title` function, and
- mark the centroid (centre of gravity) with two calls to `abline`, one specifying a vertical line (argument `v=`) and one horizontal (argument `vh=`) at the means of the two variables, computed with the `mean` function;
- The two lines are dashed, using the `lty` "line type" graphics argument,
- and coloured red using `col`;
- The centroid is shown as large diamond, using the `points` function and the `cex` graphics argument;
- Finally, the actual mean yields are displayed with the `text` function, using the `pos` and `adj` graphic argument to position the text with respect to the plotting position.

```
> plot(mhw$grain, mhw$straw, cex = 0.8, pch = 21, col = "blue",
+     bg = "red", xlab = "Grain yield, lbs plot-1", ylab = "Straw yield, lbs per
> title(main = "Mercer-Hall wheat uniformity trial")
> abline(v = mean(mhw$grain), lty = 2, col = "blue")
> abline(h = mean(mhw$straw), lty = 2, col = "blue")
> points(mean(mhw$grain), mean(mhw$straw), pch = 23, col = "black",
+       bg = "brown", cex = 2)
> text(mean(mhw$grain), min(mhw$straw), paste("Mean:",
+       round(mean(mhw$grain), 2)), pos = 4)
> text(min(mhw$grain), mean(mhw$straw), paste("Mean:",
+       round(mean(mhw$straw), 2)), adj = c(0, -1))
```



The advantage of this **programmed enhancement** is that we can store the commands as a script and reproduce the graph by running the script.

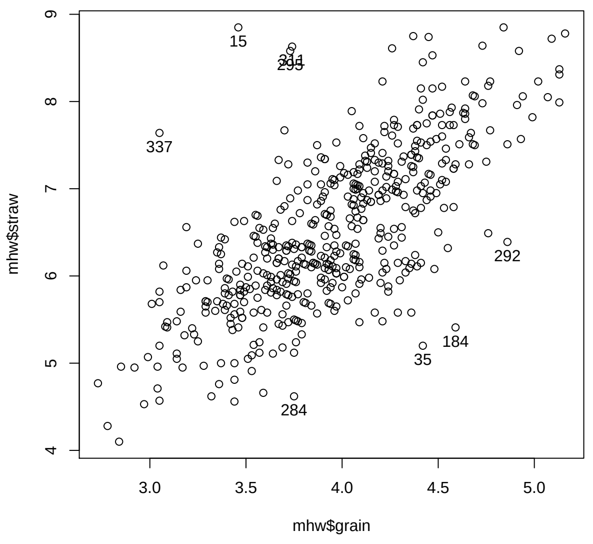
Some R graphics allow **interaction**.

Task 17 : Identify the plots which do not fit the general pattern. (In any analysis these can be the most interesting cases, requiring explanation.) •

For this we use the `identify` function, specifying the same plot coördinates as the previous `plot` command (i.e. from the plot that is currently displayed):

```
> plot(mhw$grain, mhw$straw)
> pts <- identify(mhw$grain, mhw$straw)
```

After `identify` is called, switch to the graphics window, **left-click** with the mouse on points to identify them, and **right-click** to exit. The plot should now show the row names of the selected points:



Q18 : *Which observations have grain yield that is much lower than expected (considering the straw) and which higher?* Jump to A18 •

```
> tmp <- mhw[pts, ]
> tmp[order(tmp$grain), ]

     r  c grain straw
337 17 17  3.05  7.64
15  15  1  3.46  8.85
295 15 15  3.73  8.58
311 11 16  3.74  8.63
284  4 15  3.75  4.62
35  15  2  4.42  5.20
184  4 10  4.59  5.41
292 12 15  4.86  6.39

> rm(pts, tmp)
```

4.3 Answers

A12 : 2.73 *and* 5.16 *lbs per plot, respectively. Note the placement of the decimal point, as explained in the plot header. Here it is one digit to the left of the* `|`*, so the entry* `27 | 38` *is to be read as* 2.73, 2.78. Return to Q12 •

A13 : *The stem plot shows the actual values (to some number of significant digits). This allows us to see if there is any pattern to the digits.* Return to Q13 •

A14 : *The horizontal axis is the value of the variable being summarized (in this case, grain yield). It is divided into sections ("histogram bins") whose limits are* *shown by the vertical vars. The vertical axis is the count (frequency) of observations in each bin.* *Return to Q14* •

A15 : *The two left-most histogram bins represent the values below 3 lbs per plot (horizontal axis); these appear to have 2 and 5 observations, respectively, for a total of 7; although it's difficult to estimate exactly. The stem plot, which shows the values to some precision, can show this exactly.* *Return to Q15* •

A16 : *The higher value of the* `adj` *argument to the* **`density`** *function gives a smoother curve. In this case with* `adj=2` *the curve is indistinguishable from a univariate normal distribution. The default curve is quite similar but with a slight assymetry (peak is a bit towards the smaller values) and shorter tails. But, considering the sample size, it still strongly suggests a normal distribution.* *Return to Q16* •

A17 : *They are positively associated: higher grain yields are generally associated with hihger straw yields. The relation appears to be linear across the entire range of the two measured variables. But the relation is diffuse and there are some clear excpetions.* *Return to Q17* •

A18 : *Plots 15, 337, 311 and 295 have grain yields that are lower than the general pattern; plots 308, 292, 184 and 35 the opposite.* *Return to Q18* •

5 Descriptive statistics

After visualizing the dataset, the next step is to compute some **numerical summaries**, also known as **descriptive statistics**. We can **summarize** all the variables in the dataset at the same time or individually with the `summary` function:

```
> summary(mhw)
```

```
       r               c            grain           straw
 Min.   : 1.00   Min.   : 1   Min.   :2.73   Min.   :4.10
 1st Qu.: 5.75   1st Qu.: 7   1st Qu.:3.64   1st Qu.:5.88
 Median :10.50   Median :13   Median :3.94   Median :6.36
 Mean   :10.50   Mean   :13   Mean   :3.95   Mean   :6.51
 3rd Qu.:15.25   3rd Qu.:19   3rd Qu.:4.27   3rd Qu.:7.17
 Max.   :20.00   Max.   :25   Max.   :5.16   Max.   :8.85
```

```
> summary(mhw$grain)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.73    3.64    3.94    3.95    4.27    5.16
```

Q19 : *What are the summary statistics for grain yield?* *Jump to A19* •

We can avoid the construction `frame$variable` by attaching the frame to the **search path** with the `attach` function.

```
> search()

 [1] ".GlobalEnv"        "package:gstat"     "package:sp"
 [4] "package:MASS"      "package:stats"     "package:graphics"
 [7] "package:grDevices" "package:utils"     "package:datasets"
[10] "package:lattice"   "package:methods"   "Autoloads"
[13] "package:base"

> summary(grain)

Error in summary(grain) : object "grain" not found

> attach(mhw)

> search()

 [1] ".GlobalEnv"        "mhw"               "package:gstat"
 [4] "package:sp"        "package:MASS"      "package:stats"
 [7] "package:graphics"  "package:grDevices" "package:utils"
[10] "package:datasets"  "package:lattice"   "package:methods"
[13] "Autoloads"         "package:base"

> summary(grain)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.73    3.64    3.94    3.95    4.27    5.16
```

Notice how the `mhw` object was not on the search path prior to the `attach` command.

Q20 : *What is the range in grain yields? What proportion of the median yield is this? Does this seem high or low, considering that all plots were treated the same?* *Jump to A20* •

To answer this we can use the `diff` ("difference") and `median` functions:

```
> diff(range(grain))

[1] 2.43

> diff(range(grain))/median(grain)

[1] 0.61675
```

Q21 : *Which is the lowest-yielding plot? Does it also have low straw yield?* *Jump to A21* •

To answer this, use the `which.min` function to identify the record number with the lowest yield:

```
> which.min(grain)

[1] 338

> mhw[which.min(grain), "straw"]

[1] 4.77
```

We can **select** cases based on logical criteria, for example, to find the lowest-yielding plots.

Task 18 : Find all observations with grain yield less than 3 lbs per plot, and also those with grain yield in the lowest (first) percentile. •

We can use either the `subset` function or direct matrix selection. The `quantile` function returns a list with quantiles; here we illustrate the default, the case where we use the `seq` function to ask for the ten deciles, and finally just the 1% quantile:

```
> row.names(subset(mhw, grain < 3))

[1] "149" "336" "338" "339" "441" "467" "470"

> quantile(grain)

    0%    25%    50%    75%   100%
2.7300 3.6375 3.9400 4.2700 5.1600

> quantile(grain, seq(0, 1, 0.1))

   0%   10%   20%   30%   40%   50%   60%   70%   80%   90%  100%
2.730 3.370 3.558 3.700 3.820 3.940 4.070 4.210 4.362 4.520 5.160

> mhw[grain < quantile(grain, 0.01), ]

     r  c grain straw
336 16 17  2.92  4.95
338 18 17  2.73  4.77
339 19 17  2.85  4.96
467  7 24  2.78  4.28
470 10 24  2.84  4.10
```

Q22 : *Which plots have grain yield less than 3 lbs? Which are the lowest-yielding 1%? Are those close to each other?* *Jump to A22*
•

5.1 Answers

A19 : *Minimum 2.73, maximum 5.16, arithmetic mean 3.95, first quartile 3.64, third quartile 4.27, median 3.94.* *Return to Q19* •

A20 : *The range in grain yields is 2.43, which is about 62% of the median. This seems quite high considering the "equal" treatment.* *Return to Q20* •

A21 : *The lowest-yielding plot is 338, with a grain yield of 4.77 lbs.* *Return to Q21* •

A22 : *Plots with yield less than 3 lbs are 149, 336, 338, 339, 441, 467, and 470.*

The lowest percent are plots 336, 338, 339, 467, and 470. The first three are all in field column 17 and almost adjcent field rows (16, 18, 19); this seems definitely to be a 'low-yield "hot spot" in the experimental field. The last two are both in field column 24 but a few rows apart (7 and 10). Return to Q22 •

6 Editing a data frame

If you need to fix up a few data entry errors, the data frame can be edited interactively with the `fix` function:

```
> fix(mhw)
```

In this case there is nothing to change, so just close the editor.

New variables are calculated in the local variable space. For example, the **grain-to-straw ratio** is an important indicator of how well the wheat plant has formed grain, relative to its size.

Task 19 : Compute the grain/straw ratio and summarize it. •

Note that arithmetic operations are performed on entire vectors; these are called **vectorized** operations:

```
> gsr <- grain/straw
> summary(gsr)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.391   0.574   0.604   0.611   0.642   0.850
```

Q23 : *What is the range in grain/straw ratio? Is it relatively larger or smaller than the range in grain?* Jump to A23 •

```
> range(gsr)
```

```
[1] 0.39096 0.85000
```

```
> diff(range(gsr))/median(gsr)
```

```
[1] 0.75944
```

```
> diff(range(grain))/median(grain)
```

```
[1] 0.61675
```

For further analysis we would like to include this in the data frame itself, as an additional variable.

Task 20 : Add grain-straw ratio to the `mhw` data frame and remove it from the local workspace. •

For this we use the `cbind` ("column bind") function to add a new matrix column (data frame field):

```
'data.frame':        500 obs. of  5 variables:
 $ r    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ c    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ grain: num  3.63 4.07 4.51 3.9 3.63 3.16 3.18 3.42 3.97 3.4 ...
 $ straw: num  6.37 6.24 7.05 6.91 5.93 5.59 5.32 5.52 6.03 5.66 ...
 $ gsr  : num  0.57 0.652 0.64 0.564 0.612 ...
```

```
> mhw <- cbind(mhw, gsr)
> str(mhw)
```

Note how the new matrix column took the name from the local variable.

Now we remove the local variable `gsr` so that we do not confuse it with the `gsr` field of the `mhw` data frame:

```
> ls()

[1] "gsr" "mhw"

> rm(gsr)
> ls()

[1] "mhw"
```

But now the new field is not found unless the frame is named explicitly:

```
> summary(gsr)

Error in summary(gsr) : object "gsr" not found

> summary(mhw$gsr)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.391   0.574   0.604   0.611   0.642   0.850
```

Task 21 : Detach and re-attach the `mhw` object, so that the field `gsr` can be found. •

```
> detach(mhw)
> attach(mhw)

> summary(gsr)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.391   0.574   0.604   0.611   0.642   0.850
```

Task 22 : Save the updated `mhw` object in R format. •

We use a different file name to distinguish this from the original file, without the added column.

```
> save(mhw, file = "mhw2.RData")
```

6.1 Answers

A23 : *The range is from 0.39 to 0.85, i.e. the ratio of grain to straw doubles. This is about 76% of the median ratio, which is considerably higher than the comprable figure for grain yield (about 62%).* *Return to Q23* •

7 Introduction to modelling

After descriptive statistics and visualisation comes the attempt to build **statistical models** of the underlying processes. These are **empirical** mathematical relations that describe one variable in terms of some hypothetical underlying distribution of which it is a realization, or describe several variables either as equals ("correlation") or where one is described in terms of others ("regression"). We explore these, from simple to complex.

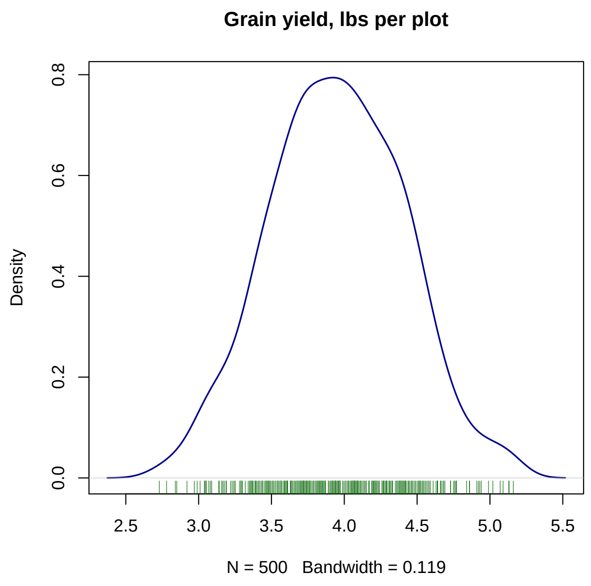
8 Univariate modelling

We suppose that the observed sample distribution is from an underlying probability distribution. This raises two questions: (1) what is the form of that distribution, and (2) what are its parameters?

To decide what **theoretical** distribution might fit, we first visualise the **empirical** distribution. This continues the ideas from §4.1.2.

Task 23 : Visualise an empirical continuous frequency distribution on the rug plot. •

```
> plot(density(mhw$grain), col = "darkblue", main = "Grain yield, lbs per plot",
+     lwd = 1.5)
> rug(mhw$grain, col = "darkgreen")
```

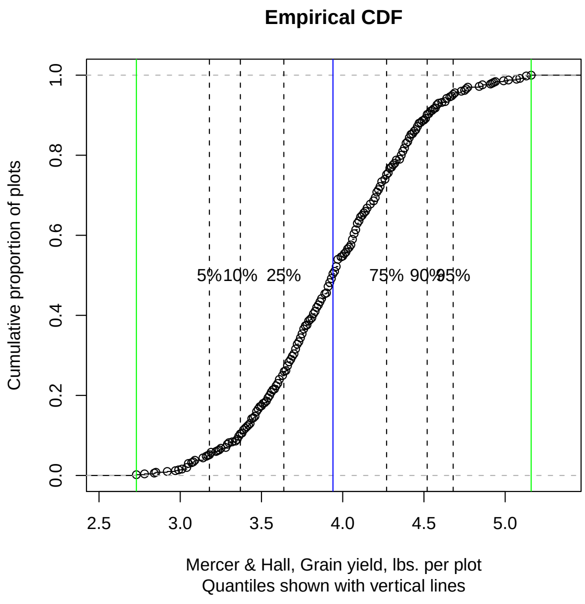


We can also view the distribution as a **cumulative** rather than **density** distribution.

Task 24 : Visualise the empirical **cumulative distribution** of grain yield. •

We use the `ecdf` ("**e**mpirical **c**umulative **d**istribution **f**unction") function to compute the distribution, then plot it with the `plot` function. Vertical lines are added to the plot with the `abline` ("add a straight line") function, at the median, extremes, and specified quantiles.

```
> plot(ecdf(grain), pch = 1, xlab = "Mercer & Hall, Grain yield, lbs. per plot",
+     ylab = "Cumulative proportion of plots", main = "Empirical CDF",
+     sub = "Quantiles shown with vertical lines")
> q <- quantile(grain, c(0.05, 0.1, 0.25, 0.75, 0.9, 0.95))
> abline(v = q, lty = 2)
> abline(v = median(grain), col = "blue")
> abline(v = max(grain), col = "green")
> abline(v = min(grain), col = "green")
> text(q, 0.5, names(q))
> rm(q)
```

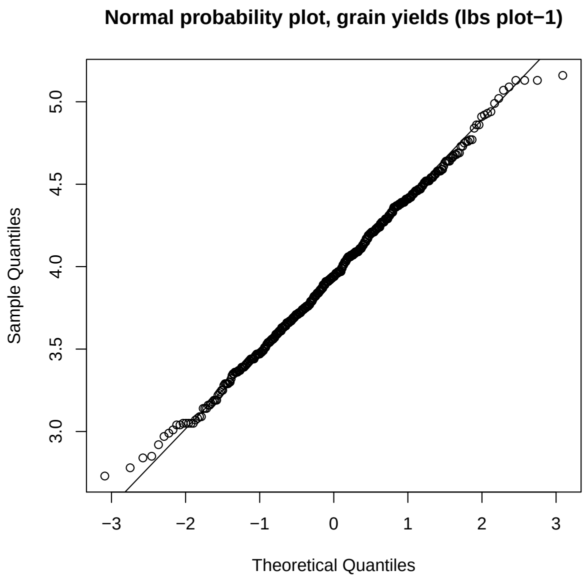


Q24 : *From the histogram, stem-and-leaf plot, empirical cumulative distribution, and theory, what probability distribution is indicated for the grain yield?* *Jump to A24* •

We can also visualize the distribution against the theoretical normal distribution computed with the sample mean and variance. There are (at least) two useful ways to visualize this.

First, compare the actual and theoretical normal distribution is with the `qqnorm` function to plot these against each other and then superimpose the theoretical line with the `qqline` function:

```
> qqnorm(grain, main = "Normal probability plot, grain yields (lbs plot-1)")
> qqline(grain)
```



Q25 : *Does this change your opinion of normality?* *Jump to A25* •

The second way to visually compare an empirical and theoretical distribution is to display the empirical density plot, superimposing the normal distribution that would be expected with the sample mean and standard deviation.

Task 25 : Fit a normal probability distribution to the empirical distribution of grain yield. •

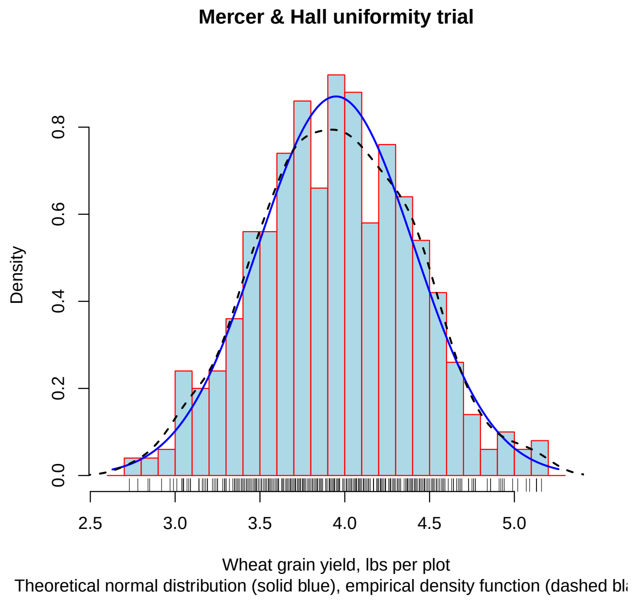
Q26 : *What are the best estimates of the parameters of a normal distribution for the grain yield?* *Jump to A26*
•

These are computed with the `mean` and `sd` functions:

```
> mean(grain)

[1] 3.9486

> sd(grain)

[1] 0.45828
```

With these in hand, we can plot the theoretical distribution against the empirical distribution:

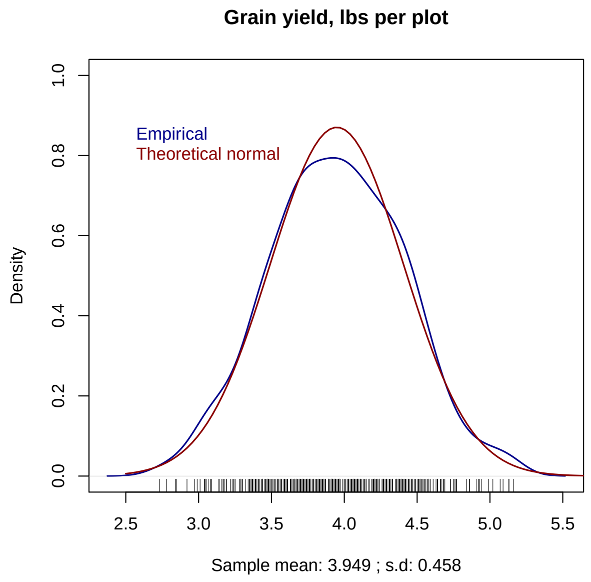
Task 26 : Graphically compare the theoretical and empirical distributions.



```
> res <- 0.1
> hist(grain, breaks=seq(round(min(grain),1)-res,
+      round(max(grain),1)+res, by=res), col="lightblue", border="red",
+      freq=F,
+      xlab="Wheat grain yield, lbs per plot",
+      main="Mercer & Hall uniformity trial",
+      sub="Theoretical normal distribution (solid blue),
+           empirical density function (dashed black)")
> rug(grain)
> x <- seq(min(grain)-res, max(grain)+res, by=.01)
> lines(x, dnorm(x, mean(grain), sd(grain)), col="blue", lty=1, lwd=1.8)
> lines(density(grain), lty=2, lwd=1.8, col="black")
> rm(res, x)
```

We can also see this on the empirical density. This version also uses the `curve` method to draw the theoretical curve.

```
> plot(density(mhw$grain), col = "darkblue", main = "Grain yield, lbs per plot",
+     lwd = 1.5, ylim = c(0, 1), xlab = paste("Sample mean:",
+         round(mean(mhw$grain), 3), "; s.d:", round(sd(mhw$grain),
+             3)))
> rug(mhw$grain)
> curve(dnorm(x, mean(mhw$grain), sd(mhw$grain)), 2.5,
+     6, add = T, col = "darkred", lwd = 1.5)
> text(2.5, 0.85, "Empirical", col = "darkblue", pos = 4)
> text(2.5, 0.8, "Theoretical normal", col = "darkred",
+     pos = 4)
```



There are several tests of normality; here we use the Shapiro-Wilk test, implemented by the `shapiro.test` function. This compares the empirical distribution (from the **sample**) with the theoretical distribution, and computes a statistic ("W") for which is known the probability that it could occur by change, **assuming** the sample is really from a normally-distributed **population**. The reported probability value is the chance that **rejecting the null hypothesis** H_0 that the sample *is* from a normal population is an **incorrect decision** (i.e. the probability of committing a Type I error).

```
> shapiro.test(grain)

        Shapiro-Wilk normality test

data:  grain
W = 0.997, p-value = 0.486
```

Q27 : *According to the Shapiro-Wilk test, what is the probability if we reject the null hypothesis that this empirical distribution is a realization of a normal distribution with the sample mean and standard deviation as parameters, we would be wrong (i.e. comit a Type I error)? So, should we reject the null hypothesis of normality? Should we consider grain yield to be a normally-distributed variable?* *Jump to A27* •

Once we've established that the yields can be reasonably modelled by a normal distribution, we can compute **confidence limits** on the mean yield. Mercer and Hall used the 50% confidence level, which they called the **probable error**: there is equal chance for the true yield to be inside as outside this interval.

Task 27 : Compute the probable error for grain yield in a 1/500 acre plot. •

Although we've fitted a normal distribution, both the mean and standard deviation were estimated from the sample. So, we should use Student's t-distribution (although with so many plots the difference with the normal z-distribution will be very small).

The probable error is computed from the limits of the interquartile range of the distribution; we get the quantiles of the t distribution with the `qt` function, specifying the degrees of freedom:

```
> (t.q13 <- qt(c(0.25, 0.75), length(mhw$grain) - 1))

[1] -0.67498  0.67498
```

This is for $\mu = 0, \sigma = 1$. We then scale this to the data:

```
> (pe <- mean(mhw$grain) + t.q13 * sd(mhw$grain))

[1] 3.6393 4.2580
```

And we can express it as a relative error; conventionally this is expressed as error ± relative to the mean:

```
> rel.error <- (diff(pe)/2)/mean(mhw$grain)
> round(100 * rel.error, 2)

[1] 7.83
```

Q28 : *If the true yield over the entire field is the observed mean yield, what yields can be expected in any one 1/500 acre plot, with 50% confidence that the mean is in this range?* *Jump to A28* •

Q29 : *What is the probable error expressed as a percentage of mean yield?* *Jump to A29* •

To put this in practical terms, since this was Mercer and Hall's main concern, we calculate the probable error in absolute kg ha^{-1}.

We saw above that the mean grain yield in this experiment was 3.95 lbs plot^{-1}. Scaling this up to a hectare basis (which is how yields are normally expressed):

```
> (yield.ha <- mean(mhw$grain) * 500/(0.40469)/(2.2046226))

[1] 2212.9
```

The two constants in this formula are `0.40469 ha acre`$^{-1}$ and `2.2046226 lbs kg`$^{-1}$, and there are 500 plots per acre.

Q30 : *If we try to estimate the yield in kg ha^{-1} from a single 1/500 acre plot in a variety trial, how much error (in absolute terms) can we expect, with 50% probabilty? What are the practical implications of this?* *Jump to A30* •

```
> round(yield.ha * rel.error, 2)

[1] 173.35
```

Clean up:

```
> rm(t.q13, pe, rel.error, yield.ha)
```

8.1 Answers

A24 : *The Normal distribution. From theory: the addition of multiple indepedent sources of noise. From the plots: visual fit to simulated normal distributions.* *Return to Q24* •

A25 : *The distribution still looks normal, except at both tails: the highest yields are not as high, and the lowest not as low, as expected if the yields were normally distributed.* *Return to Q25* •

A26 : *Parameters* $\mu = 3.95, \sigma = 0.458$. *Return to Q26* •

A27 : *According to the Shapiro-Wilk test of normality, the probability that rejecting the* ***null hypothesis*** *of normality would be an* ***incorrect*** *decision (i.e. a* ***Type I error*** *is 0.49; this is quite high, so we* ***do not reject*** *the null hypothesis. So, we consider the grain yield to be a realisation of a normal distribution.* *Return to Q27* •

A28 : *From* 3.64 *to* 4.36 *lbs plot*-1*; the observed mean yield is* 3.95 *lbs plot*-1*.* *Return to Q28* •

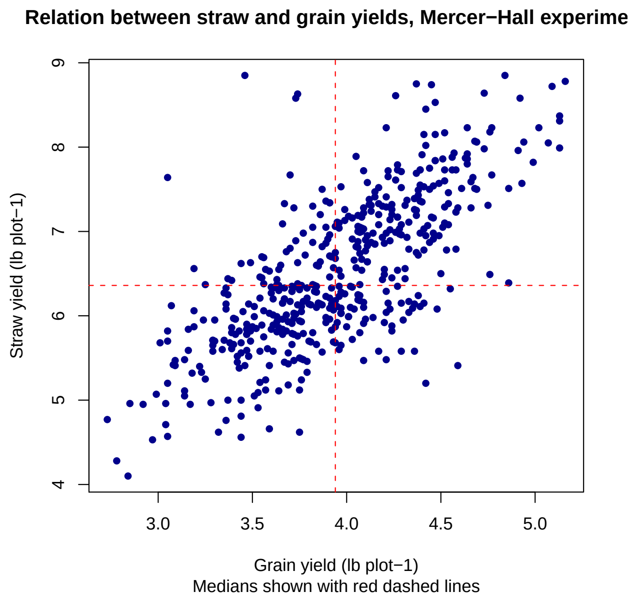
A29 : *± 7.83%.* *Return to Q29* •

A30 : *From a true yield (estimated here from our mean) of 2213 kg wheat grain, the probable error is 173 kg, a substantial amount. There is a 50% chance of observing this much deviation if we estimate the per-hectare yield from any single 1/500 acre plot. Clearly, this is where the idea of* ***replicated*** *plots originated.* *Return to Q30* •

9 Bivariate modelling: continuous variables

In §4.2 we displayed a scatterplot of straw vs. grain yield, repeated here with some additional graphical elements:

```
> plot(mhw$straw ~ mhw$grain, xlab = "Grain yield (lb plot-1)",
+     ylab = "Straw yield (lb plot-1)", pch = 20, col = "darkblue",
+     cex = 1.2, sub = "Medians shown with red dashed lines")
> title(main = "Relation between straw and grain yields, Mercer-Hall experiment"
> abline(v = median(mhw$grain), lty = 2, col = "red")
> abline(h = median(mhw$straw), lty = 2, col = "red")
```



Note the use of the ~ formula operator to indicate dependence, here of straw on grain; we will explore this further just below. The centroid is shown by intersecting horizontal and vertical lines, added with the `abline` function.

Q31 : *From the scatterplot of straw vs. grain yield, what functional relation is suggested?* *Jump to A31* •

For a **bivariate linear** relation, as hypothesized here, we can view this four ways:

1. A bivariate **linear correlation** between the two variables (straw and grain yields);
2. A univariate **linear regression** of straw (dependent) on grain (independent) yield;
3. A univariate **linear regression** of grain (dependent) on straw (independent) yield.
4. A **linear structural relation** between the two yields.

These will each be explored below.

Bivariate correlation, bivariate structural relations and univariate regression all compare two variables that refer to *the same observations*, that is, they are *paired*. This is the natural order in a data frame: each *row* represents *one observation* on which several *variables* were measured; in the present case, the row and column numbers of the plots, and the grain and straw yields.

Correlation and various kinds of regression are often misused. There are several good journal articles that explain the situation, with examples from earth science applications [12, 32]. A particularly understandable introduction to the proper use of gression is by Webster [33].

9.1 Correlation

Correlation measures the strength of the association between two variables measured on the same object, from -1 (perfect negative correlation), through 0 (no correlation), to $+1$ (perfect positive correlation). The two variables have logically equal status, so there is no concept of **causation**; in addition, there is no functional relation, so there is no way to **predict**.

There are several numerical measures of correlation; we will see two:

1. Pearson's product moment correlation coefficient (PMCC) r;
2. Spearman's ρ.

Parametric correlation The PMCC should only be used if the two variables are distributed approximately **bivariate normally**. This is two normal distributions, but with some degree of correlation. So we first check whether the relation between grain and straw yield has this distribution.

Task 28 : Visualize a bivariate normal distribution with the parameters of the grain and straw yields; visually compare with the actual bivariate distribution. •

R has a `rnorm` function to simulate a random sample of a given size from the normal distribution, by analogy to the `runif` function presented above. However, to simulate a **correlated** sample of several variables, we turn to the `mvrnorm` function in the `MASS` ("Modern Applied Statistics with S") package which corresponds to the very useful advanced text of Venables and Ripley [31]. This function uses a vector of the variable means, along with the **variance-covariance** matrix of two or more variables.

The variable means are computed with the vectorized `mean` function:

```
> mean(mhw[, c("grain", "straw")])
```

```
 grain  straw
3.9486 6.5148
```

The variance-covariance matrix is computed with the vectorized `var` function:

```
> var(mhw[, c("grain", "straw")])
```

```
        grain   straw
grain 0.21002 0.30043
straw 0.30043 0.80696
```

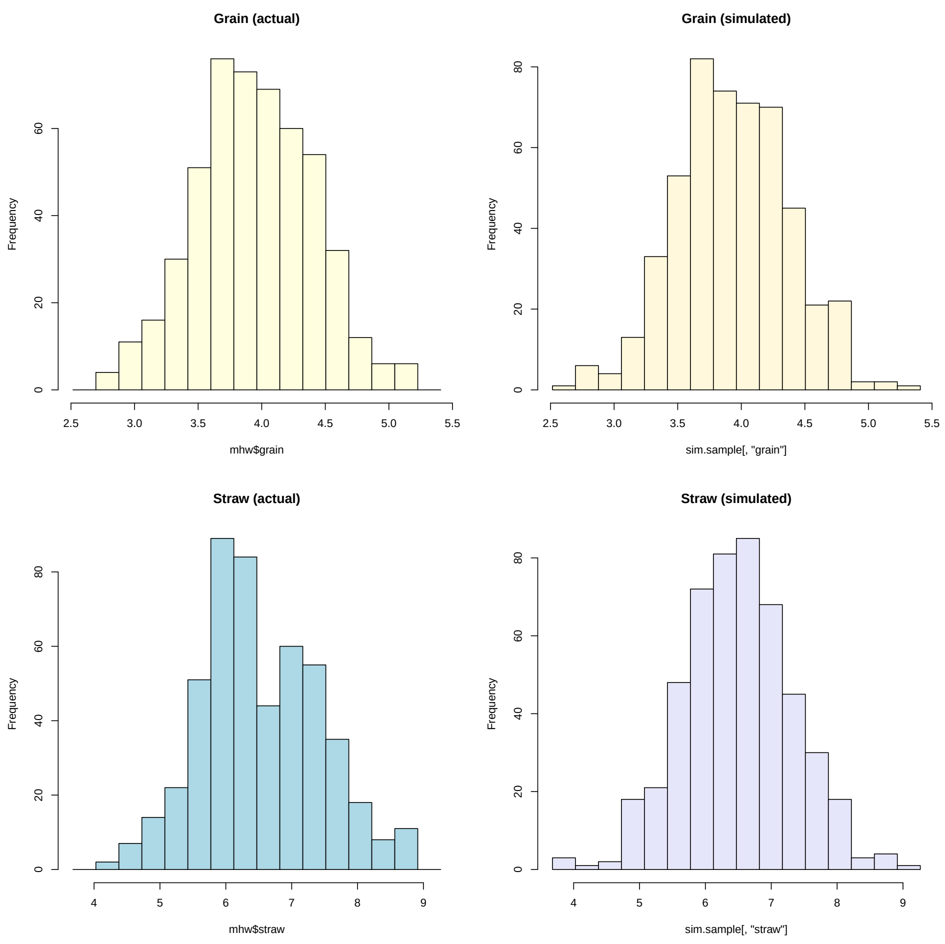
Q32 : *What do the diagonals and off-diagonals of this matrix represent? What are their units of measure?* *Jump to A32* •

From these the mvrnorm can draw a simulated random sample:

```
> library(MASS)
> sim.sample <- mvrnorm(length(mhw$grain), mu = mean(mhw[,
+     c("grain", "straw")]), Sigma = var(mhw[, c("grain",
+     "straw")]))
> head(sim.sample)

      grain  straw
[1,] 4.4978 7.8577
[2,] 3.9456 5.6364
[3,] 3.3726 6.4418
[4,] 3.7070 5.8997
[5,] 4.1051 7.2715
[6,] 3.2972 6.0774

> summary(sim.sample)

     grain          straw     
 Min.   :2.52   Min.   :3.68  
 1st Qu.:3.64   1st Qu.:5.92  
 Median :3.93   Median :6.49  
 Mean   :3.93   Mean   :6.49  
 3rd Qu.:4.24   3rd Qu.:7.04  
 Max.   :5.41   Max.   :9.26  

> summary(mhw[, c("grain", "straw")])

     grain          straw     
 Min.   :2.73   Min.   :4.10  
 1st Qu.:3.64   1st Qu.:5.88  
 Median :3.94   Median :6.36  
 Mean   :3.95   Mean   :6.51  
 3rd Qu.:4.27   3rd Qu.:7.17  
 Max.   :5.16   Max.   :8.85  
```

We can also visualize these distributions with histograms: To ensure that the visualization is comparable, we compute the overall minimum and maximum of both variables and use these to explictly set the axis limits.

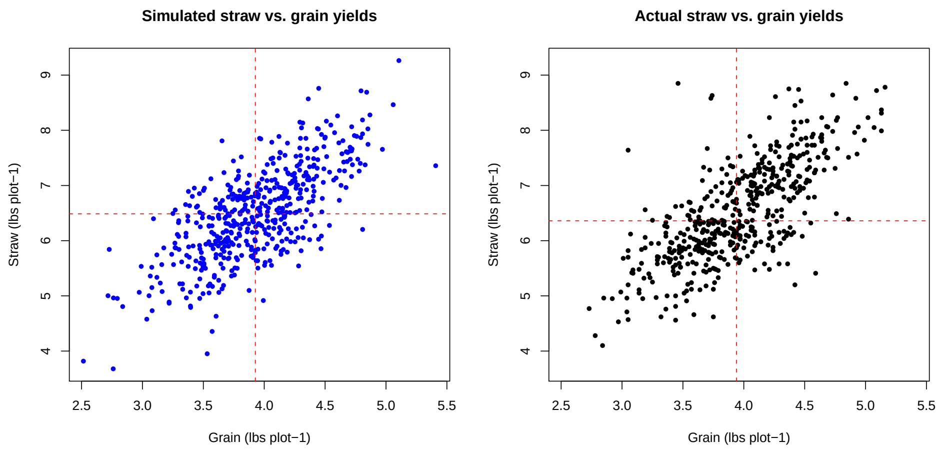
```
> par(mfrow = c(2, 2))
> grain.lim = c(min(sim.sample[, "grain"], mhw$grain),
+     max(sim.sample[, "grain"], mhw$grain))
> straw.lim = c(min(sim.sample[, "straw"], mhw$straw),
+     max(sim.sample[, "straw"], mhw$straw))
> hist(mhw$grain, xlim = grain.lim, main = "Grain (actual)",
+     col = "lightyellow", breaks = seq(grain.lim[1], grain.lim[2],
+         length = 17))
> hist(sim.sample[, "grain"], xlim = grain.lim, main = "Grain (simulated)",
+     col = "cornsilk", breaks = seq(grain.lim[1], grain.lim[2],
+         length = 17))
> hist(mhw$straw, xlim = straw.lim, main = "Straw (actual)",
+     col = "lightblue", breaks = seq(straw.lim[1], straw.lim[2],
+         length = 17))
> hist(sim.sample[, "straw"], xlim = straw.lim, main = "Straw (simulated)",
+     col = "lavender", breaks = seq(straw.lim[1], straw.lim[2],
+         length = 17))
> par(mfrow = c(1, 1))
```



Q33 : *How well do the two univariate simulations match the actual data?* *Jump to A33* •

So we can simulate a sample with the same mean and standard deviation as the grain and straw yields, and plot them against each other in a scatterplot. We display this side-by-side with the actual scatterplot.

```
> par(mfrow = c(1, 2))
> plot(sim.sample, main = "Simulated straw vs. grain yields",
+     xlab = "Grain (lbs plot-1)", ylab = "Straw (lbs plot-1)",
+     xlim = grain.lim, ylim = straw.lim, pch = 20, col = "blue")
> abline(v = median(sim.sample[, 1]), lty = 2, col = "red")
> abline(h = median(sim.sample[, 2]), lty = 2, col = "red")
> plot(mhw$grain, mhw$straw, main = "Actual straw vs. grain yields",
+     xlab = "Grain (lbs plot-1)", ylab = "Straw (lbs plot-1)",
+     xlim = grain.lim, ylim = straw.lim, pch = 20, col = "black")
> abline(v = median(mhw$grain), lty = 2, col = "red")
> abline(h = median(mhw$straw), lty = 2, col = "red")
> par(mfrow = c(1, 1))
```



Q34 : *Do the two relations (simulated vs. actual) look similar? Do they support the hypothesis of a bivariate linear relation?* *Jump to A34* •

We delete the temporary variables used in this visualization:

```
> rm(sim.sample, grain.lim, straw.lim)
```

Challenge: The single simulation is only one realization of the hypothetical process. Repeat the simulation several times and compare (1) the simulations with each other; (2) the simulations with the actual data.

With this evidence of bivariate normality, we are justified in computing the parametric correlation.

Task 29 : Compute the PMCC between grain and straw yield, and its 95% confidence limit. •

The `cor` function computes the correlation; the `cor.test` function also computes the confidence interval:

```
> cor(mhw$grain, mhw$straw)

[1] 0.72978

> cor.test(mhw$grain, mhw$straw)

        Pearson's product-moment correlation

data:  mhw$grain and mhw$straw
t = 23.821, df = 498, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.68599 0.76830
sample estimates:
    cor
0.72978
```

Q35 : *What are the lowest possible, most probable, and highest possible values of the correlation coefficient r, with 95% confidence? Do you consider this a very strong, strong, moderately strong, weak, or no correlation? Positive or negative?* *Jump to A35* •

Non-parametric correlation In case the relation can not be assumed to be bivariate normal, methods must be used that do not depend on this assumption. These are called **non-parametric**; for correlation a widely-used metric is Spearman's ρ, which is the PPMC between the **ranks** of the observations.

Task 30 : Compare the ranks of the first ten plots in the data frame. •

The `rank` function returns the ranks of the values in a vector, from low to high. Ties can be handled in several ways; the default is to average the ranks.

We display the data values and their ranks in a table:

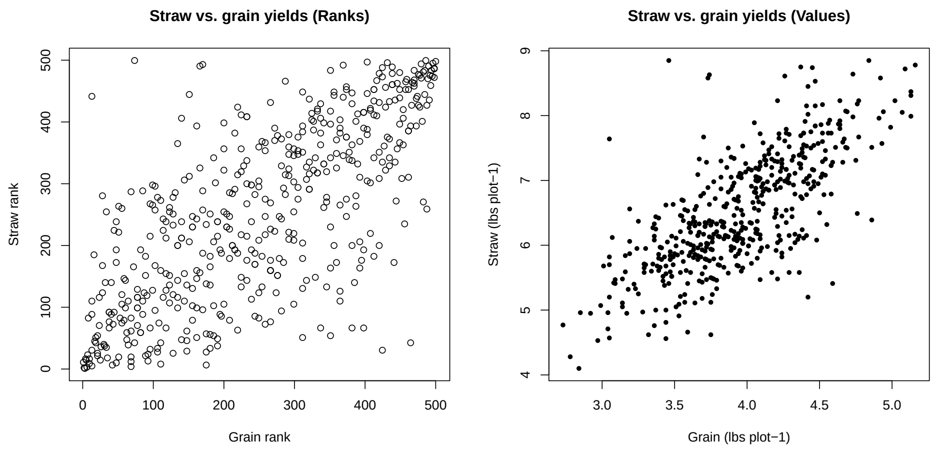
```
> head(cbind(mhw[, c("grain", "straw")], rank(mhw$grain),
+     rank(mhw$straw)), n = 10)
```

```
   grain straw rank(mhw$grain) rank(mhw$straw)
1   3.63  6.37           123.0           254.5
2   4.07  6.24           299.0           219.5
3   4.51  7.05           445.5           356.5
4   3.90  6.91           228.0           329.0
5   3.63  5.93           123.0           136.0
6   3.16  5.59            23.5            70.5
7   3.18  5.32            26.0            36.0
8   3.42  5.52            62.5            59.0
9   3.97  6.03           266.0           159.5
10  3.40  5.66            58.5            79.0
```

Q36 : *What are the data values and ranks of the first plot? Do the ranks of the two variables generally match?* *Jump to A36* •

Task 31 : Display a scatterplot of the **ranks**, alongside a scatterplot of the **values**. •

```
> par(mfrow = c(1, 2))
> plot(rank(mhw$grain), rank(mhw$straw), main = "Straw vs. grain yields (Ranks)",
+     xlab = "Grain rank", ylab = "Straw rank", pch = 1)
> plot(mhw$grain, mhw$straw, main = "Straw vs. grain yields (Values)",
+     xlab = "Grain (lbs plot-1)", ylab = "Straw (lbs plot-1)",
+     pch = 20)
> par(mfrow = c(1, 1))
```



Q37 : *How similar are the scatterplots?* *Jump to A37* •

Task 32 : Compute the numerical value of the Spearman's correlation between grain and straw yield, and compare it to the PPMC. •

The `cor` and `cor.test` functions have an optional `method=` argument; this defaults to `pearson` but can be set explicitly:

```
> cor(mhw$grain, mhw$straw, method = "pearson")

[1] 0.72978

> cor(mhw$grain, mhw$straw, method = "spearman")

[1] 0.71962
```

In this example, the two values are similar; for two variables that are not approximately bivariate normal, there can be major differences.

9.2 Univariate linear regression

Univariate **linear regression** differs from bivariate correlation in that one of the two variables is considered mathematically **dependent** on the other.

An important distinction is made between predictors which are known without error, whether fixed by the experimenter or measured, and those that are not.

Fixed effects model Webster [33] calls the first type a "Gauss linear model"[6] because only the predictand has Gaussian or "normal" error, whereas the predictor is known without error. The regression goes in one direction only,

[6] referring to the developer of least-squares regression

from the mathematical predictor to the random response, and is modelled by a **linear model with error in the response**:

$$y_i = BX_i + \varepsilon_i$$

of which the simplest case is a line with intercept:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

In this model there is no error associated with the predictors x_i, only with the predictand y_i. The predictors are known without error, or at least the error is quite small in comparison to the error in the model. An example is a designed agricultural experiment where the quantity of fertiliser added (the predictor) is specified by the design and the crop yield is measured (the predictand); there is random error ε_i in this response.

Random effects model In the present example both variables (grain and straw yields) were measured with error; they were not imposed by the experimenter. Both variables should have Gaussian error, with some correlation. This is modelled as a **bivariate normal distribution** of two random variables, X and Y with (unknown) population means μ_X and μ_Y, (unknown) population variances σ_X and σ_Y, and an (unknown) correlation ρ_{XY} which is computed as the standardised (unknown) covariance $\mathrm{Cov}(X,Y)$:

$$\begin{array}{rcl} X & \sim & \mathcal{N}(\mu_X, \sigma_X) \\ Y & \sim & \mathcal{N}(\mu_Y, \sigma_Y) \\ \rho_{XY} & = & \mathrm{Cov}(X,Y)/\sigma_X\sigma_Y \end{array}$$

This is exactly what was modelled in the previous section.

Note: In practice, the distinction between the two models is not always clear. The predictor, even if specified by the experimenter, can also have some measurement error. In the fertiliser experiment, even though we specify the amount per plot, there is error in measuring, transporting, and spreading it. In that sense it can be considered a random variable. But, since we have some control over it, the experimental error can be limited by careful procedures. We can not limit the error in the response by the same techniques.

9.2.1 Fitting a regression line

When we decide to consider one of the variables as as a response and the other as a predictor, we attempt to fit a line that best describes this relation. There are three types of lines we can fit, usually in this order:

1. Exploratory, non-parametric
2. Parametric
3. Robust

The first kind just gives a “smooth” impression of the relation. The second fits according to some optimality criterion; the classic *least-squares* estimate is in this class. The third is also parametric but optimises some criterion that protects against a few unusual data values in favour of the majority of the data.

In the present case, our analysis above provides evidence that the relation is indeed well-modelled by a bivariate normal distribution, so that a parametric approach can be taken. We will then examine regression diagnostics to see if a robust regression is needed.

Q38 : *Is there any reason to choose grain or straw as the dependent variable (predictand), and other as the independent (predictor)?* *Jump to A38* •

Task 33 : Fit a least-squares prediction line of straw as predicted by grain; display the model summary. •

Q39 : *What is the purpose of fitting this relation?* *Jump to A39* •

For this we use the **linear models** function `lm`; this is the workhorse of modelling in R. The generic `summary` function now produces a model summary; the `coefficients` **access function** extracts the regression line coefficients from the model object:

```
> model.straw.grain <- lm(straw ~ grain)
> summary(model.straw.grain)

Call:
lm(formula = straw ~ grain)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.0223 -0.3529  0.0104  0.3734  3.0342 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   0.8663     0.2387    3.63  0.00031 ***
grain         1.4305     0.0601   23.82  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.615 on 498 degrees of freedom
Multiple R-squared: 0.533,	Adjusted R-squared: 0.532 
F-statistic:  567 on 1 and 498 DF,  p-value: <2e-16 

> coefficients(model.straw.grain)

(Intercept)       grain 
    0.86628     1.43050 
```

This example illustrates the simplest **S model formula**, here `straw ~ grain`, which is signalled by the `~` operator, which can be read "is modelled by". So here the formula `straw ~ grain` can be read "straw yield is modelled by grain yield".

Q40 : *What is the linear least-squares relation between straw and grain yields?* *Jump to A40* •

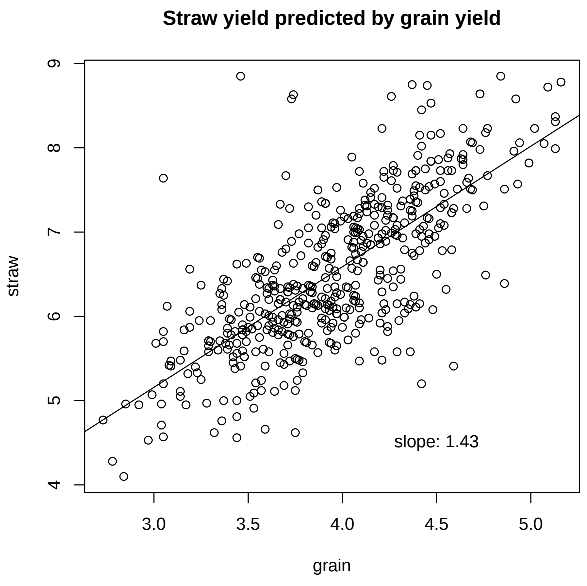
Q41 : *How much straw yield is expected if there is no grain yield? Does this seem realistic?* *Jump to A41* •

Q42 : *How much does the straw yield increase for each extra lb of grain yield?* *Jump to A42* •

Q43 : *How much of the variability in straw yield is explained by this relation?* *Jump to A43* •

Task 34 : Display the scatterplot with the best-fit line superimposed. •

```
> plot(straw ~ grain)
> title("Straw yield predicted by grain yield")
> abline(model.straw.grain)
> text(4.5, 4.5, paste("slope:", round(coefficients(model.straw.grain)[2],
+     2)))
```



Note the use of the `abline` function to add a line to the plot, the `text` function to place text on it, and the `title` function to add a title. Also note the use of the model formula syntax `straw ~ grain`. This is to be read "straw depends on grain".

9.2.2 Regression diagnostics

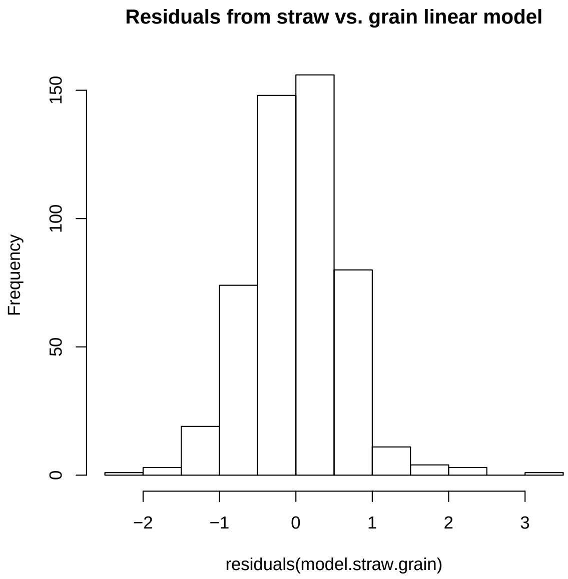
Of course, we have to treat any model with suspicion; for linear models there are some standard diagnostics. In particular, the hypothesis for the linear model is that the response is some deterministic linear function of the predictor, plus a **normally-distributed random error**:

$$y = \beta_0 + \beta_1 x + \epsilon$$

We will investigate whether the model we have just fit satisfies this criterion.

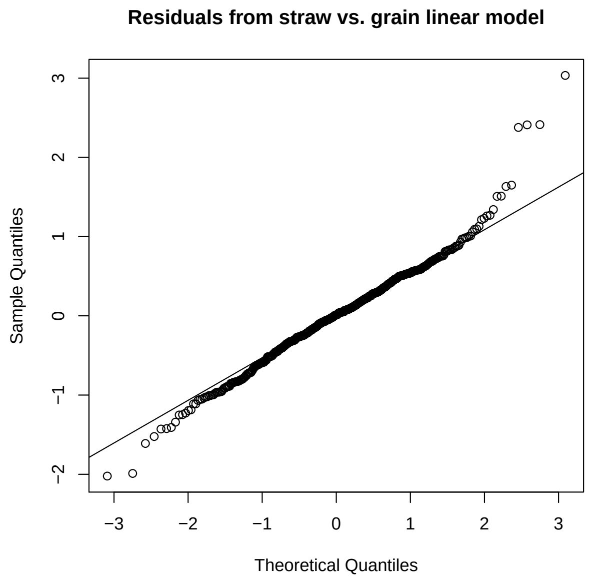
Task 35 : Display a histogram and quantile-quantile plot of the **regression residuals**; summarize these numerically. •

```
> hist(residuals(model.straw.grain), main = "Residuals from straw vs. grain linea
```



```
> qqnorm(residuals(model.straw.grain), main = "Residuals from straw vs. grain lin
> qqline(residuals(model.straw.grain))
> summary(residuals(model.straw.grain))
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
-2.02e+00 -3.53e-01  1.04e-02  1.76e-16  3.73e-01  3.03e+00 
```



Q44 : *Do the residuals appear to be normally-distributed? Are they at least symmetrically-distributed?* Jump to A44 •

We can test the normality of the residuals with a Shapiro-Wilks test:

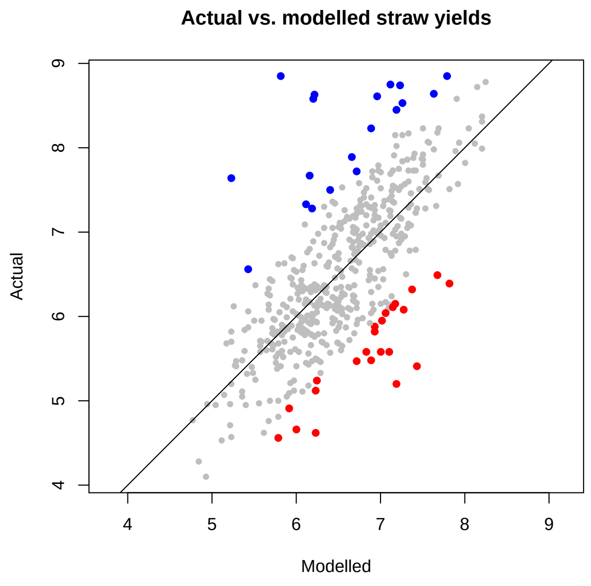
```
> shapiro.test(residuals(model.straw.grain))

        Shapiro-Wilk normality test

data:  residuals(model.straw.grain)
W = 0.9767, p-value = 3.631e-07
```

Q45 : *According to the Shapiro-Wilk test, should we reject the null hypothesis of normality for the residuals?* Jump to A45
•

Task 36 : Compare the fitted to actual values along a 1:1 line; highlight those that are more than 1 lb. per plot in error. •

To colour points we use the `col` optional argument to the `plot` function, and the `ifelse` function to select a colour based on a **logical condition**:

```
> lim <- c(min(fitted(model.straw.grain), mhw$straw), max(fitted(model.straw.grai
+     mhw$straw))
> plot(fitted(model.straw.grain), mhw$straw, xlab = "Modelled",
+     ylab = "Actual", asp = 1, xlim = lim, ylim = lim,
+     pch = 20, col = ifelse((abs(fitted(model.straw.grain) -
+         mhw$straw) < 1), "gray", ifelse(fitted(model.straw.grain) <
+         mhw$straw, "blue", "red")), cex = ifelse((abs(fitted(model.straw.grain)
+         mhw$straw) < 1), 1, 1.3))
> title("Actual vs. modelled straw yields")
> abline(0, 1)
> rm(lim)
```



Task 37 : Some of the plots are poorly-fit by this relation; identify those with an absolute residual > 1.8 lbs straw and display their records, sorted by grain/straw ratio. •

```
> high.res <- mhw[which(abs(residuals(model.straw.grain)) >
+     1.8), ]
> high.res[order(high.res$gsr), ]

     r  c grain straw     gsr in.north
15  15  1  3.46  8.85 0.39096    FALSE
337 17 17  3.05  7.64 0.39921    FALSE
311 11 16  3.74  8.63 0.43337    FALSE
295 15 15  3.73  8.58 0.43473    FALSE
184  4 10  4.59  5.41 0.84843     TRUE
35  15  2  4.42  5.20 0.85000    FALSE

> rm(high.res)
```

This example illustrates the `abs` ("absolute value") and `order` functions.

Q46 : *Which plots have absolute residuals > 1.8 lbs straw? Which are too high and which too low, according to the relation?* *Jump to A46* •

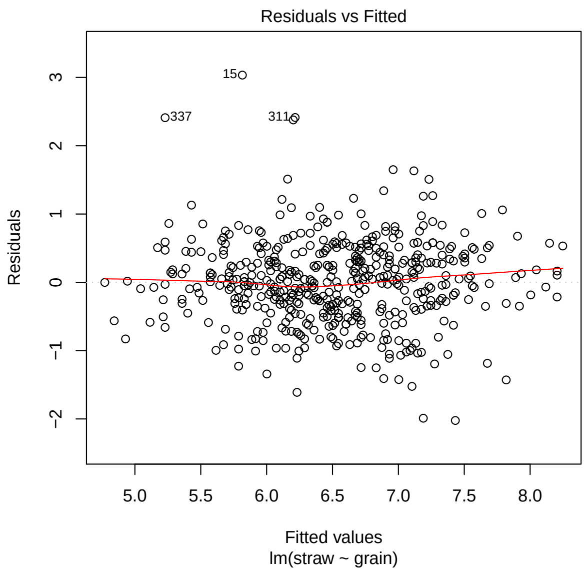
Challenge: Repeat the analysis, but reversing the variables: model grain yield as a function of straw yield. Are the slopes inverses of each other? Why or why not?

Further diagnostics The normality of residuals is one requirement for linear modelling; however there are other issues.

The generic `plot` function applied to a model result (i.e. object returned from a call to `lm`) gives a standard set of diagnostic plots, selectable with the `which=` argument.

Plot type 1 is a plot of residuals vs. fitted values; there should be no relation between these. That is, whether a residual is high or low, positive or negative should not depend on the value fitted by the model. There should not be any trend, and there should be no difference in spread (range from largest positive to negative) according to fitted value.

```
> plot(model.straw.grain, which = 1)
```



Plot type 2 is the Q-Q plot as we studied in the previous section.

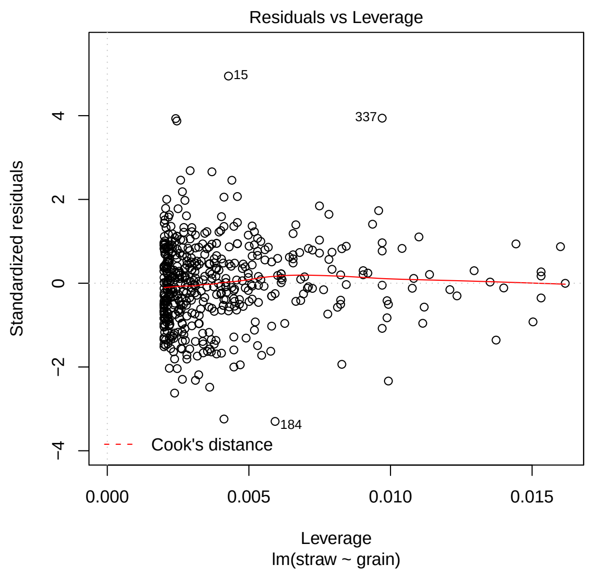
Plot type 5 shows the **leverage** of each observation and its corresponding residual.

Observations that are far from the centroid of the regression line can have a large effect on the estimated slope; they are said to have high *leverage*, by analogy with a physical lever. They are not necessarily in error, but they should be identified and verified; in particular, it is instructive to compare the estimated regression line with and without the high-leverage observations.

The leverage is measured by the *hat value*, which measures the overall influence of a single observation on the predictions. In diagnostic plot type 5 the abscissa ('x-axis') is the hat value. Two things are of interest:

- No extreme leverage, compared to other observations;
- Residuals at high-leverage observations should not be too much smaller than for other observations. Note that high-leverage observation "pull" the regression towards better fits, so their residuals are expected to be somewhat below average.

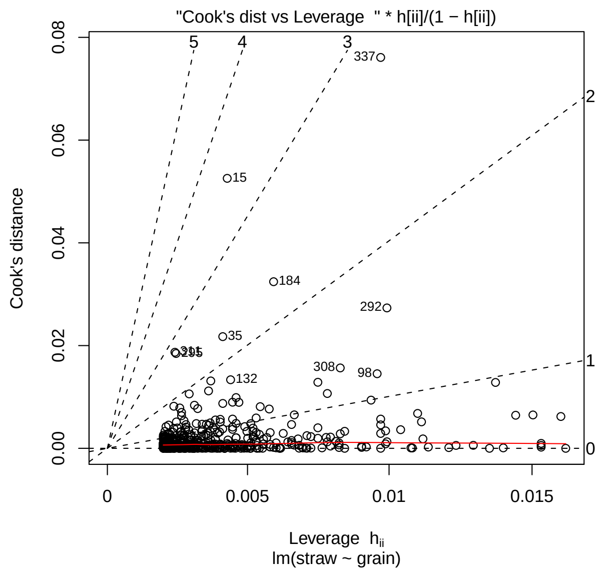
```
> plot(model.straw.grain, which = 5)
```



Plot type 6 is Cook's distance vs. leverage. Cook's distance measures how much the estimated parameter vector $\hat{\beta}$ shifts if a single observation is omitted. A high Cook's distance means that the observation has a large influence on the fitted model.

We also specify the number of extreme points to label with the `id.n` optional argument.

```
> plot(model.straw.grain, which = 6, id.n = 10)
```



This plot also shows contours of absolute standardized residuals, as labelled straight lines through the origin.

Q47 : *Which observations are flagged by their Cook's distance as having the most influence on the fit? Are these high-leverage observations? Jump to A47* •

We will compare this model to more complex ones in §11, below.

9.3 Structural Analysis*

In §9.2.1 we modelled one of the two variables as as a response and the other as a predictor, and fit a line that best describes this relation. If we reverse the relation, what happens?

Task 38 : Compare the regression of strain yield on grain yield, with the regression of grain yield on straw yield. •

```
> model.grain.straw <- lm(grain ~ straw, data = mhw)
> summary(model.grain.straw)

Call:
lm(formula = grain ~ straw, data = mhw)

Residuals:
      Min        1Q    Median        3Q       Max 
-1.358046 -0.201069  0.000394  0.191773  1.052682 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   1.5231     0.1028    14.8   <2e-16 ***
straw         0.3723     0.0156    23.8   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.314 on 498 degrees of freedom
Multiple R-squared: 0.533,	Adjusted R-squared: 0.532 
F-statistic:  567 on 1 and 498 DF,  p-value: <2e-16 

> summary(model.straw.grain)

Call:
lm(formula = straw ~ grain)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.0223 -0.3529  0.0104  0.3734  3.0342 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   0.8663     0.2387    3.63  0.00031 ***
grain         1.4305     0.0601   23.82  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.615 on 498 degrees of freedom
Multiple R-squared: 0.533,	Adjusted R-squared: 0.532 
F-statistic:  567 on 1 and 498 DF,  p-value: <2e-16 
```

Q48 : *Is the amount of variability explained by the two models the same? That is, does knowing straw yield give the same amount of information on grain yield as the reverse?* *Jump to A48* •

Intuitively it might seem that the slope of grain vs. straw would be the inverse of the slope of straw vs. grain. Is this the case?

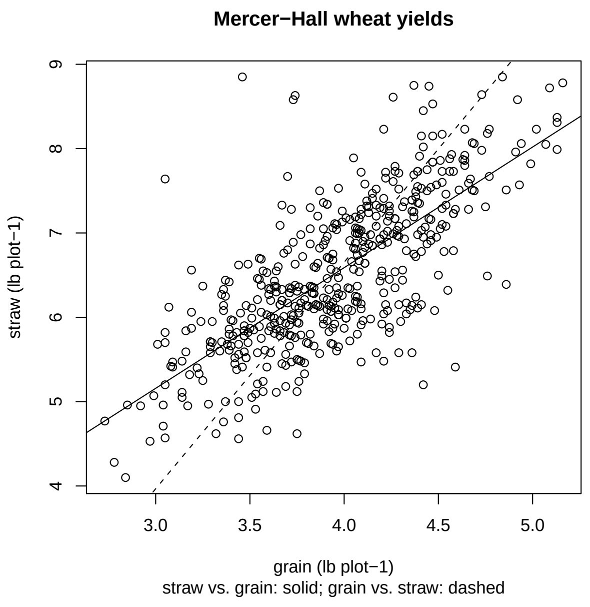
Task 39 : Compute the slope of straw vs. grain as the inverse of the modelled slope of grain vs. straw, and compare with the modelled slope of straw vs. grain. •

```
> coefficients(model.straw.grain)["grain"]

 grain
1.4305

> 1/coefficients(model.grain.straw)["straw"]

 straw
2.6860
```

We can visualize these on the scatterplot of straw vs. grain. The regression line of straw on grain can be directly plotted with `abline` on the model object; the reverse regression must be inverted from coefficients extracted from its model object. The slope is just the inverse; the intercept is the straw yield corresponding to zero grain yield:

$$\begin{aligned} \text{grain} &= b_0 + b_1\text{straw} \\ 0 &= b_0 + b_1\text{straw} \\ \text{straw} &= -b_0/b1 \end{aligned}$$

```
> plot(straw ~ grain, pch = 1, main = "Mercer-Hall wheat yields",
+     xlab = "grain (lb plot-1)", ylab = "straw (lb plot-1)")
> title(sub = "straw vs. grain: solid; grain vs. straw: dashed")
> abline(model.straw.grain)
> beta <- coefficients(model.grain.straw)
> abline(-beta["(Intercept)"]/beta["straw"], 1/beta["straw"],
+     lty = 2)
> rm(beta)
```



Q49 : *Do these two models give the same straw vs. grain relation? Why not?* *Jump to A49* •

So, the regression of two variables on each other depends on which variables is considered the predictor and which the predictand. If we are predicting, this makes sense: we get the best possible prediction. But sometimes we are interested not in prediction, but in *understanding* a relation between two variables. In the present example, we may ask what is the true relation between straw and grain in this wheat variety? Here we assume that this relation has a common cause, i.e. plant growth processes affect the grain and straw in some systematic way, so that there is a consistent relation between them. This so-called **structural analysis** is explained in detail by Sprent [28] and more briefly by Webster [33] and Davis [5, pp. 218–219].

In structural analysis we are trying to establish the best estimate for a **structural** or **law-like** relation, i.e. where we hypothesise that $y = \alpha + \beta x$, where both x and y are mathematical variables. This is appropriate when there is no need to predict, but rather to understand. This depends on the prior assumption of a true linear relation, of which we have a noisy sample.

$$X = x + \xi \tag{1}$$
$$Y = y + \eta \tag{2}$$

That is, we want to observe X and Y, but instead we observe x with random error ξ and y with random error η. These errors have (unknown) variances σ_ξ^2 and σ_η^2, respectively; the ratio of these is crucial to the analysis, and is symbolised as λ:

$$\lambda = \sigma_\eta^2 / \sigma_\xi^2 \tag{3}$$

Then the maximum-likelihood estimator of the slope $\hat{\beta}_{Y.X}$, taking Y as the predictand for convention, is:

$$\hat{\beta}_{Y.X} = \frac{1}{2s_{XY}}\left\{(s_Y^2 - \lambda s_X^2) + \sqrt{(s_Y^2 - \lambda s_X^2)^2 + 4\lambda s_{XY}^2}\right\} \qquad (4)$$

Equation 4 is only valid if we can assume that the **errors in the two variables are uncorrelated**. In the present example, it means that a large random deviation for a particular sample of the observed straw yield from its "true" value does *not* imply anything about the random deviation of the observed grain yield from *its* "true" value.

The problem is that we don't have any way of knowing the true error variance ratio λ (Equation 3), just as we have no way of knowing the true population variances, covariance, or parameters of the structural relation. We estimate the **population** variances σ_X^2, σ_Y^2 and covariance σ_{XY} from the sample variances s_x^2, s_y^2 and covariance s_{xy}, but there is nothing we've measured from which we can estimate the **error** variances or their ratio. However, there are several plausible methods to estimate the ratio:

- If we can assume that the two error variances are **equal**, $\lambda = 1$. This may be a reasonable assumption if the variables measure the same property (e.g. both measure clay content in different soil layers), use the same method for sampling and analysis, and there is an *a priori* reason to expect them to have similar variability (heterogeneity among samples). However in this case there is no reason to expect equal variances.
- The two **error** variances may be estimated by the ratio of the **sample** variances: $\lambda \approx s_y^2/s_z^2$. That is, we assume that the ratio of variability in the measured variable is also the ratio of variability in their errors. For example, if the set of straw yields in a sample is twice as variable as the set of grain yields in the same sample, we would infer that the error variance of straw yields is also twice as much that for grain yields, so that $\lambda = 2$. But, these are two completely different concepts! One is a sample variance and the other the variance of the error in some random process. However, this ratio at least normalizes for different units of measure and for different process intensities. Using this value of λ computes the **Reduced Major Axis** (RMA), which is popular in biometrics.
- The variance ratio may be known from previous studies.

Task 40 : Compute the variance ratio of straw and grain yields. •

```
> var(straw)/var(grain)

[1] 3.8423
```

Q50 : *Is straw or grain yield more variable across the 500 plots? What is the ratio?* *Jump to A50* •

Task 41 : Write an R function to compute $\hat{\beta}_{Y.X}$, given the two structural variables in the order predictand, predictor and the ratio of the error variances λ. •

```
> struct.beta <- function(y, x, lambda) {
+     a <- var(y) - lambda * var(x)
+     c <- var(x, y)
+     (a + sqrt(a^2 + 4 * lambda * c^2))/(2 * c)
+ }
```

Task 42 : Apply this function to the straw vs. grain yields:

1. with $\lambda = 1$; this is the **orthogonal** estimate;
2. with λ as the variance ratio of straw and grain yields (assuming the error variance ratio equals the variables' variance ratio); this is the **proportional** estimate.

Compare with the slopes of the forward and reverse regressions. •

```
> print(paste("Forward:", round(coefficients(model.straw.grain)["grain"],
+     4)))

[1] "Forward: 1.4305"

> print(paste("Proportional:", round(struct.beta(straw,
+     grain, var(straw)/var(grain)), 4)))

[1] "Proportional: 1.9602"

> print(paste("Inverse proportional:", round(1/struct.beta(grain,
+     straw, var(grain)/var(straw)), 4)))

[1] "Inverse proportional: 1.9602"

> print(paste("Orthogonal:", round(struct.beta(straw, grain,
+     1), 4)))

[1] "Orthogonal: 2.4031"

> print(paste("Inverse orthogonal:", round(1/struct.beta(grain,
+     straw, 1), 4)))

[1] "Inverse orthogonal: 2.4031"

> print(paste("Reverse:", round(1/coefficients(model.grain.straw)["straw"],
+     4)))

[1] "Reverse: 2.686"
```

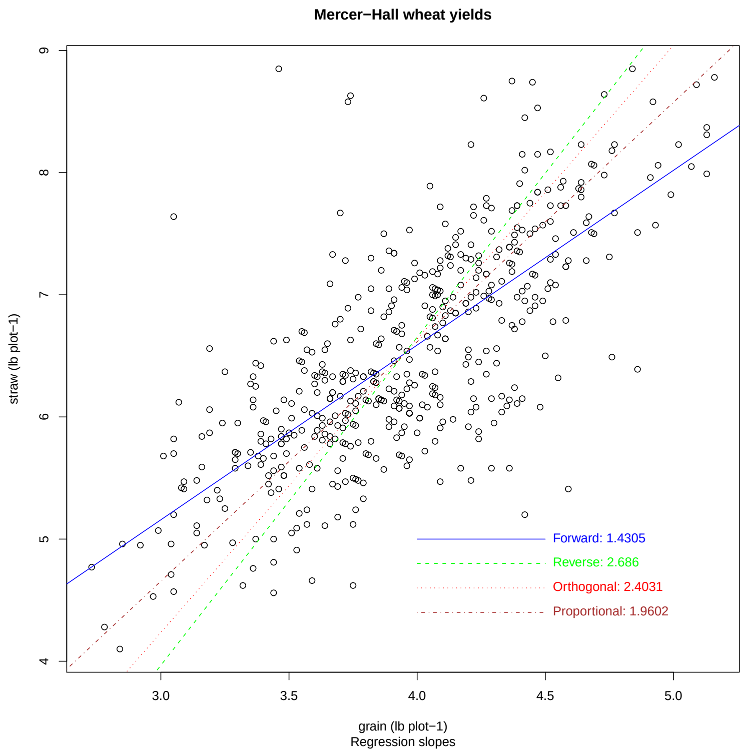
Note that it is numerically between the slopes of the forward and inverse regressions.

Task 43 : Plot the forward, reverse, orthogonal and proportional regression lines on one scatterplot of straw vs. grain yield. •

For the models fit with lm we can extract the coefficients; for the structural models we compute the slopes with our user-written function struct.beta and then their intercepts with the relation:

$$\hat{\beta}_0 = \hat{\mu}_\mathcal{Y} - \hat{\beta}_{\mathcal{Y}.\mathcal{X}} \hat{\mu}_\mathcal{X}$$

```
> plot(straw ~ grain, main = "Mercer-Hall wheat yields",
+     sub = "Regression slopes", xlab = "grain (lb plot-1)",
+     ylab = "straw (lb plot-1)")
> abline(model.straw.grain, col = "blue")
> beta <- coefficients(model.grain.straw)
> abline(-beta["(Intercept)"]/beta["straw"], 1/beta["straw"],
+     lty = 2, col = "green")
> beta <- struct.beta(straw, grain, 1)
> abline(mean(straw) - beta * mean(grain), beta, lty = 3,
+     col = "red")
> beta <- struct.beta(straw, grain, var(straw)/var(grain))
> abline(mean(straw) - beta * mean(grain), beta, lty = 4,
+     col = "brown")
> lines(c(4, 4.5), c(5, 5), lty = 1, col = "blue")
> lines(c(4, 4.5), c(4.4, 4.4), lty = 4, col = "brown")
> lines(c(4, 4.5), c(4.6, 4.6), lty = 3, col = "red")
> lines(c(4, 4.5), c(4.8, 4.8), lty = 2, col = "green")
> text(4.5, 5, paste("Forward:", round(coefficients(model.straw.grain)["grain"],
+     4)), col = "blue", pos = 4)
> text(4.5, 4.4, paste("Proportional:", round(struct.beta(straw,
+     grain, var(straw)/var(grain)), 4)), col = "brown",
+     pos = 4)
> text(4.5, 4.6, paste("Orthogonal:", round(struct.beta(straw,
+     grain, 1), 4)), col = "red", pos = 4)
> text(4.5, 4.8, paste("Reverse:", round(1/coefficients(model.grain.straw)["straw
+     4)), col = "green", pos = 4)
```



Q51 : *What do you conclude is the best numerical expression of the structural relation between straw and grain yields in this variety of wheat, grown in this field?* *Jump to A51* •

Challenge: Modify function `struct.beta` to return both the intercept and slope of the structural line[7], and use this to simplify the display of lines on the scatterplot.

9.4 Answers

A31 : *Linear.* *Return to Q31* •

A32 : *The diagonals are the variances of the two variables, both in (lbs per plot) squared; the off-diagonals are the covariances between the two variables, in this case also in (lbs per plot) squared, because the two variables have the same units of measure.* *Return to Q32* •

A33 : *The summary statistics are quite similar, for both variables; the simulation reproduces the statistics of the actual data.* *Return to Q33* •

A34 : *The relations look quite similar; this supports the hypothesis. However,*

[7] Hint: use the `c` "make a list" function

the bivariate simulation seems to have a slightly steeper slope than the actual data.
Return to Q34 •

A35 : *The most probable value is 0.73 lbs per plot; the lower and upper confidence limits are 0.686 and 0.768 lbs per plot, respectively.*

Assessment of the strength is subjective and depends on the application field; the author would call this a moderate positive correlation. Return to Q35 •

A36 : *The first plot has grain and straw yields of 3.63 and 6.37, respectively; these are ranked 123 and 254.5 in their respective vectors. The grain yield is thus just under the first quartile (123 out of 500) whereas the straw yield is higher, just under the median (254 out of 500). For this plot, the straw yield ranks considerably higher than the grain yield.*

Overall, in the ten displayed ranks, there is a clear correspondence, but not very close. Return to Q36 •

A37 : *The scatterplot of ranks is more diffuse; this is because the ranks throw away much information (the actual numeric values). In the case of bivariate normal distributions, the tails (extreme values) contribute disproportionately to the correlation.* Return to Q37
•

A38 : *No; both are caused by the same underlying process (plant growth in response to the environment), and neither is more under control of the experimenter. However, see the next question.* Return to Q38 •

A39 : *It can be used to* **understand** *plant physiology: is grain yield a direct result of straw yield, or at least do large plants (lots of straw) tend to have high yield (lots of grain)? Practically, we could use this relation to predict straw yield on other plots where only grain was weighed; it is much easier to collect and weigh grain than straw.* Return to Q39 •

A40 : straw = 0.866 + 1.43 · grain Return to Q40 •

A41 : 0.866 *lbs; maybe wheat this small would indeed have no grain, because the plants would be too weak to form grain.* Return to Q41 •

A42 : 1.43 *lbs of straw increase for each lb of grain increase.* Return to Q42 •

A43 : *53.2% (the value of the adjusted R^2).* Return to Q43 •

A44 : *It's clear that the highest residuals are too low and vice-versa; the histogram* *is somewhat peaked. The median residual is slightly biased* (-0.01)*. The range is quite high, from* -2 *to* $+3$ *lbs per plot.* Return to Q44 •

A45 : *The p-value (probability that rejecting the null hypothesis would be an error) is almost zero, so we should reject the null hypothesis: these residuals are* not *normally-distributed. This is due to the deviations at both tails.* Return to Q45 •

A46 : *Plots 15, 337, 311 and 295 have very low grain/straw ratios, so the linear relation predicts too much grain; for plots 35 and 184 it's the reverse.* Return to Q46 •

A47 : *Observations 292, 184, 15, and especially 337. Plots 337 and 292 also have high leverage. In the previous answer we saw that plots 15 and 337 have very low grain/straw ratios, so the linear relation predicts too much grain; for plot 184 it's the reverse. Plot 292 has high leverage and fairly high Cook's distance, but its standarized residual is not so high (< 2).* Return to Q47 •

A48 : *The models explain the same proportion of the total variability, 0.532.* Return to Q48 •

A49 : *The slopes are very different. The forward model gives a slope of 1.43 of straw vs. grain, whereas the inverse model gives a slope of 2.686. The reason is that the two regressions minimize different error sums-of-squares: in the forward model, residual straw yield, in the inverse model, residual grain yield. Each is the best predictive relation for its target variable.* Return to Q49 •

A50 : *The variance ratio is 3.842, that is, straw yields are almost four times as variable as grain yields. This is partly explained by the higher absolute values of all the yields: the medians are 6.36 for straw vs. 3.94 for grain.* Return to Q50 •

A51 : *The best estimate of the error variance ratio is the variable variance ratio, so the structural relation is 1.96 lb straw for each lb grain; or equivalently 0.51 lb grain for each lb straw. This is the best estimate of the plant morphology.* Return to Q51 •

10 Bivariate modelling: continuous vs. classified variables

A continuous variable can also be modelled based on some **classified** (discrete) predictor. In the present example we will consider the field half (North & South) to be such a predictor.

It has been suggested [30] that the North and South halves of the field had been treated differently prior to Mercer & Hall's experiment. To test this suggestion, we first have to code each plot according to its location in the field (N or S half); this is a **logical variable** (True or False, written in S as `TRUE` or `FALSE`, abbreviated `T` and `F`). Then we use this variable to split the field statistically and compare the yields for each half. Each plot is thus in one or the other **class**, in this case field half.

Task 44 : Add a logical field to the data frame to codes whether each plot falls in the north half or not. •

We first use a **logical expression** that evaluates to either `T` or `F` to create a **logical variable**, here named `in.north`, as a field in the data frame. This field codes whether teach plot falls in the north half or not. We then detach and re-attach the data frame to make the name `in.north` visible.

Recall from the description of the dataset in Appendix A that the rows ran W to E, with 25 plots in each row, beginning at 1 on the W and running to 25 at the E, and the columns run N to S with 20 plots in each, beginning at 1 on the N and running to to 20 at the S. So the N half of the field consists of the plots with row numbers from 1 to 10, inclusive.

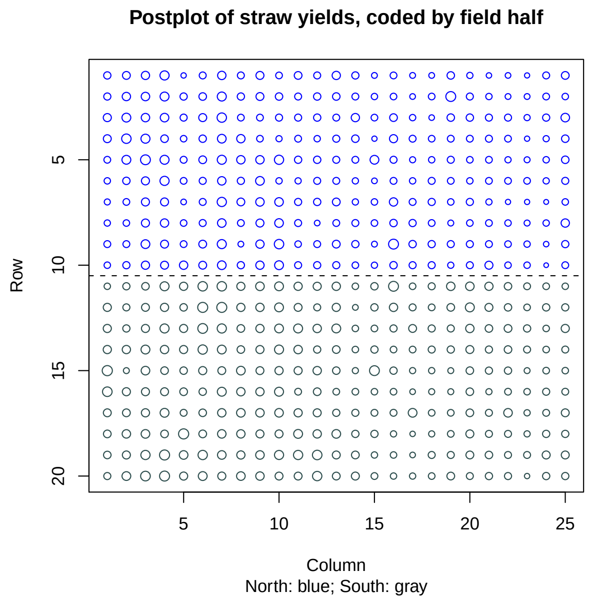
```
'data.frame':        500 obs. of  6 variables:
 $ r       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ c       : int  1 1 1 1 1 1 1 1 1 1 ...
 $ grain   : num  3.63 4.07 4.51 3.9 3.63 3.16 3.18 3.42 3.97 3.4 ...
 $ straw   : num  6.37 6.24 7.05 6.91 5.93 5.59 5.32 5.52 6.03 5.66 ...
 $ gsr     : num  0.57 0.652 0.64 0.564 0.612 ...
 $ in.north: logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
```

```
> mhw <- cbind(mhw, in.north = (r < 11))
> str(mhw)

> detach(mhw)
> attach(mhw)

> summary(in.north)

   Mode   FALSE    TRUE    NA's
logical     250     250       0
```

Task 45 : Display a postplot of grain yields with the plots in the North half coloured blue, those in the South coloured grey[8]. •

```
> plot(c, r, col = ifelse(in.north, "blue", "darkslategrey"),
+     cex = 1.3 * straw/max(straw), pch = 1, xlab = "Column",
+     ylab = "Row", ylim = c(20, 1), sub = "North: blue; South: gray")
> title(main = "Postplot of straw yields, coded by field half")
> abline(h = 10.5, lty = 2)
```

[8] An obvious historical reference



Note: Note the use of the `ylim` graphics argument, to specify that the labels of the y-axis run from 20 (lower left corner) to 1 (upper left corner); by default scatterplots drawn by the `plot` function assume the lowest-numbered row is the lower left. This is the usual case for scatterplots, but here we know the lowest-numbered row is at the N side.

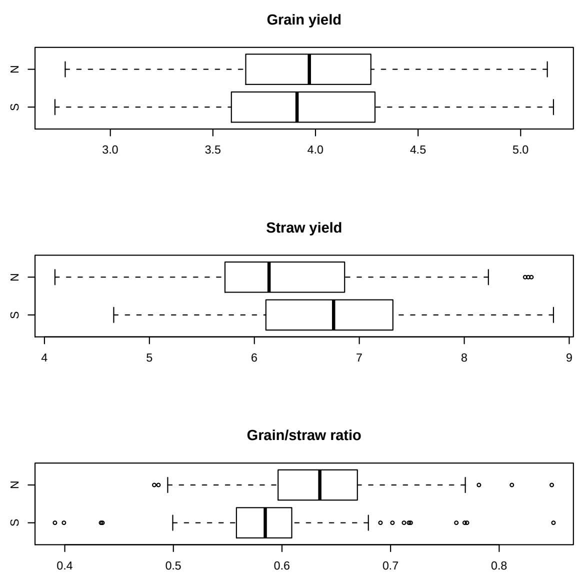
10.1 Exploratory data analysis

We first compare the two halves with exploratory graphics; a suitable graph is the **boxplot**, created with the `boxplot` function.

Task 46 : Compare the two field halves with box plots. •

To compare these on one graph, we split the graphics frame by specifying the number of rows and columns with the `mfrow` argument to the `par` ("graphics parameters") command. These plots look better if they are displayed horizontally, using the optional `horizontal=T` argument. Note the use of the optional `names` argument to label the plots; these are `S` and `N` to represent the internal values `FALSE` and `TRUE` of the `in.north` classifier.

```
> par(mfrow = c(3, 1))
> boxplot(grain ~ in.north, names = c("S", "N"), main = "Grain yield",
+     horizontal = T)
> boxplot(straw ~ in.north, names = c("S", "N"), main = "Straw yield",
+     horizontal = T)
> boxplot(gsr ~ in.north, names = c("S", "N"), main = "Grain/straw ratio",
+     horizontal = T)
> par(mfrow = c(1, 1))
```



Q52 : *Do the two halves appear to have different ranges? medians? spreads? Comment for all three variables.* *Jump to A52* •

We then compare the two halves numerically:

Task 47 : Compare the summary statistics of the two halves for all the variables. Also compare their variances. •

Any function can be applied to subsets of a data frame with the `by` function. The first argument is the argument to the function, the second is the subset classifier, and the third the function to be applied:

```
> by(grain, in.north, summary)

in.north: FALSE
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.73    3.59    3.91    3.93    4.29    5.16
----------------------------------------------------
in.north: TRUE
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.78    3.66    3.97    3.96    4.27    5.13

> by(straw, in.north, summary)

in.north: FALSE
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4.66    6.11    6.76    6.75    7.32    8.85
----------------------------------------------------
in.north: TRUE
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4.10    5.73    6.14    6.28    6.86    8.64

> by(gsr, in.north, summary)
```

```
in.north: FALSE
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.391   0.558   0.585   0.586   0.609   0.850 
---------------------------------------------------- 
in.north: TRUE
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.482   0.596   0.635   0.636   0.669   0.848 
```

```
> by(grain, in.north, var)

in.north: FALSE
[1] 0.22162
---------------------------------------------------- 
in.north: TRUE
[1] 0.19876
```

```
> by(straw, in.north, var)

in.north: FALSE
[1] 0.74777
---------------------------------------------------- 
in.north: TRUE
[1] 0.75985
```

```
> by(gsr, in.north, var)

in.north: FALSE
[1] 0.0026123
---------------------------------------------------- 
in.north: TRUE
[1] 0.0030585
```

Q53 : *Do the two halves have different summary statistics? Is one half more variable than the other?* *Jump to A53* •

From the boxplots, it appears that the straw yield is, on average, higher in the S half; can we confirm this with a statistical test?

Task 48 : Test whether the straw yield is higher in the N half. •

There are two approaches that give the same answer for a binomial classified predictor: a two-sample t-test, and a one-way ANOVA.

10.2 Two-sample t-test

The simplest way to do this is with a **two-sample unpaired t test** of the difference between means, with the default **null hypothesis** that they are identical. This only works when the classified variable is binary.

```
> t.test(straw[in.north], straw[!in.north])

        Welch Two Sample t-test

data:  straw[in.north] and straw[!in.north]
t = -6.0152, df = 497.97, p-value = 3.481e-09
alternative hypothesis: true difference in means is not equal to 0
```

```
95 percent confidence interval:
 -0.61969 -0.31455
sample estimates:
mean of x mean of y
   6.2812    6.7484
```

Q54 : *Is there a significant difference in the means? What is the probability that this apparent difference is only by chance (i.e. a Type I error would be committed if we reject the null hypothesis)? What is the 95% confidence interval of the difference?* *Jump to A54* •

10.3 One-way ANOVA

Another way to analyze this is with a **one-way Analysis of Variance** (ANOVA). This can also deal with multivalued classified predictors, although in this case we only have a binary predictor.

Task 49 : Compute a one-way ANOVA for straw yield between field halves. •

This illustrates another use of the `lm` function, i.e. modelling a continuous response variable from a categorical (here, binary) predictor:

```
> model.straw.ns <- lm(straw ~ in.north, data = mhw)
> summary(model.straw.ns)

Call:
lm(formula = straw ~ in.north, data = mhw)

Residuals:
   Min     1Q Median     3Q    Max
-2.181 -0.608 -0.108  0.572  2.359

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    6.7484     0.0549  122.90  < 2e-16 ***
in.northTRUE  -0.4671     0.0777   -6.02  3.5e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.868 on 498 degrees of freedom
Multiple R-squared: 0.0677,      Adjusted R-squared: 0.0659
F-statistic: 36.2 on 1 and 498 DF,  p-value: 3.48e-09
```

Q55 : *How much of the total variability in straw yield is explained by field half?* *Jump to A55* •

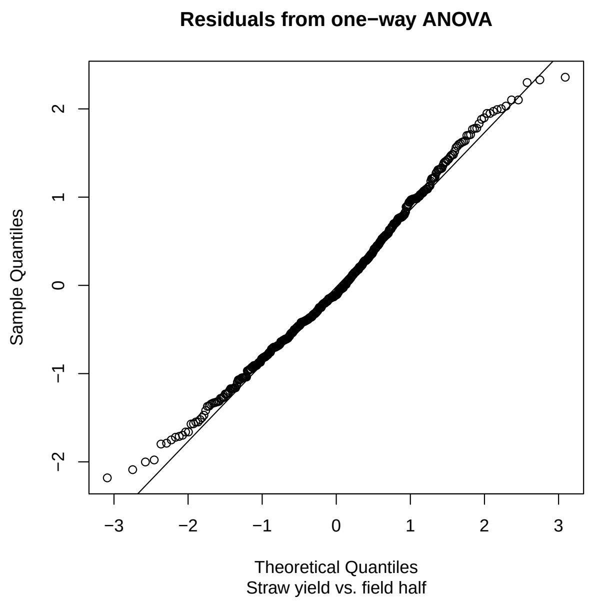
We can also see the results with a traditional ANOVA table:

```
> anova(model.straw.ns)

Analysis of Variance Table
```

```
Response: straw
           Df Sum Sq Mean Sq F value  Pr(>F)    
in.north    1     27      27    36.2 3.5e-09 ***
Residuals 498    375       1                    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And of course we must check the regression diagnostics:

```
> qqnorm(residuals(model.straw.ns), main = "Residuals from one-way ANOVA",
+     sub = "Straw yield vs. field half")
> qqline(residuals(model.straw.ns))
```



Q56 : *Are the model residuals normally-distributed? What does this imply about the process in the field?* Jump to A56 •

We will compare this model to more complex ones in §11, below.

10.4 Answers

A52 : *Grain yields appear to be almost identically distributed, although the S half is slightly more variable. Straw yields appear slightly higher in the S. The grain/straw ratio appears higher in the N. Variability between field halves seems similar for grain and straw, but the grain/straw ratio in the S half appears to have more total spread and boxplot outliers.* Return to Q52 •

A53 : *Grain: Almost identical summary statistics; Straw: The S is somewhat higher in all summary statistics, but the variability is almost the same; Grain/straw ratio: the S is higher in all summary statistics except the maximum, which is almost* *the same, but the N is more variable.* *Return to Q53* •

A54 : *Yes, very highly significant; the N has a higher ratio than the S. the probability of a Type I error is almost 0. The 95% confidence interval is -0.62, -0.315, i.e. there is only a 2.5% chance that the difference in yields is not at least 0.315.* *Return to Q54* •

A55 : *In the summary, the* `Adjusted R^2` *gives this proportion; here it is 0.066, a very low proportion. This shows that a model can be highly significant – the difference between class means is almost surely not zero – but numerically not important.* *Return to Q55* •

A56 : *For the most part yes, but the tails (highest and lowest ratios) are not well-modelled: the absolute residuals are too high at both tails, especially the lower tail. This suggests that the extreme values are caused by some process that is beyond what is causing most of the "random" variability. This is sometimes called a "contaminated" process.* *Return to Q56* •

11 Multivariate modelling

In §9.2 we determined that the regression of straw yield on grain yields, over the whole field, was significant: i.e. straw yield does vary with grain yield. In §10 we determined that straw yields of the two field halves differ significantly. This raises the question whether it is possible and useful to combine these two findings, i.e. whether straw yield can be modelled as a function of both field half and grain yield.

There are four possibilities, from simplest to most complex:

1. Straw yields can be modelled by field half only;
2. Straw yields can be modelled by grain yield only;
3. Straw yields can be modelled by field half and grain yield, additively (i.e. with these as independent predictors);
4. Straw yields must be modelled by field half and grain yield, considering also the interaction between them (i.e. the relation between straw and grain yield differs in the two field halves).

These are tested with increasingly-complex linear models:

1. One-way ANOVA (or, two-sample t-test) of straw yield; following the evidence in the boxplot of §10.1 that seemed to show slightly higher yields in the north half; this was computed in §10.3 above;
2. Linear model of straw yield predicted by grain yield only, following the evidence of the scatterplot, ignoring field halves; this was computed in §9.2 above;
3. Linear model of straw yield predicted by field half (a categorical variable) and grain yield (a continuous variable), with an additive model

(§11.1, below);

4. Linear model of straw yield predicted by field half (a categorical variable) and grain yield (a continuous variable), with an interaction model (§11.3, below);

We now compute all of these and compare the proportion of the total variation in straw yield that each explains.

Task 50 : Display the results of the field-half and bivariate regression models. •

Straw yield vs. field half was already computed in §10.3 above and saved as model object `model.straw.ns`:

```
> summary(model.straw.ns)

Call:
lm(formula = straw ~ in.north, data = mhw)

Residuals:
   Min     1Q Median     3Q    Max 
-2.181 -0.608 -0.108  0.572  2.359 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)    6.7484     0.0549  122.90  < 2e-16 ***
in.northTRUE  -0.4671     0.0777   -6.02  3.5e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.868 on 498 degrees of freedom
Multiple R-squared: 0.0677,	Adjusted R-squared: 0.0659 
F-statistic: 36.2 on 1 and 498 DF,  p-value: 3.48e-09
```

Q57 : *Is there evidence that average straw yield differs in the two field halves? What proportion of the total variation in straw yield is explained by field half?* *Jump to A57* •

Straw yield vs. grain yield was computed in §9.2 above and saved as model object `model.straw.grain`:

```
> summary(model.straw.grain)

Call:
lm(formula = straw ~ grain)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.0223 -0.3529  0.0104  0.3734  3.0342 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   0.8663     0.2387    3.63  0.00031 ***
grain         1.4305     0.0601   23.82  < 2e-16 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.615 on 498 degrees of freedom
Multiple R-squared: 0.533,      Adjusted R-squared: 0.532
F-statistic:  567 on 1 and 498 DF,  p-value: <2e-16
```

Q58 : *Is there evidence that straw yield varies with grain yield? What proportion of the total variation in straw yield is explained by grain yield?* *Jump to A58* •

Since both of these predictors (field half and grain yield) do explain some of the straw yield, it seems logical that a combination of the two, i.e. a **multivariate** model, might explain more than either separately. So, we now model straw yield vs. grain yield, also accounting for the overall difference between field halves.

11.1 Additive model: parallel regression

The simplest multivariate model is an **additive** model, also called **parallel regression** because it fits one regression line, but with the intercept at different levels, one for each field half.

Task 51 : Model straw yield as the combined effect of two independent predictors: field half and grain yield. •

We use the `lm` function, naming both predictors on the right-hand side of the model formula, combined with the `+` "additive effects" formula operator. Note this is *not* an arithmetic `+` (addition).

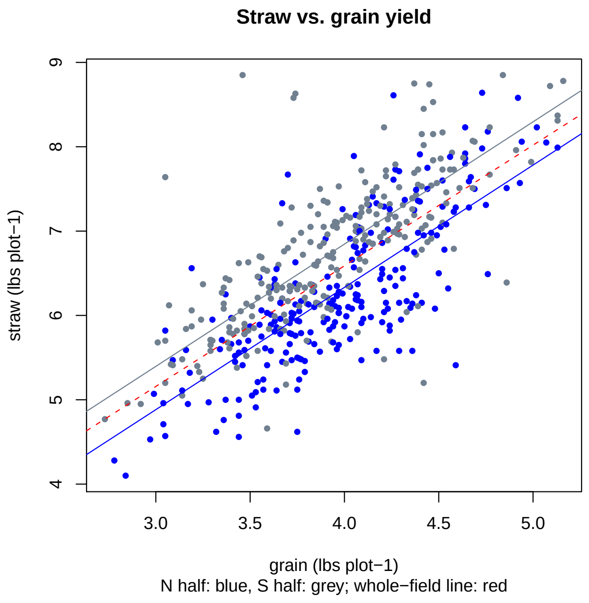
```
> model.straw.ns.grain <- lm(straw ~ in.north + grain,
+     data = mhw)
> summary(model.straw.ns.grain)

Call:
lm(formula = straw ~ in.north + grain, data = mhw)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.2548 -0.3189 -0.0276  0.3042  2.7871 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)    1.0461     0.2178     4.8  2.1e-06 ***
in.northTRUE  -0.5132     0.0500   -10.3  < 2e-16 ***
grain          1.4499     0.0546    26.5  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.559 on 497 degrees of freedom
Multiple R-squared: 0.614,      Adjusted R-squared: 0.613
F-statistic:  396 on 2 and 497 DF,  p-value: <2e-16
```

Q59 : *Is there evidence that both field half and grain yield are needed to predict straw yield? What proportion of the total variation in straw yield is explained by this model?* Jump to A59 •

Q60 : *According to the model summary, what is the difference in overall yields between the S and N halves? What is the slope of the regression line for straw vs. grain yield? Is this the same as the slope for the model that does not include field half?* Jump to A60 •

In parallel regression (additive effects of a continuous and discrete predictor) there is only one regression line, which is displaced up or down for each class of the discrete predictor. Even though there are two predictors, we can visualize this in a 2D plot by showing the displaced lines.

```
> plot(straw ~ grain, col = ifelse(in.north, "blue", "slategray"),
+     pch = 20, xlab = "grain (lbs plot-1)", ylab = "straw (lbs plot-1)")
> title(main = "Straw vs. grain yield")
> title(sub = "N half: blue, S half: grey; whole-field line: red")
> abline(coefficients(model.straw.ns.grain)["(Intercept)"],
+     coefficients(model.straw.ns.grain)["grain"], col = "slategray")
> abline(coefficients(model.straw.ns.grain)["(Intercept)"] +
+     coefficients(model.straw.ns.grain)["in.northTRUE"],
+     coefficients(model.straw.ns.grain)["grain"], col = "blue")
> abline(model.straw.grain, lty = 2, col = "red")
```



In the code for this plot, note the use of the `coefficients` function to extract the model coefficients.

11.2 Comparing models

Is a more complex model better than a simpler one? There are several ways to answer this, among which are:

- Compare the adjusted R^2 of the two models: this is the proportion of the variance explained;
- Directly compare hierarchical models with an Analysis of Variance.

Task 52 : Compare the adjusted R^2 of the three models. •

We've already seen these in the model summaries; they can be accessed directly as field `adj.r.squared` of the model summary:

```
> summary(model.straw.ns)$adj.r.squared

[1] 0.065864

> summary(model.straw.grain)$adj.r.squared

[1] 0.53164

> summary(model.straw.ns.grain)$adj.r.squared

[1] 0.61268
```

Q61 : *Which model explains the most variability?* *Jump to A61* •

Another way to compare two **hierarchical** models (i.e. where the more complex model has all the predictors of the simpler one) is with an analysis of variance: comparing variance explained vs. degrees of freedom. This is a statistical test, so we can determine whether the more complex model is provably better.

Task 53 : Compare the additive multivariate model to the two univariate models. •

The `anova` function can be used to compare the models:

```
> anova(model.straw.ns.grain, model.straw.ns)

Analysis of Variance Table

Model 1: straw ~ in.north + grain
Model 2: straw ~ in.north
  Res.Df  RSS  Df Sum of Sq   F Pr(>F)
1    497  155
2    498  375  -1      -220 704 <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> anova(model.straw.ns.grain, model.straw.grain)

Analysis of Variance Table

Model 1: straw ~ in.north + grain
Model 2: straw ~ grain
  Res.Df   RSS  Df Sum of Sq   F Pr(>F)
1    497 155.3
```

```
2    498 188.2  -1     -32.9 105 <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The `anova` function compares the **residual sum of squares** (RSS) of the two models; this is the amount of variability not explained by the model, so a lower RSS is better. It then computed the F-ratio between the two variances, and the probability that this large an F-value, with the **degrees of freedom** (d.f.) could occur by chance, if the null hypothesis of no model improvement is true. The probability of a Type I error (falsely rejecting a true null hypothesis) is reported as field `Pr(>F)`; lower is better.

Q62 : *Is the multivariate additive model proveably better than either univariate model?* *Jump to A62* •

11.3 Interaction model

We saw that the additive model is superior to either single-predictor model. However, there is also the possibility that both field half and grain yield help predict straw yield, but that the relation between straw and grain is different in the two halves; this is known as an **interaction**. This allows a different linear regression in each field half, rather than a parallel regression.

Q63 : *What is the difference between an additive and interaction model, with respect to processes in the field?* *Jump to A63* •

Task 54 : Model straw yield as the combined effect of two interacting predictors: field half and grain yield. •

We use the `lm` function, naming both predictors on the right-hand side of the model formula, combined with the `*` "interactive effects" formula operator. This is *not* an arithmetic `*` (multiplication).

```
> model.straw.ns.grain.i <- lm(straw ~ in.north * grain,
+     data = mhw)
> summary(model.straw.ns.grain.i)

Call:
lm(formula = straw ~ in.north * grain, data = mhw)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.2242 -0.3169 -0.0398  0.3136  2.7574 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)          1.2930     0.2979    4.34  1.7e-05 ***
in.northTRUE        -1.0375     0.4350   -2.39    0.017 *  
grain                1.3872     0.0752   18.44  < 2e-16 ***
in.northTRUE:grain   0.1328     0.1094    1.21    0.225    
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.559 on 496 degrees of freedom
Multiple R-squared: 0.615,	Adjusted R-squared: 0.613
F-statistic:  265 on 3 and 496 DF,  p-value: <2e-16
```

Q64 : *What do the four coefficients in the regression equation represent?* *Jump to A64* •

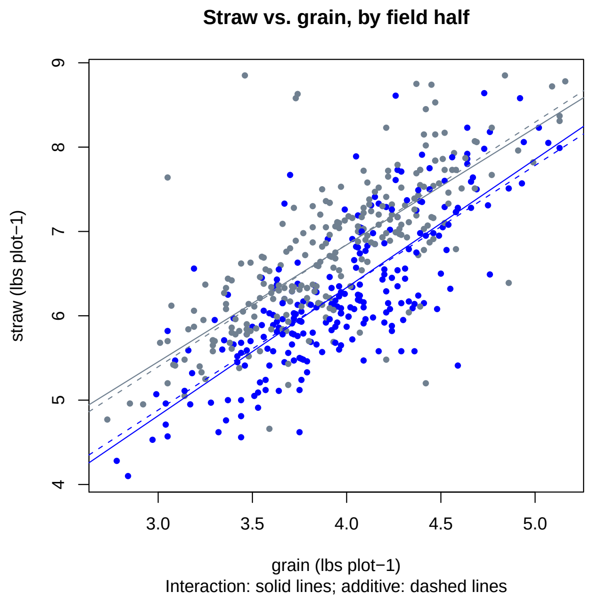
Q65 : *Are the two slopes (one for each field half) significantly different? Is an interaction model indicated? What does that imply about the processes in the field?* *Jump to A65* •

Even though the slopes are not significantly different, we show them graphically, to visualize how different they are.

Task 55 : Plot the two regressions (one for each field half). •

To do this, we use the `subset=` optional argument to the `lm` method to select just some observations, in this case, those in a zone. We plot each regression line (using the `abline` function on the model object returned by `lm`) and its associated points in different colours, using the `col` graphics argument. Dashed lines (using the `lty` graphics argument) show the parallel regression for the two field halves.

```
> plot(straw ~ grain, col = ifelse(in.north, "blue", "slategray"),
+     pch = 20, xlab = "grain (lbs plot-1)", ylab = "straw (lbs plot-1)")
> title(main = "Straw vs. grain, by field half")
> title(sub = "Interaction: solid lines; additive: dashed lines")
> abline(lm(straw ~ grain, subset = in.north), col = "blue")
> abline(lm(straw ~ grain, subset = !in.north), col = "slategray")
> abline(coefficients(model.straw.ns.grain)["(Intercept)"],
+     coefficients(model.straw.ns.grain)["grain"], col = "slategray",
+     lty = 2)
> abline(coefficients(model.straw.ns.grain)["(Intercept)"] +
+     coefficients(model.straw.ns.grain)["in.northTRUE"],
+     coefficients(model.straw.ns.grain)["grain"], col = "blue",
+     lty = 2)
```



Is this more complex interaction model significantly better than the additive model?

Task 56 : Compare the interaction and additive models by their adjusted R^2 and with an analysis of variance. •

```
> summary(model.straw.ns.grain)$adj.r.squared

[1] 0.61268

> summary(model.straw.ns.grain.i)$adj.r.squared

[1] 0.61305

> anova(model.straw.ns.grain.i, model.straw.ns.grain)

Analysis of Variance Table

Model 1: straw ~ in.north * grain
Model 2: straw ~ in.north + grain
  Res.Df   RSS  Df Sum of Sq    F Pr(>F)
1    496 154.9
2    497 155.3  -1      -0.5 1.47   0.23
```

Q66 : *Does the more complex model have a higher proportion of variance explained? Is this statistically significant?* *Jump to A66* •

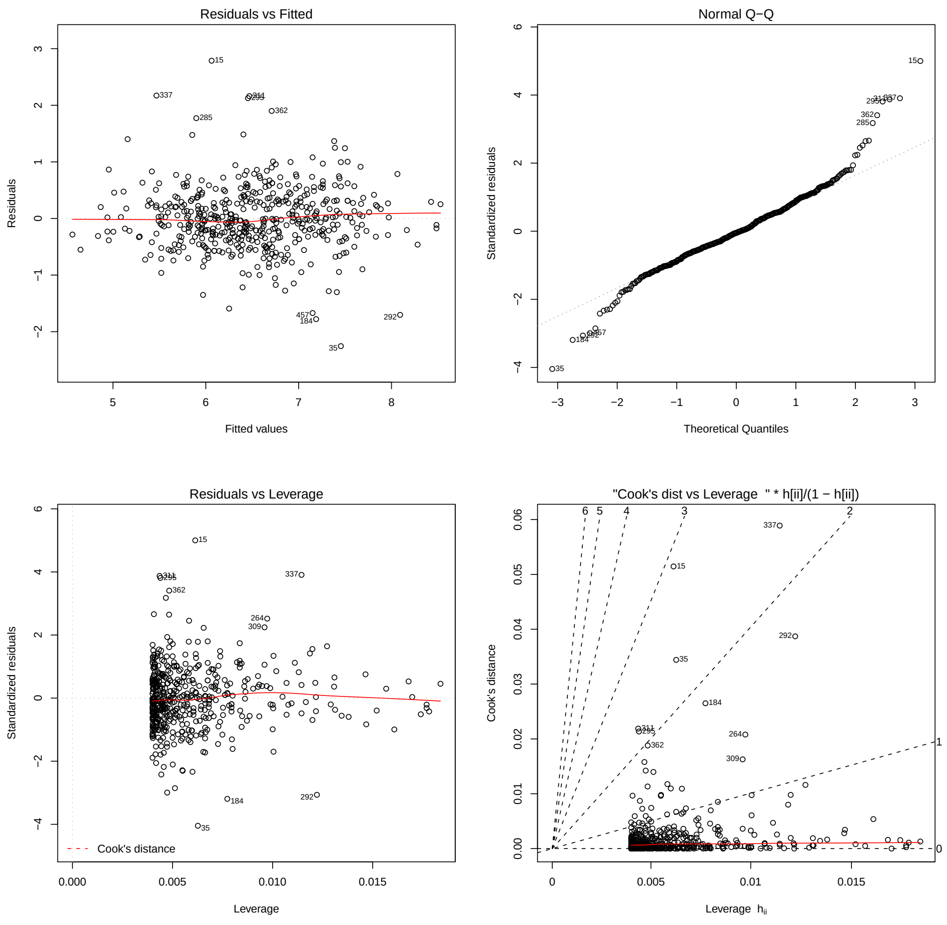
11.4 Regression diagnostics

As with univariate regression, multivariate regression models must be examined to see if they meet modelling assumptions.

Task 57 : Display the diagnostic plots for the additive model: (1) residuals vs. fits; (2) normal Q-Q plot of the residuals; (3) residuals vs. leverage; (4) Cook's distance vs. leverage. •

These are plot types 1, 2, 5, and 6, respectively, selected with the `which` optional argument to the `plot.lm` function. We also specify the number of extreme points to label with the `id.n` optional argument.

```
> par(mfrow = c(2, 2))
> plot.lm(model.straw.ns.grain, which = c(1, 2, 5, 6),
+     id.n = 10)
> par(mfrow = c(1, 1))
```



Q67 : *Which observations (plots) are marked on the plots as being potential problems?* Jump to A67 •

We can identify these numerically. First, we examine plots that were not well-modelled. We use the criterion of model residuals more than three standard deviations from zero; we want to see if there is any pattern to these.

We identify the plots with the large residuals, using the `rstandard` "standardized residuals" function, and show just these records in the data frame, using the `which` function to identify their row (record) numbers:

```
> (selected <- which(abs(rstandard(model.straw.ns.grain)) >
+     3))

 15  35 184 285 292 295 311 337 362
 15  35 184 285 292 295 311 337 362

> rstandard(model.straw.ns.grain)[selected]

     15      35     184     285     292     295     311     337
 5.0007 -4.0459 -3.1930  3.1776 -3.0646  3.8105  3.8741  3.9068
    362
 3.4074
```

Second, build a data frame with all the information for these plots, along with the residuals, using the `cbind` function to add a column:

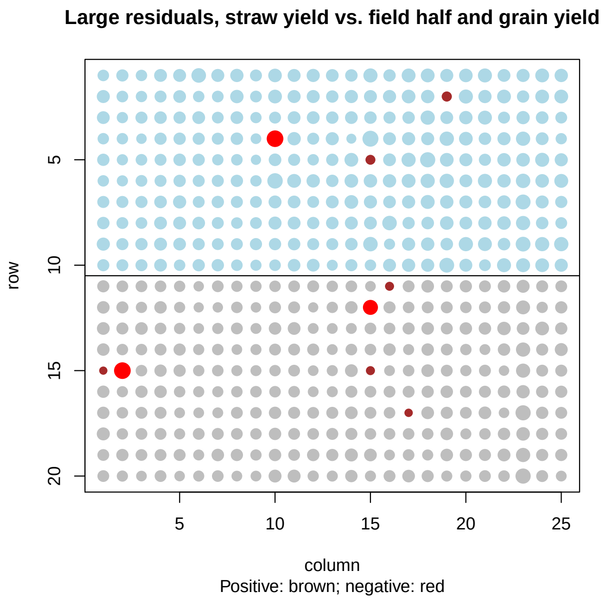
```
> mhw.hires <- cbind(mhw[selected, ], sres = rstandard(model.straw.ns.grain)[sele
> rm(selected)
> str(mhw.hires)

'data.frame':        9 obs. of  7 variables:
 $ r       : int  15 15 4 5 12 15 11 17 2
 $ c       : int  1 2 10 15 15 15 16 17 19
 $ grain   : num  3.46 4.42 4.59 3.7 4.86 3.73 3.74 3.05 4.26
 $ straw   : num  8.85 5.2 5.41 7.67 6.39 8.58 8.63 7.64 8.61
 $ gsr     : num  0.391 0.85 0.848 0.482 0.761 ...
 $ in.north: logi  FALSE FALSE TRUE TRUE FALSE FALSE ...
 $ sres    : num  5 -4.05 -3.19 3.18 -3.06 ...
```

Finally, order the selected plots by the residual, using the `order` function:

```
> mhw.hires[order(mhw.hires$sres), ]

     r  c grain straw     gsr in.north    sres
35  15  2  4.42  5.20 0.85000    FALSE -4.0459
184  4 10  4.59  5.41 0.84843     TRUE -3.1930
292 12 15  4.86  6.39 0.76056    FALSE -3.0646
285  5 15  3.70  7.67 0.48240     TRUE  3.1776
362  2 19  4.26  8.61 0.49477     TRUE  3.4074
295 15 15  3.73  8.58 0.43473    FALSE  3.8105
311 11 16  3.74  8.63 0.43337    FALSE  3.8741
337 17 17  3.05  7.64 0.39921    FALSE  3.9068
15  15  1  3.46  8.85 0.39096    FALSE  5.0007
```

We can also visualize the locations of these in the field: high positive residuals green, high negative residuals red, symbol size proportional to the grain/straw ratio:

```
> plot(c, r, ylim = c(20, 1), cex = 3 * gsr/max(gsr), pch = 20,
+     col = ifelse(rstandard(model.straw.ns.grain) > 3,
+         "brown", ifelse(rstandard(model.straw.ns.grain) <
+             (-3), "red", ifelse(in.north, "lightblue",
+             "gray"))), xlab = "column", ylab = "row")
> abline(h = 10.5)
> title(main = "Large residuals, straw yield vs. field half and grain yield")
> title(sub = "Positive: brown; negative: red")
```



Note: Note the use of the nested `ifelse` functions to select the four colours.

Q68 : *Are the high positive or negative residuals concentrated in one part of the field? Is there anything else unusual about these? Hint: look at the most negative and positive residuals.* *Jump to A68* •

The plot of leverage vs. Cook's distance shows which plots most affect the fit: high leverage and high distance means that removing that plot would have a large effect on the fit.

Q69 : *Do the two highest-residual plots identified in the previous question have high leverage? Which high-residual plots also have high leverage?* *Jump to A69* •

We can examine the effect of these on the fit by re-fitting the model, leaving out one or more of the suspect plots.

Task 58 : Fit the model without the two adjacent plots where we hypothesize sloppy field procedures and compare the goodness-of-fit and regression equations to the original model. •

To exclude some observations, we use the `-` operator on a list of dataframe row numbers created with the `c` function:

```
> model.straw.ns.grain.adj <- lm(straw ~ in.north + grain,
+     data = mhw[-c(15, 35), ])
> summary(model.straw.ns.grain.adj)

Call:
lm(formula = straw ~ in.north + grain, data = mhw[-c(15, 35),
    ])

Residuals:
    Min      1Q  Median      3Q     Max
-1.7926 -0.3106 -0.0274  0.3017  2.1942

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.9528     0.2094    4.55  6.8e-06 ***
in.northTRUE  -0.5118     0.0481  -10.64  < 2e-16 ***
grain          1.4731     0.0525   28.04  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.536 on 495 degrees of freedom
Multiple R-squared: 0.64,       Adjusted R-squared: 0.638
F-statistic:  440 on 2 and 495 DF,  p-value: <2e-16

> summary(model.straw.ns.grain)

Call:
lm(formula = straw ~ in.north + grain, data = mhw)

Residuals:
    Min      1Q  Median      3Q     Max
-2.2548 -0.3189 -0.0276  0.3042  2.7871

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.0461     0.2178     4.8  2.1e-06 ***
in.northTRUE  -0.5132     0.0500   -10.3  < 2e-16 ***
grain          1.4499     0.0546    26.5  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.559 on 497 degrees of freedom
Multiple R-squared: 0.614,       Adjusted R-squared: 0.613
F-statistic:  396 on 2 and 497 DF,  p-value: <2e-16
```

Q70 : *Does the model without the two plots fit the remaining plots better than the original model? How different are the model coefficients?* *Jump to A70* •

Challenge: Compare the diagnostic plots for the adjusted regression. What, if anything, has improved? Does the model now meet the assumptions of linear regression?

11.5 Analysis of covariance: a nested model

In the parallel-lines model there is only one regression line between the continuous predictor and predictand, which can be moved up and down according to different class means; this is an **additive** model. In the **interaction** model there is both an overall line and deviations from it according to class, allowing different slopes, as well as differences in class means.

Another way to look at this is to abandon the idea of a single regression altogether, and fit a separate line for each class. This is a **nested** model: the continuous predictor is measured only *within* each level of the classified predictor. There is no interest in the difference between classes (here, the field halves), only an interest in making the best fit in each half.

A nested model is specified with the / formula operator (this is *not* mathematical division). This is to be read as "fit the relation after the / separately for the two values of the classified variable".

```
> model.straw.ns.grain.nest <- lm(straw ~ in.north/grain,
+     data = mhw)
> summary(model.straw.ns.grain.nest)

Call:
lm(formula = straw ~ in.north/grain, data = mhw)

Residuals:
    Min      1Q  Median      3Q     Max
-2.2242 -0.3169 -0.0398  0.3136  2.7574

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)           1.2930     0.2979    4.34  1.7e-05 ***
in.northTRUE         -1.0375     0.4350   -2.39    0.017 *
in.northFALSE:grain   1.3872     0.0752   18.44  < 2e-16 ***
in.northTRUE:grain    1.5199     0.0794   19.14  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.559 on 496 degrees of freedom
Multiple R-squared: 0.615,	Adjusted R-squared: 0.613
F-statistic:  265 on 3 and 496 DF,  p-value: <2e-16
```

Note that the nested model does *not* give a difference between field halves; instead each half has its own regression line (intercept and slope).

Q71 : *What are the two slopes of straw vs. grain? Are they different? Do they differ from the single slope found in the parallel regression model?* *Jump to A71* •

Q72 : *This model has the same adjusted R-squared as the interaction model. Why?* *Jump to A72* •

Challenge: Compare the regression lines of this nested model with the regression lines implied by the interaction model. Are they the same? Why or why not?

11.6 Answers

A57 : Yes, straw yield most likely ($p \approx 0$) is higher in the South half; but this *explains very little (0.066) of the total variance.* *Return to Q57* •

A58 : *Yes, straw yield almost surely ($p \approx 0$) varies with grain yield; this explains about half (0.532) of the total variance.* *Return to Q58* •

A59 : *Both coefficients are highly significant (probability that they are really zero almost nil). Coefficient* `in.northTRUE` *represents the difference between field halves, and* `grain` *the regression between straw and grain.*

This explains over 60% (0.613) of the total variance. *Return to Q59* •

A60 : *Coefficient* `in.northTRUE` *represents the difference between field halves; the fitted value is -0.5132 lbs per plot. Coefficient* `grain` *is the regression between straw and grain; the fitted value is 1.4499 lbs straw increase per plot for each lb grain increase per plot. This is not the same as the best-fit univariate line (ignoring field half), which is 1.4305* *Return to Q60* •

A61 : *The multivariate additive model is clearly best; it explains about two-thirds (66%) of the variability, whereas the whole-field straw vs. grain model only explains just more than half (53%) and the field-half model very little (6%).* *Return to Q61* •

A62 : *Yes, in both cases the probability that we'd be wrong by rejecting the null hypothesis of no difference is practically zero. The RSS decreases from 375.4 for the field-half model, and 188.2 for the whole-field straw vs. grain model, to 155.3 for the combined model. That is, much less variability in straw yield remains unexplained after the combined model.* *Return to Q62* •

A63 : *The parallel regression models the case where one half of the field is more productive, on average, than the other, but the relation between grain and straw is the same in both cases (i.e. the grain/straw ratio is the same in both field halves). There is no difference in plant morphology, just overall size.*

By contrast, the interaction model allows that the two field halves may also differ in the grain/straw ratio, i.e. the relation between grain and straw yield – different plant morphology. *Return to Q63* •

A64 : *The coefficients are:*

1. `(Intercept)`*: estimated straw yield at* ***zero*** *grain yield and in the S field half;*
2. `in.northTRUE`*:* ***difference*** *in* ***average*** *yield in the N vs. the S;*
3. `grain`*:* ***increase*** *in straw yield for each unit increase in grain yield, in the S field half;*
4. `in.northTRUE:grain`*:* ***difference*** *in* ***slope*** *(increase in straw yield for each unit increase in grain yield) in the N vs. the S.*

Return to Q64 •

A65 : *The coefficient* `in.northTRUE:grain` *is not significant; the probabilty of falsely rejecting the null hypothesis is quite high, 0.2255, so we should accept the hypothesis that this difference is really 0.*

According to this test, the interaction is not significant. Plants grow with the same morphology in the two field halves. Return to Q65 •

A66 : *The interaction model has slightly higher adjusted R^2: 0.61305 vs. 0.61268. However, the ANOVA table shows that this increase has a* `p`*=0.225 probability of occurring by chance, so we conclude the interaction model is not justified. This is the same conclusion we reached from the model summary, where the interaction term (coefficient) was not significant.* Return to Q66 •

A67 : *Plots labelled on the diagnostic graphs are 15, 337, 311, 295, 362, 285 (positive residuals) and 35, 184, 292, 457 (negative residuals). These have residuals more extreme than expected by theory (normal Q-Q plot of residuals). Plots 292, 337, 264, 184 and 309 are large residuals with high leverage.* Return to Q67 •

A68 : *The highest four positive residuals are all in the S half, but otherwise do not seem clustered. The most negative residual is from plot 35 (r=15, c=2) and the most positive from the immediately adjacent plot 15 (r=15, c=1). Could some of the grain from plot 15 have been accidentally measured as part of the yield of plot 35? If these two are combined, the grain/straw ratio is 0.56085, close to the mean grain/straw ratio of the whole field, 0.61054.* Return to Q68 •

A69 : *Plots 15 and 35 do not have high leverage, i.e. their removal would not change the equation very much. The high-leverage plots that also have high residuals are 292 amd 184 (negative residuals) and 337, 264 and 309 (positive residuals).* Return to Q69 •

A70 : *The fit is considerably better without these badly-modelled plots: 0.63841 without the two plots vs. 0.61268 with them. Another 2.5% of the variation is explained.*

The coefficient for field half hardly changes, but the regression line changes substantially: higher intercept: (0.9528 vs. 1.0461) and shallower slope (1.4731 vs. 1.4499 . As predicted by the high leverage, removing these points changes the functional relation. Return to Q70 •

A71 : *The slope in the S half is 1.3872, in the N half 1.5199. These differ considerably from each other, and from the parallel regression slope: 1.4499. The slope in the S half is less steep, in the N half steeper, than the parallel regression slope.* Return to Q71 •

A72 : *Both models have four parameters to fit the same dataset. Both model* *difference between levels (either means or intercepts) and slopes.* *Return to Q72* •

We are done with these models and some other variables, so clean up the workspace:

```
> rm(model.straw.ns, model.straw.grain, model.straw.ns.grain,
+     model.straw.ns.grain.adj, model.straw.ns.grain.i,
+     model.straw.ns.grain.nest)
> rm(struct.beta, beta, mhw.hires)
```

12 Spatial analysis

The remainder of this tutorial goes well beyond what is required to understand R. It is included here because of the intrinsic interest of the Mercer & Hall dataset. It also shows how to load and work with **add-in packages** that are not part of the base R distribution.

To this point we have only considered the wheat yields in **feature** space (also known as **attribute** or **property** space). For example, the grain and straw yields form a two-dimensional 'space'. But we have ignored an additional piece of information: the **relative location** of the plots in the field. Mercer & Hall clearly stated that there could be hot spots or geographic trends, meaning that the plots are not necessarily **spatially independent**. We now investigate this.

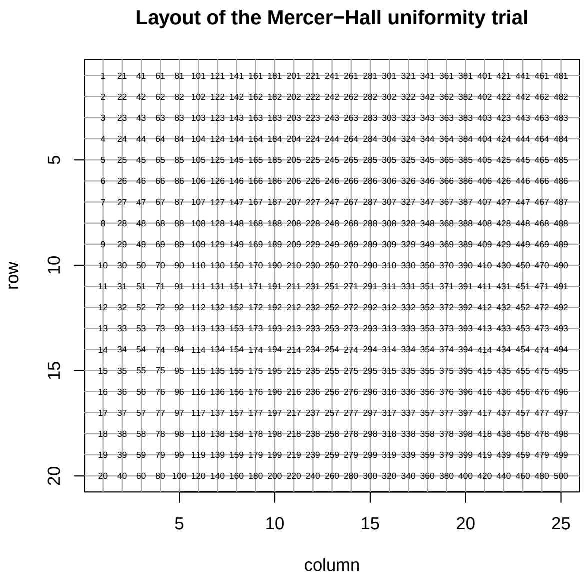
12.1 Geographic visualisation

We begin by visualisaing the agricultural field:

Task 59 : Make a correctly-scaled map of the plot locations in the field, showing the plot numbers. •

The plots are rectangular (longer N-S than wide E-W), so that by plotting the 25 columns and 20 rows in a square (the shape of the field, and the default for a bivariate `plot`), we get a geometrically-correct map. However, the default plotting function is from low to high indices, so that row 1 would be plotted at the bottom, when in fact it is at the top. We can specify the axis with the `ylim` argument, reversing the row order:

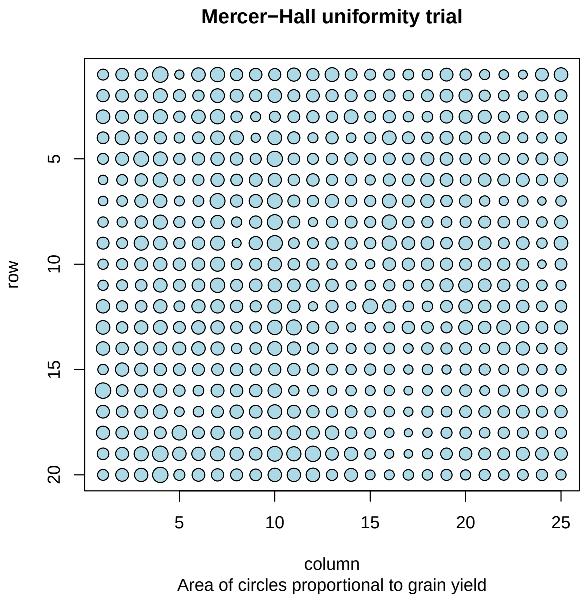
```
> plot(c, r, type = "n", xlab = "column", ylab = "row",
+     ylim = c(20, 1), main = "Layout of the Mercer-Hall uniformity trial")
> abline(v = 1:25, lty = 1, col = "darkgray")
> abline(h = 1:20, lty = 1, col = "darkgray")
> text(c, r, rownames(mhw), cex = 0.5)
```



Note the use of the `xlab` and `ylab` graphics parameter to specify the axis names.

Task 60 : Make a **post-plot** of the grain yield, i.e. a map of the plot locations with symbol size proportional to the data value. •

```
> plot(c, r, pch = 21, col = "black", bg = "lightblue",
+     ylim = c(20, 1), xlab = "column", ylab = "row", main = "Mercer-Hall uniform
+     sub = "Area of circles proportional to grain yield",
+     cex = 2 * grain/max(grain))
```



We can visualise this better by displaying each point in a colour ramp. First we classify the observations into octiles (eight groups) with the `cut` function, using the quantile function to compute the octiles.

```
> (q8 <- quantile(grain, seq(0, 1, length = 9)))

    0%  12.5%    25%  37.5%    50%  62.5%    75%  87.5%   100% 
2.7300 3.4238 3.6375 3.7812 3.9400 4.0900 4.2700 4.4700 5.1600 

> grain.c <- cut(grain, q8, include.lowest = T, labels = F)
> sort(unique(grain.c))

[1] 1 2 3 4 5 6 7 8
```

So the 500 yields have been grouped into eight classes.

A **colour ramp** is a list of colours in some visually-meaningful sequence. One example is produced by the terrain.colors function; the colours are given as hexidecimal numbers from 00 (absence of colour) to FF (saturation with the colour), for the three primary colours Red, Green and Blue:

```
> terrain.colors(8)

[1] "#00A600FF" "#3EBB00FF" "#8BD000FF" "#E6E600FF" "#E9BD3AFF"
[6] "#ECB176FF" "#EFC2B3FF" "#F2F2F2FF"
```

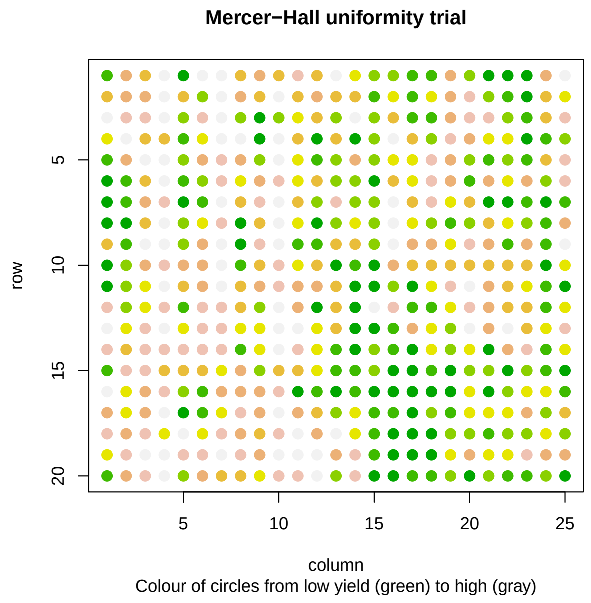
For example, the final colour in this ramp is #F2F2F2, which is a dark gray: equal saturations of the three primaries, and each of these has #F2/#FF, i.e. 95% saturation:

```
> 0xf2/0xff

[1] 0.94902
```

This example shows that hexidecimal numbers may be used in R; they are indicated with the prefix 0x.

Now we use this colour ramp, selecting the appropriate colour for each quantile:

```
> plot(c, r, pch = 20, cex = 2, bg = "lightblue", ylim = c(20,
+     1), xlab = "column", ylab = "row", main = "Mercer-Hall uniformity trial",
+     sub = "Colour of circles from low yield (green) to high (gray)",
+     col = terrain.colors(8)[grain.c])
```

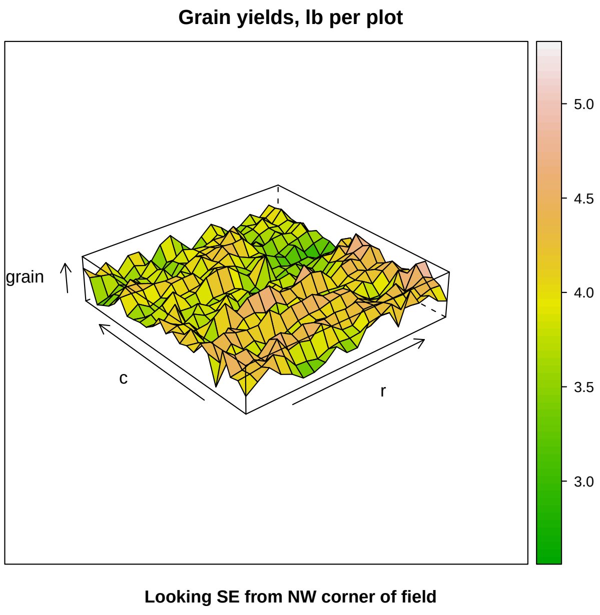


Finally, we remove the temporary variables:

```
> rm(q8, grain.c)
```

Another way to visualize the field is with a **3D plot** using the `wireframe` graphics function of the `lattice` package. The optional `aspect` argument controls the vertical exaggeration:

```
> plot(wireframe(grain ~ r + c, data = mhw, drape = T,
+     aspect = c(1, 0.2), col.regions = terrain.colors(128),
+     main = "Grain yields, lb per plot", sub = "Looking SE from NW corner of fie
```



Q73 : *Does there appear to be* ***local spatial dependence****, i.e. similar* *values near each other? Does this appear to vary over the field?* *Jump to A73* •

12.2 Setting up a coördinate system

Analysis of the **spatial structure** requires metric coördinates, rather than row and colum numbers, since the plots are not square.

Task 61 : Determine the field size and the plot dimensions. •

The original experiment was in English units, but we will use metres. So, we start with some conversions to get the dimensions of each plot:

First, some conversion factors; note that 1 ha = 10 000 m^2:

```
> ha2ac <- 2.471054
> ft2m <- 0.3048
> (field.area <- 10000/ha2ac)

[1] 4046.9
```

Then we divide the side of the 1-acre field evenly into 20 rows and 25 columns to obtain the dimensions in meters and the area in m^2:

```
> (plot.area <- field.area/500)

[1] 8.0937

> (plot.len <- sqrt(field.area)/20)

[1] 3.1807

> (plot.wid <- sqrt(field.area)/25)

[1] 2.5446

> rm(ha2ac, ft2m, field.area)
```

Task 62 : Compute the total length and width in metres, confirm they are equal (because the field is a square, and confirm that they multiply to 1 ha (4 045.9 m^2). •

```
> (tot.len <- plot.len * 20)

[1] 63.615

> (tot.wid <- plot.wid * 25)

[1] 63.615

> tot.len * tot.wid

[1] 4046.9

> rm(tot.len, tot.wid)
```

Task 63 : Compute coördinates for the centre of each plot. •

Coördinates are assigned from an arbitrary origin of $(0, 0)$ at the SW corner of the field, so that the coördinates of the centre of plot $[1, 1]$ are half the plot size in both directions:

```
> plot.wid/2

[1] 1.2723

> plot.len/2

[1] 1.5904
```

Now we build a data frame of coordinates; first with the `seq` function to make vectors of the midpoints of the E and N directions, respectively; and then with the `expand.grid` function to make a dataframe with one row per combination:

```
> nrow <- length(unique(r))
> ncol <- length(unique(c))
> sx <- seq(plot.wid/2, plot.wid/2 + (ncol - 1) * plot.wid,
+     length = ncol)
> sy <- seq(plot.len/2 + (nrow - 1) * plot.len, plot.len/2,
+     length = nrow)
> xy <- expand.grid(x = sx, y = sy)
> rm(nrow, ncol, sx, sy)
```

Note the order of the y sequence: starting with the highest coordinate for row 1 (which is at the top of the plot).

We keep `plot.wid`, `plot.len`, and `plot.area` to be used later.

12.3 Loading add-in packages

Most of the analysis from here on will use two **add-in packages**; these are representative of the hundreds which have been implemented by practising statisticians. Here we will use `sp` package [17], which is a foundation for spatially-explicit analysis in R, and the `gstat` package [16], which is an R implementation of the **gstat** geostatistics program [18]. Both of these are extensively discussion and illustrated in the textbook "Applied Spatial Data Analysis with R" by Bivand et al. [1].

When R starts, a number of basic packages are loaded; we can see these with the `search` function:

```
> search()

 [1] ".GlobalEnv"        "mhw"                "package:MASS"
 [4] "package:stats"     "package:graphics"   "package:grDevices"
 [7] "package:utils"     "package:datasets"   "package:lattice"
[10] "package:methods"   "Autoloads"          "package:base"
```

Additional packages must be **loaded** with the `library` function.

Task 64 : Load the sp and gstat packages. •

```
> library(sp)
> library(gstat)
> search()
```

```
 [1] ".GlobalEnv"        "package:gstat"     "package:sp"
 [4] "mhw"               "package:MASS"      "package:stats"
 [7] "package:graphics"  "package:grDevices" "package:utils"
[10] "package:datasets"  "package:lattice"   "package:methods"
[13] "Autoloads"         "package:base"
```

12.4 Creating a spatially-explicit object

The sp package adds a number of **spatial data types**, i.e. new **object classes**.

Task 65 : Copy the mhw dataframe to a new object and convert the copy to class SpatialPointsDataFrame. •

By making a copy we have the data in two forms, spatial and non-spatial, so we don't have to keep converting between them.

We do this by adding the computed coordinates to the data frame with the coordinates function; this automatically converts to the spatial data type defined by the sp package:

```
> mhw.sp <- mhw
> coordinates(mhw.sp) <- xy
> rm(xy)
> summary(mhw.sp)
```

```
Object of class SpatialPointsDataFrame
Coordinates:
     min    max
x 1.2723 62.343
y 1.5904 62.025
Is projected: NA
proj4string : [NA]
Number of points: 500
Data attributes:
       r               c          grain           straw
 Min.   : 1.00   Min.   : 1   Min.   :2.73   Min.   :4.10
 1st Qu.: 5.75   1st Qu.: 7   1st Qu.:3.64   1st Qu.:5.88
 Median :10.50   Median :13   Median :3.94   Median :6.36
 Mean   :10.50   Mean   :13   Mean   :3.95   Mean   :6.51
 3rd Qu.:15.25   3rd Qu.:19   3rd Qu.:4.27   3rd Qu.:7.17
 Max.   :20.00   Max.   :25   Max.   :5.16   Max.   :8.85
      gsr          in.north
 Min.   :0.391   Mode :logical
 1st Qu.:0.574   FALSE:250
 Median :0.604   TRUE :250
 Mean   :0.611   NA's :0
 3rd Qu.:0.642
 Max.   :0.850
```

Q74 : *What is the data type of the* `mhw.sp` *object? What is the* ***bounding box****, i.e. limits of the the coördinates?* *Jump to A74* •

Now that we've built the spatial object, we can save it for later use in another session:

```
> save(mhw.sp, file = "mhw_spatial.RData")
```

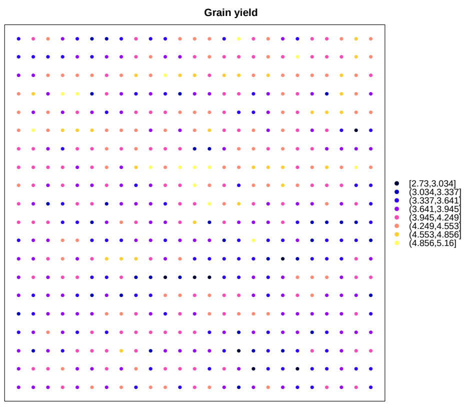
12.5 More geographic visualisation

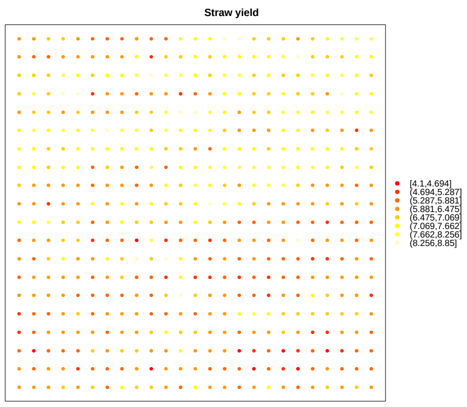
Once an object is in a spatial class, the `spplot` function can be used to make a nicely-coloured post plot:

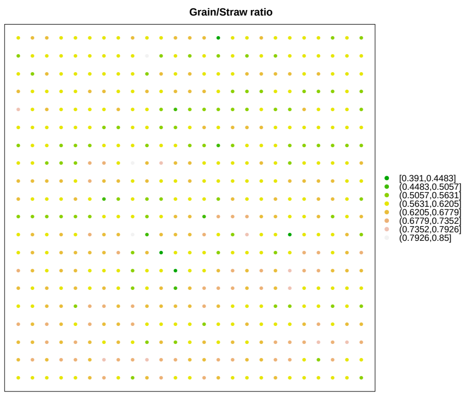
Task 66 : Plot the grain, straw, and their ratio, coloured by their octile. •

We take this chance to illustrate some more colour ramps, produced by the `bpy.colors` and `heat.colors` functions, as well as the `terrain.colors` function we saw before. To put several plots on the same page, we create each plots with `spplot` and save it in a local variable; we then use the generic `print` function, with the optional `more` argument:

```
> f1 <- spplot(mhw.sp, zcol = "grain", cuts = 8, col.regions = bpy.colors(8),
+     main = "Grain yield", key.space = "right")
> f2 <- spplot(mhw.sp, zcol = "straw", cuts = 8, col.regions = heat.colors(8),
+     main = "Straw yield", key.space = "right")
> f3 <- spplot(mhw.sp, zcol = "gsr", cuts = 8, col.regions = terrain.colors(8),
+     main = "Grain/Straw ratio", key.space = "right")
> print(f1, split = c(1, 1, 2, 2), more = T)
> print(f2, split = c(2, 1, 2, 2), more = T)
> print(f3, split = c(1, 2, 2, 2), more = F)
> rm(f1, f2, f3)
```

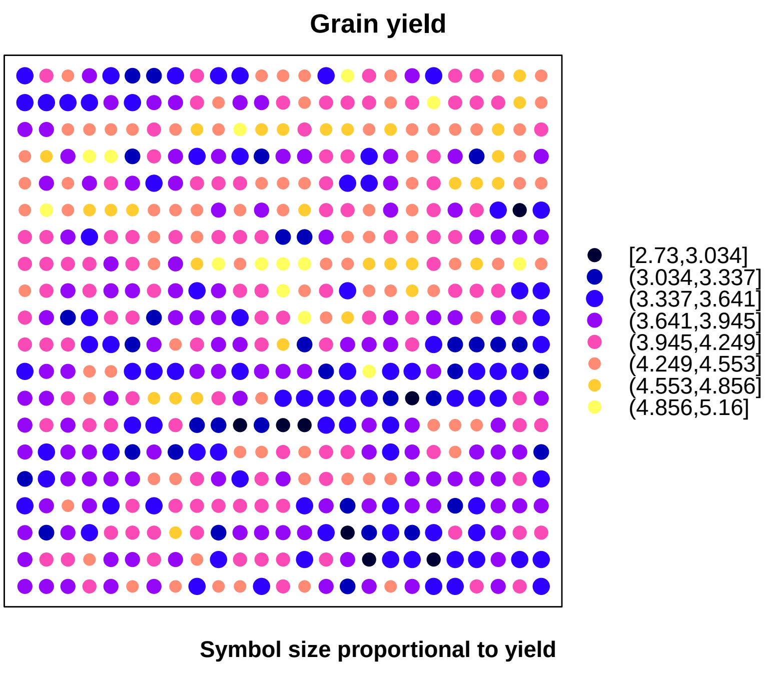






And we can make a postplot with both colour and size:

```
> print(spplot(mhw.sp, zcol = "grain", cuts = 8, cex = 1.6 *
+     grain/max(grain), col.regions = bpy.colors(8), pch = 19,
+     main = "Grain yield", sub = "Symbol size proportional to yield",
+     key.space = "right"))
```



12.6 Answers

A73 : *There are clear clusters of similar values, e.g. the patch of low values centred near (16, 18). Clusters seem more obvious in the north than the south.* *Return to Q73* •

A74 : `SpatialPointsDataFrame`; *the bounding box is 1.2723 to 62.3426 (W-E) and 1.5904 to 62.0245 (S-N)* *Return to Q74* •

13 Spatial structure

Now that we have a spatially-explicit object, we can examine it for its **spatial structure**. This can be of two kinds: a **trend** across the entire area or a **local structure** that does not depend on absolute location.

13.1 Spatial structure: trend

One possibility for spatial structure is a **trend** across the field.

Task 67 : Explore whether there is any trend in grain yield by row or column. •

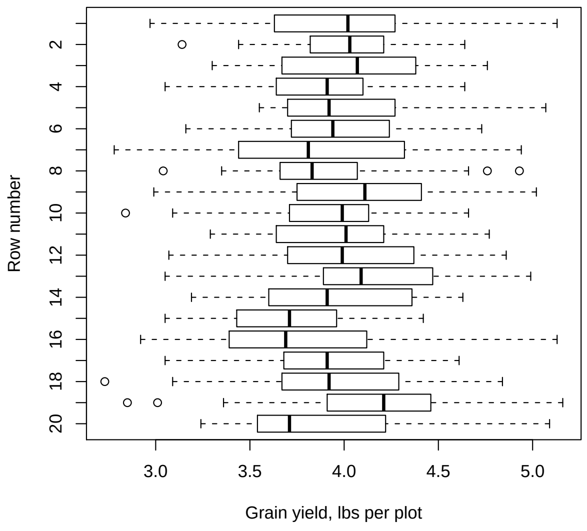
One way to do this is to compute the row and column mean yields, and then sort them from lowest to highest with the `sort` function:

```
> sort(by(grain, r, mean), decreasing = FALSE)

r
    15     16      7      8     20     17     10      4     18     14
3.7072 3.7432 3.8708 3.8796 3.8816 3.9012 3.9136 3.9144 3.9164 3.9352
     6     11      1     12      2      5      3      9     13     19
3.9536 3.9540 3.9576 3.9848 4.0072 4.0276 4.0420 4.0788 4.1160 4.1880

> sort(by(grain, c, mean), d = F)

c
    17     15     24     14     18     22     23      5     21     16
3.5280 3.6075 3.6565 3.7390 3.7545 3.7585 3.7925 3.8150 3.8165 3.8635
    19     12     13      1      9     25      8      2     20      6
3.8740 3.8955 3.8970 3.9165 3.9420 3.9450 3.9635 3.9650 4.0025 4.0570
    11      3      7     10      4
4.1125 4.2820 4.4630 4.5280 4.5410
```

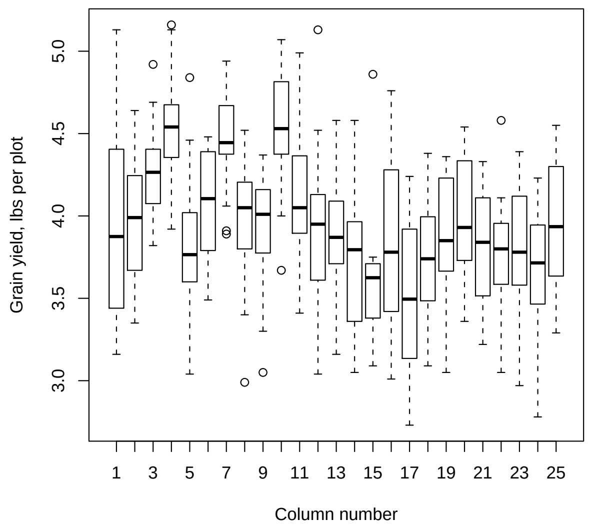
Q75 : *Does there appear to be any trend or pattern in the sequence of row or column numbers?* *Jump to A75* •

We can see both the means and variability with a grouped boxplot, first by row and then by column. Note the use of the `xlim` argument to display the row boxplots in correcg geographical order.

```
> boxplot(grain ~ r, horizontal = T, data = mhw, xlim = c(20,
+     1), ylab = "Row number", xlab = "Grain yield, lbs per plot")
```



```
> boxplot(grain ~ c, data = mhw, xlab = "Column number",
+     ylab = "Grain yield, lbs per plot")
```



Q76 : *Does there appear to be any pattern by row or column, either in the median yield or the variability with each row or column?* *Jump to A76* •

Although only the columns show a slight trend, there could be a trend not oriented with these.

Task 68 : Compute a **first-order trend surface** of grain yield. •

```
> ts1 <- lm(grain ~ coordinates(mhw.sp))
> summary(ts1)

Call:
lm(formula = grain ~ coordinates(mhw.sp))

Residuals:
    Min      1Q  Median      3Q     Max 
-1.1352 -0.2936  0.0069  0.3140  1.1711 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)           3.731260   0.051850   71.96  < 2e-16 ***
coordinates(mhw.sp)x -0.000664   0.001067   -0.62     0.53    
coordinates(mhw.sp)y  0.007498   0.001068    7.02  7.2e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.438 on 497 degrees of freedom
Multiple R-squared: 0.0909,	Adjusted R-squared: 0.0873 
F-statistic: 24.9 on 2 and 497 DF,  p-value: 5.14e-11
```

Q77 : *Is there a significant trend? How much of the variance is explained?* *Jump to A77* •

This is not a promising approach, so we remove the trend surface object from the workspace:

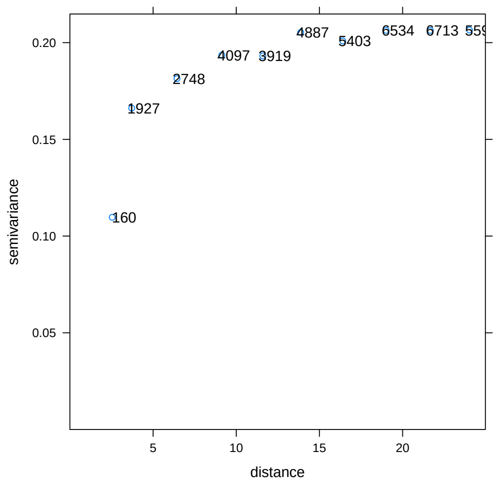
```
> rm(ts1)
```

13.2 Spatial structure: local

There is only a very weak trend; but are there hotspots?

We use the `variogram` function of the `gstat` package to analyze the local spatial structure. We also specify the optional `plot.numbers = T` argument to print the number of point-pairs next to the variogram values; the optional `width` argument to specify the bin size (here, the plot width), and the optional `cutoff` argument (by default it is 1/3 of the largest distance between point pairs); here it is 10 plot widths.

```
> v <- variogram(grain ~ 1, mhw.sp, cutoff = plot.wid *
+     10, width = plot.wid)
> print(plot(v, plot.numbers = T))
```

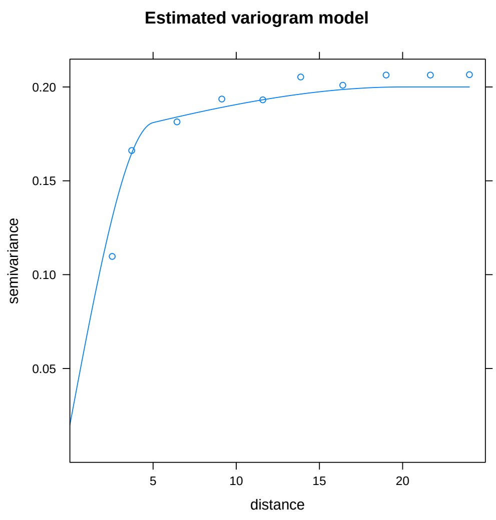


Q78 : *Describe the shape of the variogram. Is there evidence of local spatial structure? What is the approximate* **range** *of local spatial dependence, i.e. the separation at which the experimental variogram reaches its* **sill** *(maximum)?* Jump to A78
•

Q79 : *Across how many adjacent plots is there expected to be some spatial dependence?* Jump to A79 •

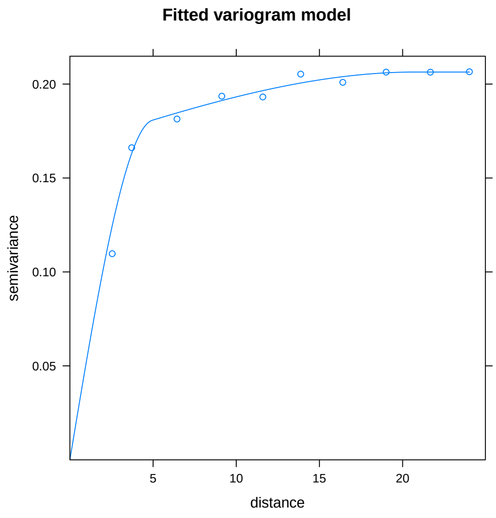
We now try to fit a theoretical variogram model to this empirical variogram. There appear to be two structures: a short-range to about 5 m with rapid increase in semivariance with separation, and then a gradual increase to a sill. So we use the the `vgm` (specify a variogram) function twice, with the `add.to` argument the second time to combine variogram models:

```
> (vm <- vgm(0.15, "Sph", 5, 0.02))

  model psill range
1   Nug  0.02     0
2   Sph  0.15     5

> (vm <- vgm(0.03, "Sph", 20, add.to = vm))

  model psill range
1   Nug  0.02     0
2   Sph  0.15     5
3   Sph  0.03    20

> print(plot(v, model = vm, main = "Estimated variogram model"))
```



We then adjust the variogram with the `fit.variogram` function:

```
> (vmf <- fit.variogram(v, vm))

  model    psill   range
1   Nug 0.000000  0.0000
2   Sph 0.166884  4.9488
3   Sph 0.039522 20.8112

> print(plot(v, model = vmf, main = "Fitted variogram model"))
```



The fit has reduced the nugget to zero; so at each point (inside a plot) there should be, as theory predicts, no spatial dependence. The nested structure clearly suggests that most of the spatial variability is within the first 5 m, so grouping a few plots should greatly reduce the between-plot variability of an experiment (Mercer & Hall's objective).

13.3 Answers

A75 : *The row and column numbers don't seem to show any pattern or trend.* Return to Q75 •

A76 : *Although there are differences among rows and columns, there does not appear to be a trend. The higher-numbered columns (East half of the field) appear to be slightly lower (as a group) than the lower-numbered columns. There appears to be some short-range periodicity at a one-plot range (high followed by low) in both dimensions, although this is not regular.* Return to Q76 •

A77 : *Only about 9% of the variance is explained; only the x-coördinate (E-W, across the columns from low- to high-numbered) is significant; it shows a slight trend towards the East; this agrees with the by-column boxplot. This trend was also noted by Patankar [15].* Return to Q77 •

A78 : *There is evidence of spatial structure: the semivariance increases up to about 13 m separation; the seminvariance then fluctuates around a total sill of about* $\gamma = 0.21\text{lb}^2$ *There appears to be a "nugget" effect (semivariance at zero separation) of about* $\gamma = 0.05\text{lb}^2$. Return to Q78 •

A79 : *Plot size is (3.18 m long x 2.55 m wide) (§A), so the range of about 13 m corresponds to about four adjacent plots column-wise and five adjacent plots row-wise.* Return to Q79
•

14 Spatial structure of field halves

In §10 we computed an indicator variable to show which half of the field each plot is in. In a spatial analysis we may now ask whether these two halves have different spatial structures.

Task 69 : Separate the dataset into two halves, one for each field half. •

To get a suitable data structure we use the `split` function to create one object with a list of two data frames, one for each half. We then assign their coordinates.

```
> mhw.sp.ns <- split(as.data.frame(mhw.sp), in.north)
> coordinates(mhw.sp.ns$T) <- ~x + y
> coordinates(mhw.sp.ns$F) <- ~x + y
> summary(mhw.sp.ns)
```

```
      Length Class                  Mode
FALSE 8      data.frame             list
TRUE  8      data.frame             list
T     1      SpatialPointsDataFrame S4
F     1      SpatialPointsDataFrame S4
```

```
> summary(mhw.sp.ns$T)
```

```
Object of class SpatialPointsDataFrame
Coordinates:
     min    max
x 1.2723 62.343
y 1.5904 62.025
Is projected: NA
proj4string : [NA]
Number of points: 250
Data attributes:
       r               c          grain           straw
 Min.   : 1.0   Min.   : 1   Min.   :2.78   Min.   :4.10
 1st Qu.: 3.0   1st Qu.: 7   1st Qu.:3.66   1st Qu.:5.73
 Median : 5.5   Median :13   Median :3.97   Median :6.14
 Mean   : 5.5   Mean   :13   Mean   :3.96   Mean   :6.28
 3rd Qu.: 8.0   3rd Qu.:19   3rd Qu.:4.27   3rd Qu.:6.86
 Max.   :10.0   Max.   :25   Max.   :5.13   Max.   :8.64
      gsr          in.north
 Min.   :0.482   Mode:logical
 1st Qu.:0.596   TRUE:250
 Median :0.635   NA's:0
 Mean   :0.636
 3rd Qu.:0.669
 Max.   :0.848
```

Task 70 : Compute the variograms for each half, and plot these along with the combined variogram. •

We first compute the variograms for the two field halves; we already have the variogram for the entire field.

```
> v.n <- variogram(grain ~ 1, mhw.sp.ns$T, cutoff = 30)
> v.s <- variogram(grain ~ 1, mhw.sp.ns$F, cutoff = 30)
```

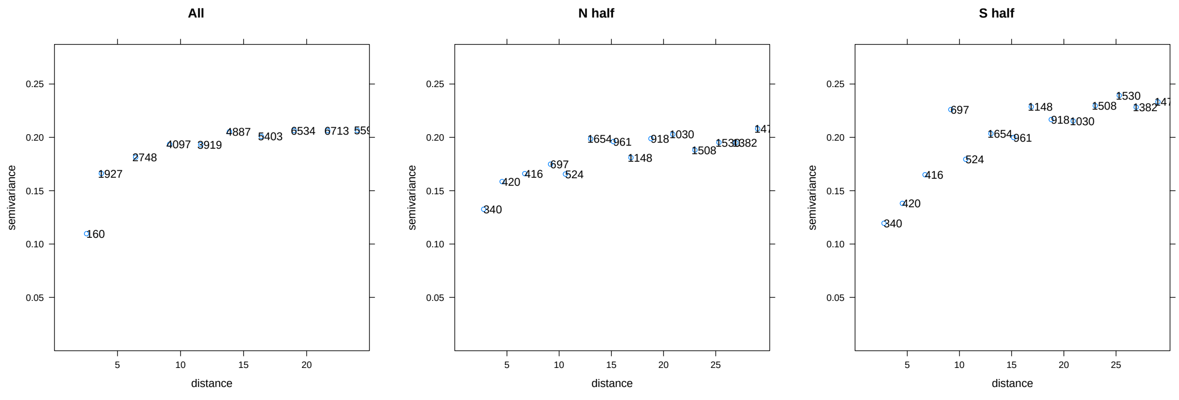
We now compute the figures, but do not print them right away; instead we store them as plotting objects:

```
> g.max = max(v$gamma, v.n$gamma, v.s$gamma) * 1.2
> plot.vgm.all <- plot(v, plot.numbers = T, main = "All",
+     ylim = c(0, g.max))
> plot.vgm.N <- plot(v.n, plot.numbers = T, main = "N half",
+     ylim = c(0, g.max))
> plot.vgm.S <- plot(v.s, plot.numbers = T, main = "S half",
+     ylim = c(0, g.max))
```

Note how we compute a common vertical axis from the maximum value of all three variograms, so we can compare them side-by-side.

Now we print these on one screen, specifying their positions in the plot:

```
> print(plot.vgm.all, split = c(1, 1, 3, 1), more = T)
> print(plot.vgm.N, split = c(2, 1, 3, 1), more = T)
> print(plot.vgm.S, split = c(3, 1, 3, 1), more = F)
```

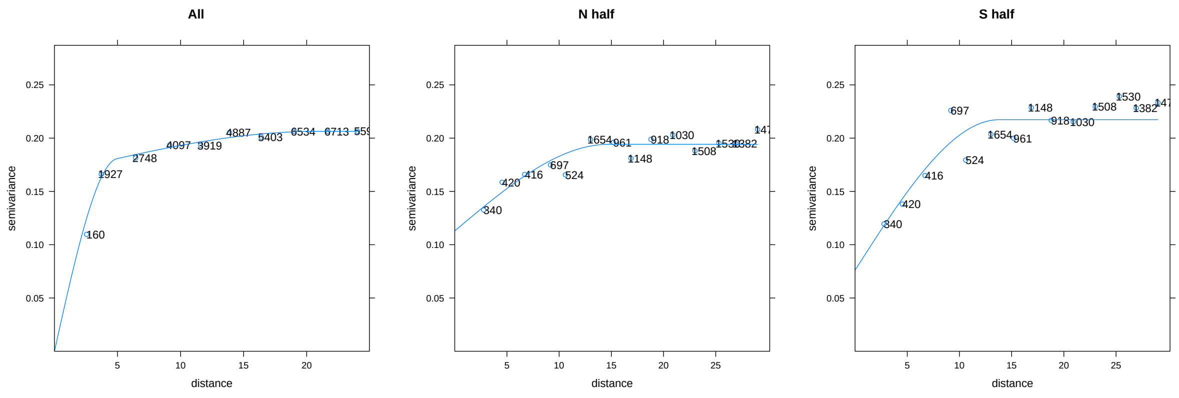


We now try to model the half-field variograms, as we did for the whole field in the previous section. The half-field variogams do not seem to show the nested structure of the whole-field variogram.

```
> (vmS <- vgm(0.14, "Sph", 20, 0.09))

  model psill range
1   Nug  0.09     0
2   Sph  0.14    20

> (vmN <- vgm(0.08, "Sph", 13, 0.11))

  model psill range
1   Nug  0.11     0
2   Sph  0.08    13

> (vmSf <- fit.variogram(v.s, vmN))

  model    psill  range
1   Nug 0.076024  0.000
2   Sph 0.141284 13.787
```

```
> (vmNf <- fit.variogram(v.n, vmS))

  model    psill  range
1   Nug 0.112956  0.000
2   Sph 0.081287 14.747

> plot.vgm.all <- plot(v, plot.numbers = T, main = "All",
+     model = vmf, ylim = c(0, g.max))
> plot.vgm.N <- plot(v.n, plot.numbers = T, main = "N half",
+     model = vmNf, ylim = c(0, g.max))
> plot.vgm.S <- plot(v.s, plot.numbers = T, main = "S half",
+     model = vmSf, ylim = c(0, g.max))
```

```
> print(plot.vgm.all, split = c(1, 1, 3, 1), more = T)
> print(plot.vgm.N, split = c(2, 1, 3, 1), more = T)
> print(plot.vgm.S, split = c(3, 1, 3, 1), more = F)
```



Remove the plotting objects and scale:

```
> rm(g.max, plot.vgm.all, plot.vgm.N, plot.vgm.S)
```

Q80 : *Do the two halves appear to have different local spatial structure?*

Jump to A80 •

Remove the variograms and models from the workspace:

```
> rm(v, v.n, v.s, vm, vmf, vmN, vmNf, vmS, vmSf)
```

We are also done with the field halves:

```
> rm(mhw.sp.ns)
```

14.1 Answers

A80 : *There is a big difference between the structures in the two halves. The S half is more variable overall (higher total sill, about $\gamma = 0.22\mathrm{lb}^2$), with a longer range around 20 m; the N half reaches a lower total sill (about $\gamma = 0.19\mathrm{lb}^2$) at a* *shorter range, about 12 m. Both have a nugget effect, about* $\gamma = 0.075 \text{lb}^2$ *in the S and* $\gamma = 0.011 \text{lb}^2$ *in the N.* *Return to Q80* •

15 The effect of plot size

Mercer and Hall's original research objective was to determine how within-plot variability is affected by plot size. To investigate this, they grouped the 1 acre field into plots of 1/500 acre (the original plots), 1/250 acre, 1/125 acre, 1/50 acre, 1/25 acre, and finally 1/10 acre; they measured the variability in the resulting samples and graphed this by plot size. They then could determine how large a plot would be necessary to reduce the variability to an acceptable level. We will repeat their analysis here, with the benefit of technology they could not have imagined in 1911.

Q81 : *Based on the geostatistical analysis (§13.2), what size plot would be expected to remove most of the* ***local*** *variation?* *Jump to A81* •

Before proceeding, we need an operational definition of heterogeneity, so we can compare the variability between different plot sizes. In §8 we used the **probable error**, but that required modelling a normal distribution. A simpler, and commonly-used, measure of variability for samples that are approximately normally-distributed is the *coefficient of variation* (CV), which is defined as:

$$\text{CV} = s/\bar{x}$$

This normalizes the sample standard deviation s by the sample mean $\bar{x}$. It is commonly expressed in percent. This measure was also used by Mercer and Hall[9].

There is no R function to compute the CV; we can compute it on any sample by first using the `mean` function and then the `sd` function, and then taking their ratio:

```
> round(100 * sd(mhw$grain)/mean(mhw$grain), 1)
```

```
[1] 11.6
```

To avoid writing this out for each vector whose CV we want to compute, we can write a small **function** to do this computation on any sample; this uses the `function` function to define the algorithm.

Task 71 : Write a **function** to compute the CV of a vector. •

```
> cv <- function(x) {
+     round(100 * sd(x)/mean(x), 1)
+ }
> cv
```

[9] although they mistakenly refer to it as "standard deviation"

```
function (x)
{
    round(100 * sd(x)/mean(x), 1)
}

> class(cv)

[1] "function"
```

The object `cv` in the workspace is a function which can now be applied to any vector, just like a built-in R function.

Q82 : *What is the CV of the grain yields of the entire set of 500 plots?* *Jump to A82* •

```
> cv(mhw$grain)

[1] 11.6
```

Now we group the plots into increasingly-larger plots and see how the CV of the set is affected. Mercer and Hall restricted their series to six sizes (1/250, 1/125, 1/100, 1/50, 1/25 and 1/10 acre), which are all possible combinations of rows and columns to this size; larger plots were considered unrealistic for agricultural experimentation.

Task 72 : Determine how adjacent plots can be grouped in increasingly-large blocks, up to 1/10 acre. •

We are restricted to divisors of 20 (rows), i.e. 2, 2 and 5, and 25 (columns), i.e. 5:

1/500 acre :	Original layout; grid is (20 x 25);
1/250 acre :	Combine the plots in each two adjacent rows; resulting grid is (10 x 25);
1/125 acre :	Combine the plots in each four adjacent rows; resulting grid is (5 x 25);
1/100 acre :	Combine the plots in each five adjacent columns; resulting grid is (20 x 5);
1/50 acre :	Combine the plots in each five adjacent columns and two adjacent rows; resulting grid is (10 x 5);
1/25 acre :	Combine the plots in each five adjacent columns and four adjacent rows; resulting grid is (5 x 5);
1/10 acre :	Combine plots in each five adjacent columns and ten adjacent rows; resulting grid is (2 x 5);

Three larger sizes are possible: 1/5, 1/4 and 1/2 acre; however these do not have enough plots to evaluate variability.

The resulting plots have quite different shapes:

Task 73 : Make a data structure with each plot size and its dimensions. •

From §12.2 we have the plot length, width, and area:

```
> plot.len

[1] 3.1807

> plot.wid

[1] 2.5446

> plot.area

[1] 8.0937
```

We can use these to compute dimensions for each combinations.

We first make a data frame with information about the combinations, using the `data.frame` function to combine three lists, each made with the `c` function; we also name the rows with the `row.names` function for easy reference. We name each field (column) explicity with the `field.name = ...` syntax.

```
> plots <- data.frame(acre.fraction = c(500, 250, 125,
+     100, 50, 25, 10), adj.rows = c(1, 2, 4, 1, 2, 4,
+     10), adj.col = c(1, 1, 1, 5, 5, 5, 5))
> row.names(plots) <- c("1/500", "1/250", "1/125", "1/100",
+     "1/50", "1/25", "1/10")
> str(plots)

'data.frame':        7 obs. of  3 variables:
 $ acre.fraction: num  500 250 125 100 50 25 10
 $ adj.rows     : num  1 2 4 1 2 4 10
 $ adj.col      : num  1 1 1 5 5 5 5
```

Now we can compute dimensions and add them to the frame with the `cbind` function:

```
> plots <- cbind(plots, len = plot.len * plots$adj.row)
> plots <- cbind(plots, wid = plot.wid * plots$adj.col)
> plots <- cbind(plots, area = plots$len * plots$wid)
> plots

      acre.fraction adj.rows adj.col     len     wid     area
1/500           500        1       1  3.1807  2.5446   8.0937
1/250           250        2       1  6.3615  2.5446  16.1874
1/125           125        4       1 12.7230  2.5446  32.3748
1/100           100        1       5  3.1807 12.7230  40.4686
1/50             50        2       5  6.3615 12.7230  80.9371
1/25             25        4       5 12.7230 12.7230 161.8742
1/10             10       10       5 31.8075 12.7230 404.6856
```

Note how the row names are shown when the entire frame is printed; this allows us to identify each combination.

Q83 : *What are the dimensions of these in meters, and their areas in m*^{2}*?* *Jump to A83* •

Task 74 : Compute the grain yields for the 1/250 acre plots made up of two adjacent rows, and its CV. •

To group in pairs, we make use of the **modulus** arithmetic operator %% to identify the odd and even rows:

```
> head(mhw$r%%2, 20)
```

```
 [1] 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
```

Now we split the plots into two groups using the `unstack` function and sum the two adjacent plots:

```
> tmp <- unstack(mhw, grain ~ r%%2)
> str(tmp)
```

```
'data.frame':        250 obs. of  2 variables:
 $ X0: num  4.07 3.9 3.16 3.42 3.4 4.43 4.46 5.13 4.38 3.61 ...
 $ X1: num  3.63 4.51 3.63 3.18 3.97 3.39 4.52 3.46 4.23 3.85 ...
```

```
> grain.250 <- tmp$X0 + tmp$X1
> rm(tmp)
> str(grain.250)
```

```
 num [1:250] 7.7 8.41 6.79 6.6 7.37 7.82 8.98 8.59 8.61 7.46 ...
```

```
> cv(grain.250)
```

```
[1] 10.1
```

Q84 : *Is the variation reduced, as expected, when plot size is doubled? By how much?* *Jump to A84* •

We now build a data frame of the combined plots, using the `data.frame` function, labeling each with its original column number and the average of the two rows:

```
> plots.250 <- data.frame(r = seq(1.5, 19.5, by = 2), c = rep(1:25,
+     each = 10), grain = grain.250)
> str(plots.250)
```

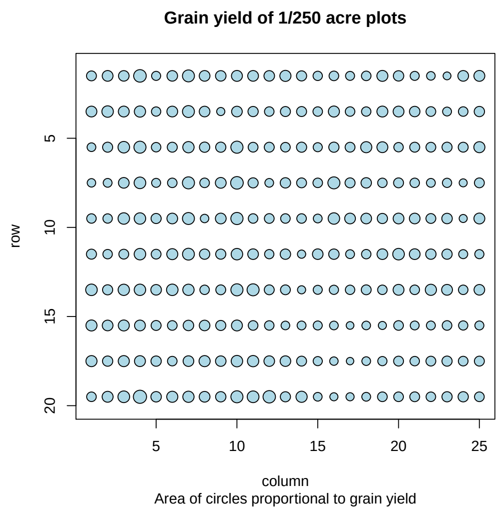
```
'data.frame':        250 obs. of  3 variables:
 $ r    : num  1.5 3.5 5.5 7.5 9.5 11.5 13.5 15.5 17.5 19.5 ...
 $ c    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ grain: num  7.7 8.41 6.79 6.6 7.37 7.82 8.98 8.59 8.61 7.46 ...
```

```
> rm(grain.250)
```

Task 75 : Visualise the variation across the field with the 1/250 acre plot size. •

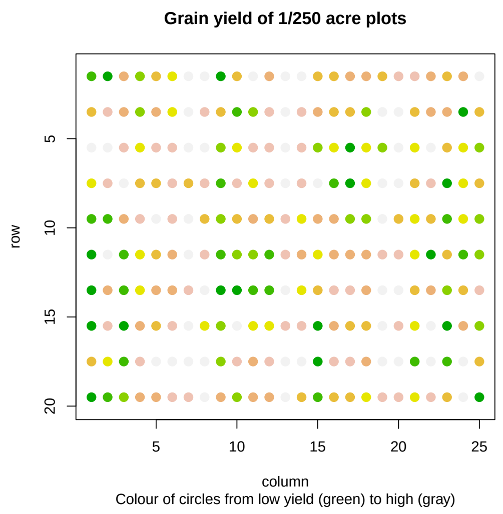
First by symbol size:

```
> plot(plots.250$c, plots.250$r, pch = 21, col = "black",
+     bg = "lightblue", main = "Grain yield of 1/250 acre plots",
+     sub = "Area of circles proportional to grain yield",
+     cex = 2 * plots.250$grain/max(plots.250$grain), xlim = c(1,
+         25), ylim = c(20, 1), xlab = "column", ylab = "row")
```



Then by colour ramp:

```
> plot(plots.250$c, plots.250$r, pch = 20, cex = 2, bg = "lightblue",
+     xlab = "column", ylab = "row", main = "Grain yield of 1/250 acre plots",
+     sub = "Colour of circles from low yield (green) to high (gray)",
+     xlim = c(1, 25), ylim = c(20, 1), col = terrain.colors(8)[cut(grain,
+         quantile(grain, seq(0, 1, length = 9)), include.lowest = T,
+         labels = F)])
```



These figures can be compared to those for the 1/500 acre plots in §12.1.

Q85 : *Does the field appear more homogeneous with 250 vs. 500 plots? What about the spatial pattern?* *Jump to A85* •

Task 76 : Repeat this process for the other combinations. •

First, for 1/125 acre. We introduce the very useful `apply` function, which applies any other function (here, the `sum`) across an array margin; here the `1` as second argument specifies that the sum is across rows of the matrix. Since the results of `unstack` are organized into a set of rows, this will add the four plots.

```
> tmp <- unstack(mhw, grain ~ r%%4)
> str(tmp)
```

```
'data.frame':	125 obs. of  4 variables:
 $ X0: num  3.9 3.42 4.43 5.13 3.61 4.64 3.35 3.7 3.89 4.22 ...
 $ X1: num  3.63 3.63 3.97 4.52 4.23 4.15 4.27 3.61 3.79 3.87 ...
 $ X2: num  4.07 3.16 3.4 4.46 4.38 4.21 3.55 3.71 4.09 4.12 ...
 $ X3: num  4.51 3.18 3.39 3.46 3.85 4.29 3.5 3.64 4.42 4.28 ...
```

```
> grain.125 <- apply(tmp, 1, sum)
> rm(tmp)
> str(grain.125)
```

```
 num [1:125] 16.1 13.4 15.2 17.6 16.1 ...
```

```
> cv(grain.125)
```

```
[1] 8.9
```

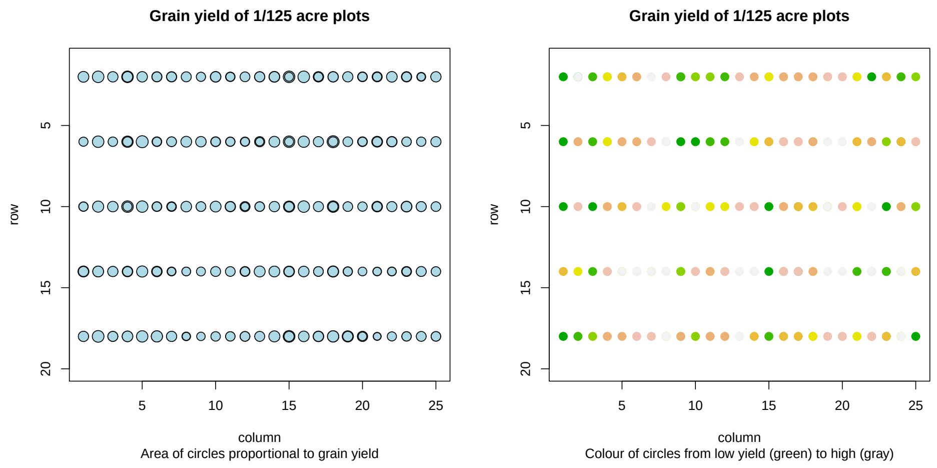
```
> plots.125 <- data.frame(r = seq(2, 18, by = 4), c = rep(1:25,
+     each = 10), grain = grain.125)
> str(plots.125)
```

```
'data.frame':	250 obs. of  3 variables:
 $ r    : num  2 6 10 14 18 2 6 10 14 18 ...
 $ c    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ grain: num  16.1 13.4 15.2 17.6 16.1 ...
```

```
> rm(grain.125)
```

Task 77 : Visualise the variation across the field with the 1/125 acre plot size. •

```
> par(mfrow = c(1, 2))
> plot(plots.125$c, plots.125$r, pch = 21, col = "black",
+     bg = "lightblue", main = "Grain yield of 1/125 acre plots",
+     sub = "Area of circles proportional to grain yield",
+     cex = 2 * plots.125$grain/max(plots.125$grain), xlim = c(1,
+         25), ylim = c(20, 1), xlab = "column", ylab = "row")
> plot(plots.125$c, plots.125$r, pch = 20, cex = 2, bg = "lightblue",
+     xlab = "column", ylab = "row", main = "Grain yield of 1/125 acre plots",
+     sub = "Colour of circles from low yield (green) to high (gray)",
+     xlim = c(1, 25), ylim = c(20, 1), col = terrain.colors(8)[cut(grain,
+         quantile(grain, seq(0, 1, length = 9)), include.lowest = T,
+         labels = F)])
> par(mfrow = c(1, 1))
```

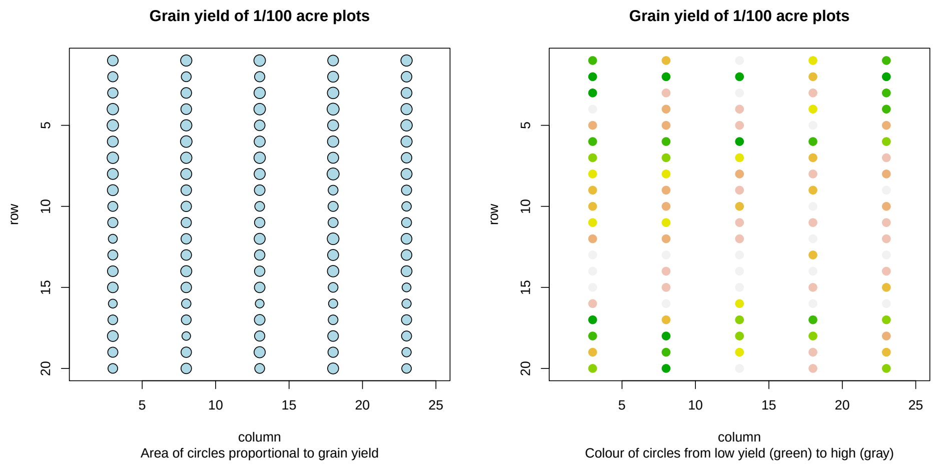


For the 1/100 acre plots the combination is by column:

```
> tmp <- unstack(mhw, grain ~ c%%5)
> grain.100 <- apply(tmp, 1, sum)
> rm(tmp)
> cv(grain.100)

[1] 7

> plots.100 <- data.frame(r = rep(1:20, each = 5), c = seq(3,
+     23, by = 5), grain = grain.100)
> str(plots.100)

'data.frame':        100 obs. of  3 variables:
 $ r    : int  1 1 1 1 1 2 2 2 2 2 ...
 $ c    : num  3 8 13 18 23 3 8 13 18 23 ...
 $ grain: num  20 21.1 21.7 20.1 21.2 ...

> rm(grain.100)
```

Task 78 : Visualise the variation across the field with the 1/100 acre plot size. •

```
> par(mfrow = c(1, 2))
> plot(plots.100$c, plots.100$r, pch = 21, col = "black",
+     bg = "lightblue", main = "Grain yield of 1/100 acre plots",
+     sub = "Area of circles proportional to grain yield",
+     cex = 2 * plots.100$grain/max(plots.100$grain), xlim = c(1,
+         25), ylim = c(20, 1), xlab = "column", ylab = "row")
> plot(plots.100$c, plots.100$r, pch = 20, cex = 2, bg = "lightblue",
+     xlab = "column", ylab = "row", main = "Grain yield of 1/100 acre plots",
+     sub = "Colour of circles from low yield (green) to high (gray)",
+     xlim = c(1, 25), ylim = c(20, 1), col = terrain.colors(8)[cut(grain,
+         quantile(grain, seq(0, 1, length = 9)), include.lowest = T,
+         labels = F)])
> par(mfrow = c(1, 1))
```

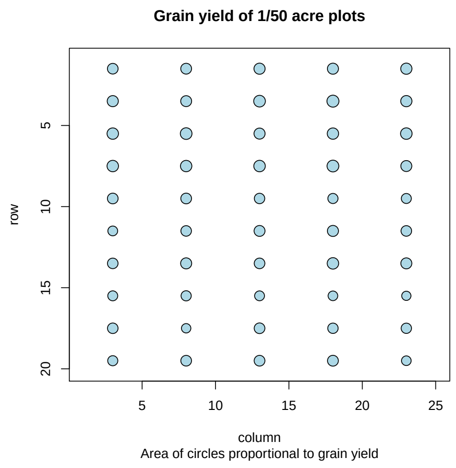


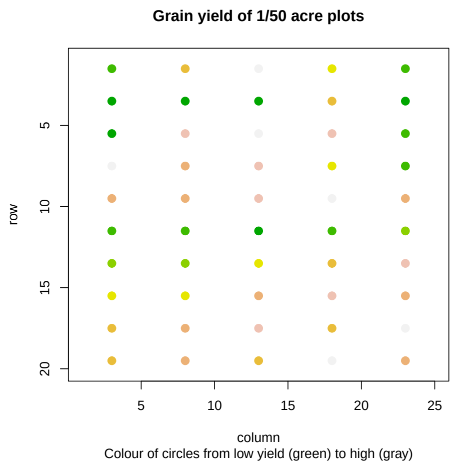
The 1/50 acre plots are the first where both rows and columns are combined. So we have to repeat the unstacking process twice. However, we can start from the 1/100 acre frame which already has combined the columns.

```
> tmp <- unstack(plots.100, grain ~ r%%2)
> grain.50 <- apply(tmp, 1, sum)
> rm(tmp)
> cv(grain.50)

[1] 5.9

> plots.50 <- data.frame(r = rep(seq(1.5, 19.5, by = 2),
+     each = 5), c = seq(3, 23, by = 5), grain = grain.50)
> str(plots.50)

'data.frame':        50 obs. of  3 variables:
 $ r    : num  1.5 1.5 1.5 1.5 1.5 3.5 3.5 3.5 3.5 3.5 ...
 $ c    : num  3 8 13 18 23 3 8 13 18 23 ...
 $ grain: num  39.1 39.7 40.9 40.6 41.1 ...

> rm(grain.50)
```

Task 79 : Visualise the variation across the field with the 1/50 acre plot size. •

```
> par(mfrow = c(1, 2))
> plot(plots.50$c, plots.50$r, pch = 21, col = "black",
+     bg = "lightblue", main = "Grain yield of 1/50 acre plots",
+     sub = "Area of circles proportional to grain yield",
+     cex = 2 * plots.50$grain/max(plots.50$grain), xlim = c(1,
+         25), ylim = c(20, 1), xlab = "column", ylab = "row")
> plot(plots.50$c, plots.50$r, pch = 20, cex = 2, bg = "lightblue",
+     xlab = "column", ylab = "row", main = "Grain yield of 1/50 acre plots",
+     sub = "Colour of circles from low yield (green) to high (gray)",
+     xlim = c(1, 25), ylim = c(20, 1), col = terrain.colors(8)[cut(grain,
+         quantile(grain, seq(0, 1, length = 9)), include.lowest = T,
+         labels = F)])
> par(mfrow = c(1, 1))
```





The 1/25 acre plots are constructed similarly, but combining four instead of two rows:

```
> tmp <- unstack(plots.100, grain ~ r%%4)
> grain.25 <- apply(tmp, 1, sum)
> rm(tmp)
> cv(grain.25)

[1] 4.8

> plots.25 <- data.frame(r = rep(seq(2.5, 18.5, by = 4),
+     each = 5), c = seq(3, 23, by = 5), grain = grain.25)
> str(plots.25)

'data.frame':        25 obs. of  3 variables:
 $ r    : num  2.5 2.5 2.5 2.5 2.5 6.5 6.5 6.5 6.5 6.5 ...
 $ c    : num  3 8 13 18 23 3 8 13 18 23 ...
 $ grain: num  79.9 79.8 83.8 84.6 82.4 ...

> rm(grain.25)
```

With so few plots, visualization of the variation is not effective.

The 1/10 acre plots are constructed similarly, but combining ten rows:

```
> tmp <- unstack(plots.100, grain ~ r%%10)
> grain.10 <- apply(tmp, 1, sum)
> rm(tmp)
> cv(grain.10)

[1] 3.9

> plots.10 <- data.frame(r = rep(seq(5.5, 15.5, by = 10),
+     each = 5), c = seq(3, 23, by = 5), grain = grain.10)
> str(plots.10)

'data.frame':        10 obs. of  3 variables:
 $ r    : num  5.5 5.5 5.5 5.5 5.5 15.5 15.5 15.5 15.5 15.5
 $ c    : num  3 8 13 18 23 3 8 13 18 23
 $ grain: num  203 204 205 207 202 ...

> rm(grain.10)
```

Now we attempt to answer Mercer & Hall's research question.

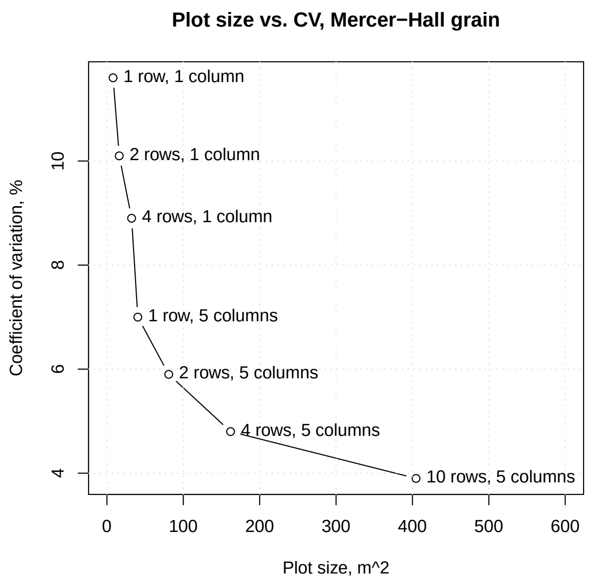
Q86 : *What is the trend of the summary statistics (extremes, mean, median, IQR) as the plot size increases, normalized to a 1/500 acre basis? Jump to A86* •

```
> summary(mhw$grain)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.73    3.64    3.94    3.95    4.27    5.16

> summary(plots.250$grain)/2

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.890   3.695   3.945   3.950   4.205   5.100

> summary(plots.125$grain)/4

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.025   3.675   3.925   3.950   4.150   4.800

> summary(plots.100$grain)/5

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.22    3.78    3.98    3.94    4.14    4.50

> summary(plots.50$grain)/10

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.40    3.82    3.95    3.95    4.11    4.40

> summary(plots.25$grain)/20

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.665   3.780   3.900   3.950   4.130   4.250

> summary(plots.10$grain)/50

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.72    3.82    3.98    3.94    4.08    4.14
```

Q87 : *What is the trend in the CV as the plot size increases? Jump to A87* •

```
> size.cv <- data.frame(area = plots$area, cv = c(cv(mhw$grain),
+     cv(plots.250$grain), cv(plots.125$grain), cv(plots.100$grain),
+     cv(plots.50$grain), cv(plots.25$grain), cv(plots.10$grain)))
> size.cv

      area   cv
1   8.0937 11.6
2  16.1874 10.1
3  32.3748  8.9
4  40.4686  7.0
5  80.9371  5.9
6 161.8742  4.8
7 404.6856  3.9
```

```
> plot(size.cv$area, size.cv$cv, xlab = "Plot size, m^2",
+     ylab = "Coefficient of variation, %", main = "Plot size vs. CV, Mercer-Hal
+     type = "b", xlim = c(0, 600))
> grid()
> text(size.cv$area, size.cv$cv, pos = 4, paste(plots$adj.rows,
+     " row", ifelse(plots$adj.rows == 1, "", "s"), ", ",
+     plots$adj.col, " column", ifelse(plots$adj.col ==
+         1, "", "s"), sep = ""))
```



Q88 : *What plot size do you recommend?* *Jump to A88* •

We now clean up from this section:

```
> rm(cv, plot.len, plot.wid, plot.area, plots, plots.250,
+     plots.125, plots.100, plots.50, plots.25, plots.10,
+     size.cv)
```

15.1 Answers

A81 : *The range of local spatial dependence was about 8 m; plot size is (3.30 m long x 2.45 m wide) (§A); grouping six plots into one plot of (6.60 m long x 7.35 m wide) would remove most of this structure; grouping 12 plots into one plot (9.90 m long x 9.70 m wide) would remove all of it.* *Return to Q81* •

A82 : *11.6 %.* *Return to Q82* •

A83 : *The dimensions are:*

```
> plots[, c("len", "wid", "area")]
```

```
            len     wid     area
1/500   3.1807  2.5446   8.0937
1/250   6.3615  2.5446  16.1874
1/125 12.7230  2.5446  32.3748
1/100   3.1807 12.7230  40.4686
1/50    6.3615 12.7230  80.9371
1/25   12.7230 12.7230 161.8742
1/10   31.8075 12.7230 404.6856
```

Return to Q83 •

A84 : *Yes, it is reduced somewhat, from 11.6% to 10.1%.* *Return to Q84* •

A85 : *There are half the plots so the detailed spatial structure is lost. However there are still clear patches of higher and lower yields.* *Return to Q85* •

A86 : *The means are almost identical* (3.94 *or* 3.95)*, and the medians close* (3.90 *to* 3.98)*; however the extremes (and hence the range) are reduced as plot size increases (from* 2.73 . . . 5.16 *in the full set to* 3.72 . . . 4.14 *in the largest plot) and the IQR is somewhat narrower (from* 3.64 . . . 4.27 *in the full set to* 3.82 . . . 4.08 *in the largest plot).* *Return to Q86* •

A87 : *The CV decreases rapidly at first, from 11.6% (500 plots) to 10.1% (250 plots) to 8.9% (125 plots) to 7.0% (100 plots), and then less dramatically, to 5.9% (50 plots), 4.8% (25 plots), and 3.9% (10 plots). The graph has a hyperbolic shape and shows a clear inflection point around 80 m^2 plot size (50 plots per acre).* *Return to Q87* •

A88 : *This depends on the precision required, which depends on the purpose of each experiment. However, a dramatic reduction in variability was achieved by grouping five adjacent columns.* *Return to Q88* •

16 Wrapup

If you are done with the dataset (both spatial and non-spatial), you can remove it from the workspace with the `rm` function; however you may want to keep this for further analysis.

If you do want to remove it:

```
> rm(mhw, mhw.sp)
```

A reminder: much more detailed information on R and the S language may be found in the "Introduction to the R Project for Statistical Computing for use at ITC" [26]; many more methods are covered in other technical notes [21, 22, 23, 24, 25, 27].

All done!

```
> q()
```

References

[1] R S Bivand, E J Pebesma, and V Gómez-Rubio. *Applied Spatial Data Analysis with R*. UseR! Springer, 2008. URL http://www.asdar-book.org/. 84

[2] G Casella and R L Berger. *Statistical inference*. Duxbury, Pacific Grove, CA, 2nd edition, 2002. 4

[3] W G Cochran. A catalogue of uniformity trial data. *Supplement to the Journal of the Royal Statistical Society*, 4(2):233–253, 1937. 114

[4] Noel Cressie. *Statistics for spatial data*. John Wiley & Sons, revised edition, 1993. Discussion of Mercer & Hall data on p. 455 ff. 114

[5] JC Davis. *Statistics and data analysis in geology*. John Wiley & Sons, New York, 3rd edition, 2002. 51

[6] H Fairfield Smith. An empirical law describing heterogeneity in the yields of agricultural crops. *The Journal of Agricultural Science (Cambridge)*, 28:1–3, 1938. 114

[7] J C Gower. Statistics and agriculture. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 151(1):179–200, 1988. 114

[8] R Ihaka and R Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996. 1

[9] R M Lark and J V Stafford. Information on within-field variability from sequences of yield maps: multivariate classification as a first step of interpretation. *Nutrient Cycling in Agroecosystems*, 50(1 - 3):277–281, 1998. 114

[10] F Leisch. *Sweave User's Manual*. TU Wein, Vienna (A), 2.1 edition, 2006. URL http://www.ci.tuwien.ac.at/~leisch/Sweave. 2

[11] F Leisch. Sweave, part I: Mixing R and LaTeX. *R News*, 2(3):28–31, December 2002. URL http://CRAN.R-project.org/doc/Rnews/. 2

[12] DM Mark and M Church. On the misuse of regression in earth science. *Mathematical Geology*, 9(1):63–77, 1977. 35

[13] A B McBratney and R Webster. Detection of ridge and furrow patterns by spectral analysis of crop yield. *International Statistical Review*, 49: 45–52, 1981. 114, 115

[14] W B Mercer and A D Hall. The experimental error of field trials. *The Journal of Agricultural Science (Cambridge)*, 4:107–132, 1911. 114

[15] V N Patankar. The goodness of fit of frequency distributions obtained from stochastic processes. *Biometrika*, 41:450–462, 1954. 93, 114, 115

[16] E J Pebesma. Multivariable geostatistics in S: the gstat package. *Computers & Geosciences*, 30(7):683–691, 2004. 84

[17] E J Pebesma and R S Bivand. Classes and methods for spatial data in R. *R News*, 5(2):9–13, November 2005. URL http://CRAN.R-project.org/doc/Rnews/. 84

[18] E J Pebesma and C G Wesseling. Gstat: a program for geostatistical modelling, prediction and simulation. *Computers & Geosciences*, 24(1): 17–31, 1998. URL http://www.gstat.org/. 84

[19] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2004. URL http://www.R-project.org. ISBN 3-900051-07-0. 1

[20] B D Ripley. *Spatial statistics*. John Wiley and Sons, New York, 1981. 114, 115

[21] D G Rossiter. *Technical Note: Optimal partitioning of soil transects with R*. (unpublished, online), Enschede (NL), 2004. URL http://www.itc.nl/personal/rossiter/teach/R/R_OptPart.pdf. 108

[22] D G Rossiter. *Technical Note: Statistical methods for accuracy assesment of classified thematic maps*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 2004. URL http://www.itc.nl/personal/rossiter/teach/R/R_ac.pdf. 108

[23] D G Rossiter. *Technical Note: Co-kriging with the gstat package of the R environment for statistical computing*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 2005. URL http://www.itc.nl/personal/rossiter/teach/R/R_ck.pdf. 108

[24] D G Rossiter. *Technical Note: An example of data analysis using the R environment for statistical computing*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 1.7 edition, 2005. URL http://www.itc.nl/personal/rossiter/teach/R/R_corregr.pdf. 108

[25] D G Rossiter. *Technical Note: Fitting rational functions to time series in R*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 2005. URL http://www.itc.nl/personal/rossiter/teach/R/R_rat.pdf. 108

[26] D G Rossiter. *Introduction to the R Project for Statistical Computing for use at ITC*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 3.6 edition, 2009. URL http://www.itc.nl/personal/rossiter/teach/R/RIntro_ITC.pdf. 1, 7, 108

[27] D G Rossiter and A Loza. *Technical Note: Analyzing land cover change with R*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 1.3 edition, 2004. URL http://www.itc.nl/personal/rossiter/teach/R/R_LCC.pdf. 108

[28] P Sprent. *Models in regression and related topics*. Methuen's monographs on applied probability and statistics. Methuen, London,, 1969. 51

[29] J.W. Tukey. *Exploratory data analysis*. Addison-Wesley, New York, 1977. 12

[30] H M van Es and C L van Es. Spatial nature of randomization and its effects on the outcome of field experiments. *Agronomy Journal*, 85: 420–428, 1993. 57, 114, 115

[31] WN Venables and BD Ripley. *Modern applied statistics with S*. Springer-Verlag, New York, fourth edition, 2002. 15, 35

[32] R Webster. Is regression what you really want? *Soil Use & Management*, 5(2):47–53, 1989. 35

[33] R Webster. Regression and functional relations. *European Journal of Soil Science*, 48(3):557–566, 1997. 35, 40, 51

[34] P Whittle. On stationary processes in the plane. *Biometrika*, 41:434–449, 1954. 114

[35] P Whittle. On the variation of yield variance with plot size. *Biometrika*, 43:337–343, 1956. 114

Index of R Concepts

A Example Data Set

In the early days of scientific agriculture, Mercer and Hall [14] were trying to determine the optimum plot size for agricultural yield trials:

- Plots that are too small will be too variable;
- Plots that are too large waste resources (land, labour, seed); if the land area is limited, the number of treatments (varieties, fertilizers, herbicides ...) will be restricted.

So, they performed a very simple experiment at the famous Rothamsted Experiment Station (Harpenden, Herts, England): an apparently homogeneous field was selected, prepared as uniformly as possible and planted to the same variety of wheat. They attempted to treat all parts of the field field exactly the same in all respects during subsequent farm operations. When the wheat had matured, the field was divided into 500 equally-size plots. Each plot was harvested separately. Both grain and straw were air-dryed, then hand-threshed and weighed to a precision of 0.01 lb (4.54 g = 0.00454 kg). The reported values are thus air-dry weight in pounds per plot.

The field was a square of 1 acre, a historical English measure of land area which is equivalent to 0.40469 ha (4,046.9 m^2), or 63.615 m on a side. The field was divided into a 20 rows by 25 columns, giving 500 plots, each of 1/500 acre. Dividing the square by the number of rows and columns, we obtain plots 3.1807 m long x 2.5446 m wide, with an area of 8.0937 m^2. We do not have records of the original orientation of the field, so[10] we assume that the rows ran W to E, with 25 plots in each row, beginning at 1 on the W and running to 25 at the E. Then the columns run S to N with 20 plots in each, beginning at 1 on the S and running to to 20 at the N.

Research questions This experiment was one of a series of so-called **uniformity trials** which were conducted early in the 20th century [3, 6], mainly to determine optimum plot sizes [35], field layouts and numbers of replications[11].

This data set has attracted many statisticians since 1911 [4, 6, 9, 13, 15, 20, 30, 34] because of a simple fact: although the yields should be identical, they are not; in fact they vary considerably. How is this to be explained?

Mercer and Hall distinguished two possible causes:

> "If we consider the causes of variation in the yield of a crop it seems that broadly speaking they are divisible into two kinds. The first are random, occurring at haphazard all over the field. Such would be attacks by birds, the incidence of weeds or the presence of lumps of manure. The second occur with more regularity, increasing from point to point or having centres from

[10] along with other statisticians who have analyzed this data, in particular McBratney and Webster [13]

[11] The intimate relation between the development of applied statistics and scientific agriculture is given in fascinating detail by Gower [7]

> which they spread outwards; we may take as instances of this kind changes of soil, moist patches over springs or the presence of rabbit holes along a hedge."

The first we recognize as multiple small random effects, with no spatial pattern, which should typify **normally-distributed** (Gaussian) effects. But it may also be the result of a **contaminated** process, where small effects are mixed with unusually-large deviations.

The second, however, may have a **spatial pattern**, so that this dataset is thus a testbed for *geo-statistical* analysis [13, 20, 30].

Some authors [13, 15, 30] have suggested that the field was not as uniform as assumed, so that there are both random effect as mentioned by Mercer and Hall but also systematic effects from previous management or trends in soil properties across the field.

The CSV file The data has been prepared as the **comma-separated values** ("CSV") file `mhw.csv` in a plain-text editor. The first line gives the four field names:

```
"r","c","grain","straw"
```

These represent:

`r` :	Row number in the field
`c` :	Column number in the field
`grain` :	Grain yield, lbs plot^{-1}
`straw` :	Straw yield, lbs plot^{-1}

The following 500 lines each represent a plot; the four fields are separated by commas. For example, the first line is:

```
1,1,3.63,6.37
```

Other sources of this dataset The `spdep` package includes the grain yields (only) and spatial positions of plot centres in its example dataset `wheat`.

B Colours

Colours may be specified in several ways; the most intuitive is by predefined **name**; these can be listed with the `colours` or `colors` methods.

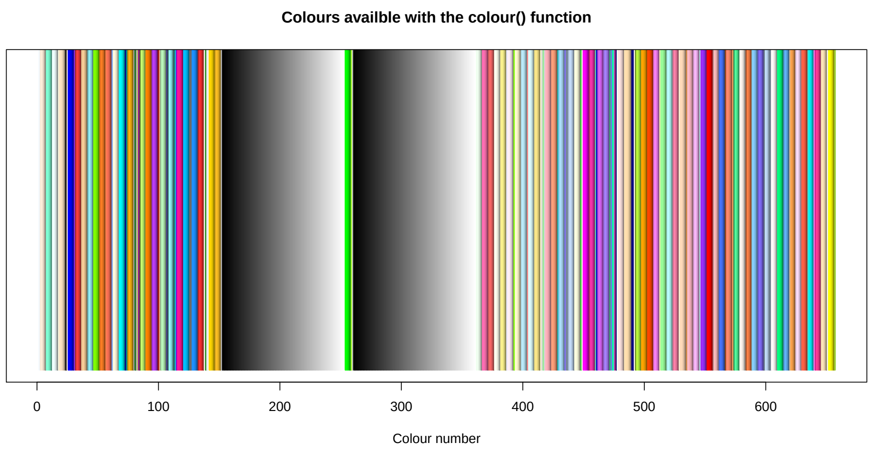
```
> colours()
```

```
 [1] "white"         "aliceblue"     "antiquewhite"  "antiquewhite1"
 [5] "antiquewhite2" "antiquewhite3" "antiquewhite4" "aquamarine"
 ...
```

> **Note:** These colours are shown various ways on the PDF colour chart http://research.stowers-institute.org/efg/R/Color/Chart/ColorChart.pdf.

These colours can be visualised as a bar graph:

```
> plot(seq(1:length(colors())), rep(2, length(colours())),
+     type = "h", lwd = 2, col = colors(), ylim = c(0,
+         1), xlab = "Colour number", ylab = "", yaxt = "n",
+     main = "Colours availble with the colour() function")
```



An individual colour number can be identified interactively with the `identify` function; left-click on the vertical colour bar at its midpoint; right-click anywhere in the graph when done.

```
> abline(h = 0.5, lwd = 3)
> (selected <- identify(seq(1:length(colors())), rep(0.5,
+     length(colors()))))
> colors()[selected]
> rm(selected)
```

For example, clicking on the light blue bar near colour 430, and then right-clicking to end the interaction, shows the colour number and name:

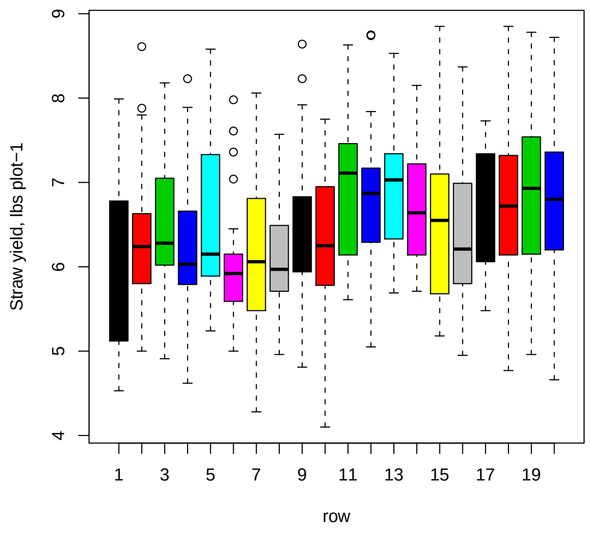
```
[1] 432

[1] "lightskyblue2"
```

Colours can also be refered by **number**; this is their position in the active **palette**. These names are displayed or extraced with the `palette` function:

```
> palette()

[1] "black"   "red"     "green3"  "blue"    "cyan"    "magenta"
[7] "yellow"  "gray"

> palette()[2]

[1] "red"
```

Numbered colours are often used when the graphical element matches a numbered element in some data structure:

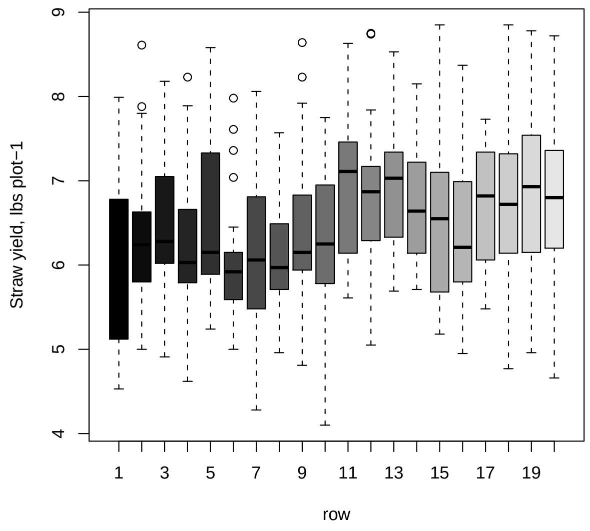
```
> boxplot(mhw$straw ~ mhw$r, col = mhw$r, xlab = "row",
+     ylab = "Straw yield, lbs plot-1")
```



Here the row number is directly used as the colour: row 1 black, row 2 red, row 3 green etc. Note that the colours are **recycled** if there are more plot elements than colours in the palette.

The `palette` function can also be used to set the palette. For example to make a 20-element grey-scale to match the 20 rows of wheat plots:

```
> palette(gray(seq(0, 0.9, len = 20)))
> palette()

 [1] "black"    "#0C0C0C"  "#181818"  "gray14"   "gray19"   "#3C3C3C"
 [7] "#484848"  "#555555"  "gray38"   "#6D6D6D"  "#797979"  "gray52"
[13] "gray57"   "#9D9D9D"  "darkgray" "gray71"   "#C1C1C1"  "#CDCDCD"
[19] "gray85"   "#E6E6E6"

> boxplot(mhw$straw ~ mhw$r, col = mhw$r, xlab = "row",
+     ylab = "Straw yield, lbs plot-1")
> palette("default")
> palette()

[1] "black"   "red"     "green3"  "blue"    "cyan"    "magenta"
[7] "yellow"  "gray"
```



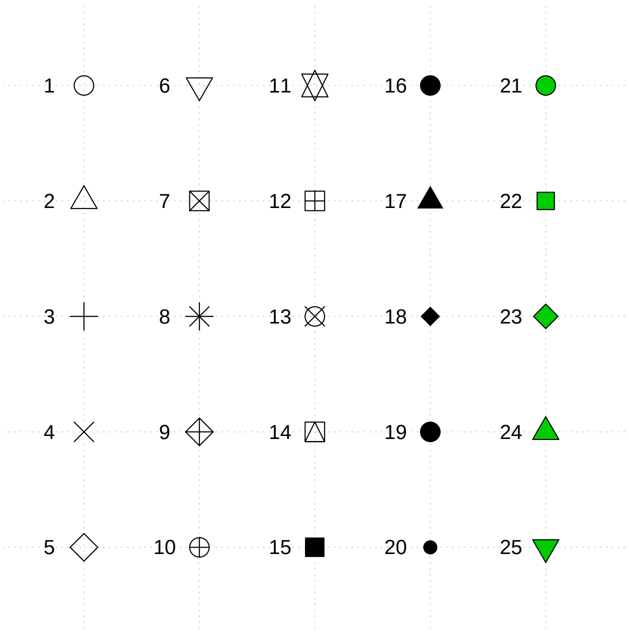
Note how some colours are given by their Red-Green-Blue saturations, as two hexidecimal digits per hue, e.g. **`#E6E6E6#`** is approximately 90% of white:

```
> 230/255
```

```
[1] 0.90196
```

This example also shows the **`gray`** function to set up a grey-scale colour ramp; other colour ramp functions are **`rgb`**, **`hsv`**, **`rainbow`**, and **`heat.colors`**.

Plotting symbols There are 25 pre-defined plotting symbols which can be specified with the **`pch`**, **`fg`** and **`bg`** graphics arguments:



In addition, ASCII characters can be used; e.g. **`pch=51`** prints a ‘1’.