Technical Note: Literate data analysis using the R environment for statistical computing and the `knitr` package

D G Rossiter

December 26, 2012

Contents

1 Overview

In 1992 Donald Knuth published a book with the title "Literate Programming" [3], showing the advantages of, and techniques for, writing computer programs to be read and understood by humans, as well as executed by a digital computer. This technical note advocates the same approach for data analysis: the computer code (here, in the R environment), and the output from executing it, is an integral part of a document that explains what the analyst did, why, and what was discovered. This is part of **reproducible research** [2, 6, 9]:

> "By reproducible research, we mean research papers with accompanying software tools that allow the reader to directly reproduce the results and employ the computational methods that are presented in the research paper."
>
> – Gentleman and Lang [1]; see also the CRAN Task View "Reproducible Research"[1]

The advantages of this approach are several:

1. Every processing step is transparent, since the R code is shown in the document;
2. Therefore, anyone can repeat the analysis if they are given access to the same data;
3. The analysis can easily be expanded or adapted;
4. If the data sources are edited, the entire analysis can be re-run and the results updated without any editing;
5. The analyst's explanations (motivation, justification, choice of methods, interpretations . . .) can be presented along with the results of the analysis;
6. The results of the computer processing are generated with the document, so they are by definition synchronized;
7. Figures are generated from code and are part of the output.

The approach presented here is only one element of fully reproducible research; this also requires that the original data and full details of its acquisition and manipulation be presented; for details see Mesirov [6].

The tools we use are:

[1] http://cran.r-project.org/web/views/ReproducibleResearch.html

Data processing — The R environment for statistical computing[2] [7];

Literate programming — The `knitR` ("knit R") R package[3] [10], which processes the literate programming source "NoWeb" file to produce both a LaTeX document and R code;

Text processing — LaTeX[4] [5], a document preparation system, to produce the final PDF document;

Text and code editor — To prepare the literate programming source. There are several good choices:

- Emacs[5], with the AUCTeX extension for working with LaTeX documents and the ESS ("Emacs Speaks Statistics")[6] extension for running R under Emacs. Learning Emacs is an investment in a lifetime of programming productivity, but not an overnight business.
- RStudio[7], an attractive GUI for R and NoWeb source files, with a reasonable code editor and built-in help on R commands.
- Microsoft Windows only: WinEdt[8] and the `R-WinEdt` R package to communicate with it; another option is Tinn-R[9].
- Any plain-text editor such as Notepad.

The flow is as follows:

1. You create a **source** document in a text editor with extension `.Rnw` (a so-called "NoWeb" file[10]); this source document includes LaTeX markup, your own text, and "chunks" of executable R code, using the NoWeb syntax (explained below) to show which parts of the source are executable code.

2. You run this NoWeb source through R with the R function `knit` of the `knitr` package; this produces a LaTeX file (extension `.tex`) which includes your original LaTeX markup and text, with the output from R (which may include graphics).

[2] http://www.r-project.org/
[3] http://yihui.name/knitr/
[4] http://www.latex-project.org/
[5] http://www.gnu.org/software/emacs/
[6] http://ess.r-project.org/
[7] http://rstudio.org/
[8] http://www.winedt.com/
[9] http://www.sciviews.org/Tinn-R/
[10] NoWeb, http://www.cs.tufts.edu/~nr/noweb/

3. You process the LaTeX file with LaTeX to produce a PDF document.

4. Optional: You run the NoWeb source through R with the R function `purl`, also of the `knitr` package, to produce an R source code file with the same name and extension `.R`; this can be executed in an R session with the R function `source`.

As you create your source document, you can also execute lines or chunks of code in the R console to see their effect. From some text editors (Emacs + ESS, RStudio) you can directly send lines or chunks of code from the NoWeb source to a linked R console; otherwise you have to work in the two environments separately. Thus you have an **interactive** data analysis as you work, but write it up in a document to be read by others.

> **Note:** The term "knit" is a wordplay on the original "Weave" from Knuth[11], who used that term as a reference to a poem by Sir Walter Scott: "Oh, what a tangled web we weave when first we practise to deceive"[12]; Knuth's original literate programming system was called WEB, so he decided to use "Weave" for the process of making the readable document and "Tangle" for the process of making the executable code. The author of the `knitr` package uses the term "purl" for the latter; this is a type of knitting. So now you know.

Task 1 : Set up your computing environment: text editor, R environment, LaTeX, and a PDF viewer. Within the R envionment, install the `knitr` package and its dependencies from CRAN[13]. •

To install the package from the R prompt:

```
> install.packages("knitr", dependencies=TRUE)
```

We now give a tutorial example (§2), and then get into some of the details and complications (§3).

[11] think "knitter", one who knits
[12] *Marmion*, VI:17
[13] http://cran.r-project.org/web/packages/knitr/index.html

2 Tutorial

We will do a small literate data analysis on one of R's example datasets, `trees`:

1. Examine the dataset structure;
2. Summarize the variables;
3. Graph the relation between them;
4. Build a linear model to predict tree volume from tree girth and height.

All of this is accompanied by our commentary – this is where we explain ("literately" we hope) what we are doing, why, and what conclusions we draw.

2.1 First version

Task 2 : Create a NoWeb file source file named `test1.Rnw`, open it in the text editor, and set up the LaTeX document. •

> **Note:** The `.Rnw` extension is used for NoWeb source files containing R code.

This is the usual document skeleton, naming the document class, loading packages, defining macros, etc. A minimal skeleton is:

```
\documentclass[11pt]{article}
\begin{document}
% LaTeX macros and text go here
\end{document}
```

There is usually a title, author, and date:

```
\documentclass[11pt]{article}
\title{Modelling tree volume}
\author{D.\ W.\ Luo}\date{\today}
\begin{document}
\maketitle
% LaTeX macros and text go here
\end{document}
```

Task 3 : Write the introductory text in the document section of the NoWeb source file (i.e., within the `document` environment). •

This should be your description (to your reader) of the purpose of this data analysis. It can be any valid LaTeX source. Here is my text:

```
Here we use the \texttt{trees} dataset supplied with R
to illustrate a simple data analysis:
\begin{enumerate}
\item describing the variables and cases;
\item investigating the inter-relation between variables; and
\item modelling tree volume as a function of tree height and/or tree girth.
\end{enumerate}
```

Task 4 : Write the code and commentary to load the example dataset. •

For this first example you just need to know one thing about NoWeb syntax: a **code chunk** is written between `<<>>=` and `@`; these must be the only text on their respective lines of NoWeb source. Anything between these is considered R code and will be formatted, executed, and the output written to the LaTeX source file.

```
<<>>=
# R code here
@
```

Anything *not* in a code chunk is regular LaTeX source – this is where you write comments and explanations.

My code and commentary is shown in §A.1; I added the following:

```
First, load the dataset, examine its structure, and summarize the variables:
\par
<<>>=
data(trees)
str(trees)
summary(trees)
@
```

Task 5 : Run this source file through the `knit` function within R to creates LaTeX source file with the same name but extension `.tex`. •

Note that the `knitr` package must be loaded into the R search path to make the `knit` function available. The `require` function loads a package if it's not already in the search path.

```
> require(knitr)
> knit("test1.Rnw")
processing file: test1.Rnw
  |>>>>>>>>>>>>>>>>>>>>>                                          |  33%
  ordinary text without R code
  |>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>                   |  67%
  label: unnamed-chunk-1
  |>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>| 100%
  ordinary text without R code

output file: test1.tex
```

The resulting file is named `test1.tex`; it should look like §B.1. The `knitr` package has provided some macros and environments, and has used various R packages to format the document. In general you never have to look at this file.

Task 6 : TEXify the LATEX source `test1.tex`: run LATEX to produce the PDF file, which will be named `test1.pdf`. •

The output should look like Figure 4.

2.2 Second version: adding graphics

The `knitr` package can produce graphical output in two ways:

1. The author writes the usual graphics commands in the code chunk; a figure is automatically generated, named, stored on your computer, and incorporated in the PDF via the LATEX `\includegraphics` macro;

2. The author explicitly opens a graphics device (e.g., a PDF file) and writes to it with R graphics commands and then closes the device.

The second option is only needed if you want to generate a figure specially formatted for publication; see §3.3.

There are a few details that make this process go smoothly. The first is to use a special code chunk to set default graphics (and other `knitr`) options. Within this chunk the `opts_chunk$set` function is used to specify the options. Arguments are either numbers or character strings. This special chunk should be the **first** code chunk in the NoWeb source (`.Rnw`) file.

Note: The full list of chunk options is given in the `knitr` documentation[14].

[14] http://yihui.name/knitr/options#chunk_options

For graphics the most important options are:

`fig.path` the location and the prefix of the file name of automatically-produced graphics files; the default is a `figure` directory under current directory and no file name prefix[15];

`fig.align` the figure alignment in the page; the default is no adjustment;

`fig.show` whether the figures should be displayed immediately or held until the end of the code chunk; default is `'asis'`, i.e., plots are shown where they were generated;

`fig.width` along with `fig.height`: the figure width and height in inches[16] in the PDF file; default for both is 7"; specifying a larger `fig.width` and `fig.height` results in smaller fonts relative to the graphic elements;

`out.width` the width of the figure on the printed page; the default is the line width.

You may well want to modify some of these defaults, for example to center figures, hold them all till the end of a code chunk, and use less of the line width to show figures. These options can also be changed in individual code chunks; here we over-ride some of the `knitr` defaults.

Note the difference between `fig.width` and `out.width`. The first specifies the width of the PDF figure printed at full size, the second how large it is shown in the document.

Task 7 : Change the default graphics options by adding the following code chunk at the beginning of your NoWeb source (i.e., the `.Rnw` file), immediately after the `\maketitle` LaTeX macro. Note the `setup` chunk option; this enables the special `opts_chunk` functions.

```
<<setup, include=FALSE, cache=FALSE>>=
opts_chunk$set(fig.path='figure/test-', fig.align='center',
               fig.show='hold',fig.width=6,
               fig.height=6,out.width='.75\\linewidth')
@
```

•

This says to put figures in the `figure` subdirectory (relative to the working directory), prefix the names with `test-`, make the PDF graphics each 6" by 6", print all figures from a chunk after it completes, and print each figure at 0.75 times the current line width.

[15] a subdirectory named in `fig.path` will be created if necessary
[16] 1" = 2.54 cm = 72 points exactly

> **Note:** The `include=FALSE` option in the chunk header tells `knitr` not to write this code into the PDF document; it's not part of the reproducible research, it's only used to set up the document.

Graphs will be generated with the names like `test-unnamed-chunk-1.pdf`; these names will be used in the `\includegraphics` LaTeX command.

Any of these options can be changed in an individual code chunk header, e.g.,

```
<<out.width='0.3\\linewidth'>>=
# R code to produce graphics, e.g., plot(), hist()
@
```

If `fig.show` is set to `'hold'` (as in the default chunk above), this will show each graphic at 0.3 times the current line width, thus allowing up to three separate figures side-by-side.

With this preparation, we can add a graph to our test document.

Task 8 :

1. Add code to the NoWeb source to draw a pairs plot of the three variables measured on the trees, i.e., pairwise scatterplots, using the `pairs` R function;

2. Display the graph in your R environment to check the graph is what you want and to interpret it;

3. Add some interpretative text to the NoWeb source explaining the graph;

4. Convert this NoWeb source file to LaTeX source within R, with the `knit` function;

5. TeXify the resulting LaTeX file to produce a PDF document.

•

My code chunk was:

```
<<>>=
pairs(trees, pch=20, cex=1.2)
@
```

My interepretation was:

```
There  appears to be a very strong relation between girth
and volume; this seems slightly non-linear (parabolic). The relation
between height and volume is also positive but much weaker. Height
and girth are very weakly related; this suggests that the trees have
different morphologies.
```

Now when we knit the source, this commentary is given right after the figure. The reader can see the figure and the analyst's interpretation.

My revised NoWeb source, with graphics commands and some comments, is shown in §A.2.

After "knitting" this source:

```
> knit("test2.Rnw")
```

we get the LaTeX source file `test2.tex`; after TeXifying this file we get the PDF file `test2.pdf` shown in Figure 5.

2.3 Third version: in-line calculations

The `knitr` package is also able to write calculated numbers as in-line text. For example, you might want to comment on the success of a model with something like: "The adjusted R^2 of the model is quite high (0.86)". But how do you know the number? You could compute it interactively in R and then cut-and-paste, but that is error-prone, and would have to be repeated if you change the model or dataset. Far better is to use the `\Sexpr` macro, which is processed by `knitr` and converted to LaTeX source. Most R expressions that produce a single number, and that do not go over a single line of LaTeX source when compiled, can be arguments to this macro; the results of the R calculation are written to the LaTeX source when the source file is processed by `knitr`.

For example, the LaTeX source text:

```
\Sexpr{round(2*pi/360, 5)}
```

will produce `0.01745` in the document.

You can compute interactively in R, see what works, and then add the relevant output to your in-line text in the NoWeb source.

Task 9 : Compute a linear model of tree volume modelled as an interaction between height and girth, and report its goodness-of-fit in-line with the

`\Sexpr` macro. Explain the processing steps in the text, and interpret the result. •

The model is built with the `lm` "linear models" function and reported with the `summary` generic method. I examined the model summary interactively, and decided to report the goodness-of-fit in the text as an adjusted R^2; this is given by the `adj.r.squared` field of the model summary given by `summary.lm`. My revised NoWeb source is shown in §A.3.

The relevant code chunk is:

```
<<>>=
# note: `*' is used to specify an interaction effect
m <- lm(Volume ~ Girth * Height, data=trees)
summary(m)
@
```

Notice that comments, prefixed with `#`, can also be included in the R code.

This is then explained with the following text; note the use of the `\Sexpr` macro, which uses the `summary` and `round` functions to format the adjusted R^2:

```
The success is quite good, as measured by the adjusted $R^2$
(\Sexpr{round(summary(m)$adj.r.squared*100,1)}\%).
```

After Sweaving this source, by running the R command:

```
> knit("test3.Rnw")
```

to produce file `"test3.tex"`, and TEXifying this file, we get the PDF file `test3.pdf` shown in Figure 7.

2.4 Writing an R source code file

You may want the R code as a separate file, for inclusion in an automatic process, as source for further experimentation, or to send to a collaborator. This is the job of the `purl` function of the `knitr` package.

Task 10 : Produce R code from the final NoWeb source. •

Recall, the source code is in file `test4.Rnw`. So, at the R prompt:

```
> purl("test3.Rnw")
```

The result is shown in §C.2. This source can now be run in R with the `source` function:

```
> source("test3.R")
```

This would run all the analysis and produce all the graphics (but not the document).

3 Details

The `knitr` documentation is somewhat scattered.

At YiHui Xie's main site[17]:

- the main manual[18]
- the graphics manual[19]
- demos[20]

Here we list a few details that may be especially useful or catch the unwary.

3.1 Multiple graphics on one line

If a code chunk produces multiple graphs, and the `fig.show` chunk option is `'hold'` (the default), `knitr` puts as many of them on a line as will fit, and then continues on other lines as necessary. If you would like several graphs side-by-side, all you need to do is ensure that the `out.width` key in the chunk header is narrow enough, relative to the line width, to allow several graphs on a line. For example, this code:

```
Here are two ways to visualize the relation between volume
and height. In the right-hand graph the tree girth is shown;
this is then a `2.5D' plot.
<<out.width='0.42\\linewidth',fig.width=5, fig.height=5>>=
plot(trees$Volume ~ trees$Height,
     xlab="DBH (inches)", ylab="Timber volume (cubic feet)")
grid()
with(trees, symbols(Height, Volume, circles = Girth/16,
            inches = FALSE, bg = "deeppink", fg = "gray30"))
@
```

will produce the plots of Figure 1.

[17] http://yihui.name/knitr/
[18] https://github.com/downloads/yihui/knitr/knitr-manual.pdf
[19] https://github.com/downloads/yihui/knitr/knitr-graphics.pdf
[20] http://yihui.name/knitr/demos

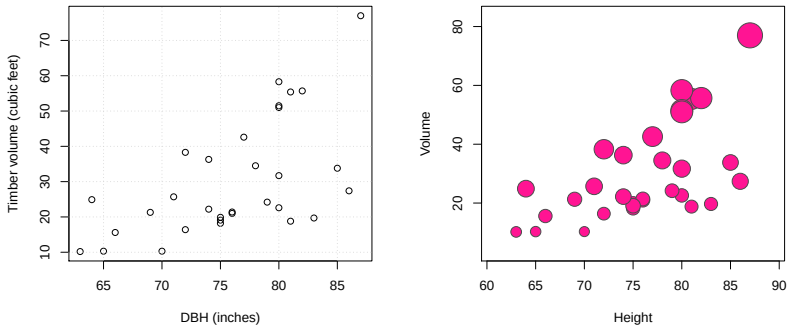


Figure 1: Two ways to visualize tree volume vs. height, side-by-side

3.2 Putting graphics in the `figure` environment

It is possible to put `knitr`-produced graphics inside the usual LaTeX `figure` environment, with a caption and label which can be referred to in the text. In this case, the figure "floats" to an aesthetically-pleasing[21] position on the page. For example, the following code makes Figure 2, which has floated to its current position automatically. Note the use of the `echo=FALSE` chunk option to prevent the code from being displayed in the `figure` environment.

The first argument to the chunk is the **chunk name**. The generated figure will use this name as part of the PDF file name.

```
  \begin{figure}
    \centering
<<volgirth, echo=FALSE>>=
plot(trees$Volume ~ trees$Girth, main="black cherry trees",
     xlab="DBH (inches)", ylab="Timber volume (cubic feet)")
grid()
@
    \caption{Thirty-one black Cherry trees}
    \label{fig:trees-size}
  \end{figure}
```

You would then refer to this figure in the text, in the usual LaTeX way, with the cross-reference to the label you assigned to the figure. For example:

```
Figure \ref{fig:trees-size} shows the relation between the
diameter at breast height (DBH) and harvestable volume.
```

[21] to LaTeX

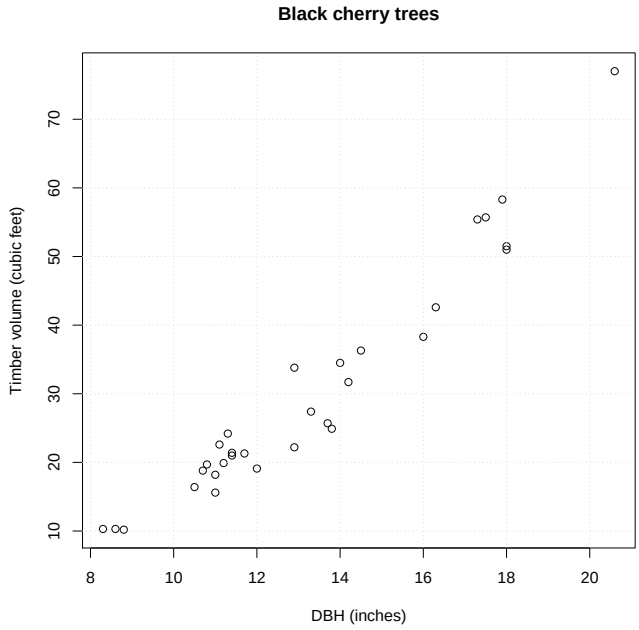


Figure 2: Thirty-one black cherry trees

3.3 Production graphics

The graphics produced by R are included in your PDF document and stored on your system. Each graphic is a separate PDF file and may be used by itself, e.g., for a journal article or thesis. These will have names like `test-unnamed-chunk-2.pdf`, according to the `fig.prefix` setup option; see §2.2.

However, you may want a different formatting for a production graphic, perhaps for submission to a journal according to their formatting requirements. To do this, within a code chunk open a graphics device, e.g., with the `pdf`, `jpeg` or `png` functions, write code to produce the graph, and close the graphics device with the `dev.off` function. For example:

```
<<fig.keep='none', echo=FALSE, results='hide'>>=
pdf(file="graph/scatterGirthHeight.pdf",
    width=5, height=5, title="Figure 1",
    bg="lightgray", fg="darkred")
plot(trees$Girth ~ trees$Height, pch=20, cex=1.5,
     xlab="Height (feet)", ylab="Girth (inches)")
dev.off()
@
```

Note the `fig.keep='none'` key in the chunk header. This specifies that the figure should not be retained by `knitr` for output within the PDF. Also note the many options that can be given the function that opens the graphics

device, here `pdf`. The `echo=FALSE` key prevents `knitr` from printing the code in the PDF document; the `results='hide'` key supresses the results of any calculation.

This produces the nice graphic shown in Figure 3. If you do want to show this in your document, you can explicitly name it with the `\includgraphics` LaTeX command (or equivalent), e.g., and refer to it as shown in §3.2:

```
\begin{figure}
  \centering
  \caption{Relation between girth and height, 31 cherry trees}
\includegraphics{graph/scatterGirthHeight.pdf}
  \label{fig:trees-scatter}
\end{figure}

Figure \ref{fig:trees-scatter} shows that there is a poor relation
between girth and height.
```

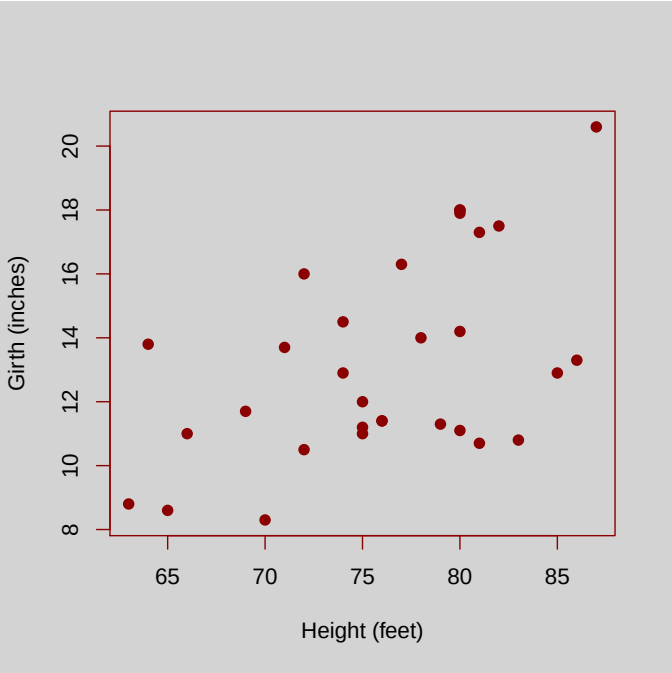


Figure 3: Relation between girth and height, 31 cherry trees

3.4 R code formatting and comments

If you are an experienced R programmer, you surely follow good programming practice:

- Formatting your code for readability, for example adding line breaks;
- Adding R comments (introduced with the `#` character) to explain your R code.

You can do these in your `knitr` source, and by default they will appear in your final document, as well as in any R code generated with the `purl` function, see §2.4). `knitr` uses several R packages to properly format and highlight thde code, including any comments.

3.5 Hiding code from the reader

You may want to execute some code that is irrelevant to readers, for example, changing to a directory on your system that will not be on their systems. You can hide code with the `echo=FALSE` chunk option:

For example, you may wish to set the working directory on your system, but this is irrelevant to the reader of the PDF document.

```
<<echo=FALSE>>=
setwd("/Users/Goliath/projects/secret/notell")
@
```

Any output will still be shown.

3.6 Showing code without executing it

You may want to show some code to explain it, but not want to execute it. You can prevent `knitr` from executing the code with the `eval=FALSE` chunk option:

```
You can show all the PDF fonts on your system as follows:
<<eval=FALSE>>=
str(pdfFonts())
@
```

This will appear in the document as:

> You can show all the PDF fonts on your system as follows:
>
> ```
> str(pdfFonts())
> ```

without producing any of the voluminous output from `pdffonts`. The reader now knows the command.

3.7 Hiding output from the reader

You may want to hide some output, probably because it is too long or verbose, but you want to show the reader what you did. You can hide output with the `results='hide'` chunk option:

```
Import the diurnal temperature differences for four Julian days:
<<>>=
DTD304 <- readGDAL("./images/CK_DTD_304.img")
@
<<results='hide'>>=
DTD306 <- readGDAL("./images/CK_DTD_306.img")
DTD307 <- readGDAL("./images/CK_DTD_307.img")
DTD308 <- readGDAL("./images/CK_DTD_308.img")
@
```

Here we want to show the results of the first import, but since the other three are exactly the same, there is no need to show the results.

This will appear in the document as:

Import the diurnal temperature differences for four Julian days:

```
DTD304 <- readGDAL("./images/CK_DTD_304.img")
## ./images/CK_DTD_304.img has GDAL driver HFA
## and has 28 rows and 28 columns
DTD306 <- readGDAL("./images/CK_DTD_306.img")
DTD307 <- readGDAL("./images/CK_DTD_307.img")
DTD308 <- readGDAL("./images/CK_DTD_308.img")
```

3.8 Formatting R code and output

The designer of `knitr` prefers to display R code without any R prompt symbols as displayed by the R console, with the logic that the R code can then be directly cut from the PDF document and pasted into an R console. For the same reason he prefers to display R output preceded by comment characters `##`.

These choices can be reversed by adding the arguments `prompt=TRUE` and `comment=NA` to the `opts_chunk$set` function. Here is some source code showing this:

```
\documentclass[11pt]{article}

This example shows the effect of using the \verb|prompt=TRUE| and
\verb|comment=NA| arguments to the \verb|opts_chunk$set| function
    in
the setup chunk. We show that chunk here:
```

```
<<setup, cache=FALSE>>=
opts_chunk$set(prompt=TRUE, comment=NA)
# options to be read by formatR
options(replace.assign=TRUE, width=72)
@

Here is some formatted code and output. We fit a model to predict
    tree
volume from the girth, height and their interaction two ways: (1)
    by
least squares, (2) with a resistant (``robust'') fit using the
\texttt{lqs} function of the \texttt{MASS} package; we then
    compare
the coefficients.

<<>>=
data(trees)
m1 <- lm(Volume ~ Girth * Height, data=trees)
require(MASS)
m2 <- lqs(Volume ~ Girth * Height, data=trees)
(coefficients(m1) - coefficients(m2))
@


\end{document}
```

The output is shown in Figure 9

4 Learning to use the tools

We've explained the interaction between the various tools; here we list some resources to get you started if you don't know how to use them,

4.1 `knitr`

This document has shown basic use of `knitr`; however this package is quite sophisticated and has many more capabilities; for example, it can produce HTML output and process Python input, and supports many graphics devices, not just base R graphics and `lattice` graphics. You are encouraged to at least skim the documentation at the `knitr` home page[22].

4.2 LaTeX

An excellent starting point is the LaTeX Wikibook[23]. This explains installation, simple and advanced usage, and tricks. It includes an "Absolute Beginners" section. Of course, the LaTeX project home page[24] is the definitive portal.

There are many texts; for serious work I recommend Kopka and Daly [4].

[22] http://yihui.name/knitr/
[23] http://en.wikibooks.org/wiki/LaTeX
[24] http://www.latex-project.org/

4.3 R

The R environment for statistical computing home page[25] is the entry point for information, downloads, and documentation.

I have written an introduction to R for use at the ITC faculty of the University of Twente [8]; §10 of that document lists some learning resources. The most useful for beginners may be Appendix A "A sample session" of the *Introduction to R* from the R Project[26]. This will give you some familiarity with the style of R sessions and more importantly some instant feedback on what actually happens. Don't worry if you don't understand everything; this is just to give you a feel for how R works and what it can do. For individual commands, it is always best to look at its help topic.

Many other introductions to R have been written, both as formal textbooks and on-line documents; see the "Documents" link in the table of contents of the R home page.

4.4 Emacs

If you choose to use Emacs, you face a steep learning curve but end up with a programming and text editing environment of unequalled power.

The reference manual at the GNU Emacs home page[27] is comprehensive and systematic, but slow going. The same group produces an Emacs Tour[28] which shows some of the capabilities.

Probably the best way to get started is to follow the tutorial built in to Emacs . This is accessed by using the "help" system and then pressing the `t` (for "tutorial") key. Unfortunately, different platforms and even different keyboard mappings have different ways to access the "help" system.

- Under X11 or Mac OS/X terminal, press the <f1> key.

If you start Emacs without a file name, the opening screen explains how to access the help system.

Emacs has many useful extensions, which may be installed by default, or you may have to install them. For editing LaTeX source, the AUCTeX extension can be used[29]. For communicating with R, and running R within the Emacs editor, the solution is the ESS ("Emacs Speaks Statistics")[30] extension.

To save files with non-Latin-1 characters (e.g., Chinese), use the Emacs command `set-buffer-file-coding-system` (which is usually mapped to key combination `Ctrl-X <RET> f`) and select the appropriate encoding, e.g., `utf-8`.

[25] http://www.r-project.org/
[26] http://cran.r-project.org/doc/manuals/R-intro.pdf
[27] http://www.gnu.org/software/emacs/#Manuals
[28] http://www.gnu.org/software/emacs/tour/
[29] http://www.gnu.org/software/auctex/
[30] http://ess.r-project.org/

References

[1] Robert Gentleman and Duncan Temple Lang. Statistical analyses and reproducible research. *Journal of Computational and Graphical Statistics*, 16(1):1–23, Mar 2007. doi: 10.1198/106186007X178663. 3

[2] Rob J. Hyndman. Encouraging replication and reproducible research. *International Journal of Forecasting*, 26(1):2–3, Jan 2010. doi: 10.1016/j.ijforecast.2009.12.003. 3

[3] D E Knuth. *Literate programming*. Center for the Study of Language and Information, 1992. ISBN 0937073814 (cloth) 0937073806 (paper). 3

[4] Helmut Kopka and Patrick W. Daly. *Guide to LaTeX*. Addison-Wesley, 2004. ISBN 0321173856. 19

[5] L Lamport. *LaTeX : a document preparation system : user's guide and reference manual*. Addison-Wesley Pub. Co., 1994. ISBN 0201529831. 4

[6] Jill P. Mesirov. Accessible reproducible research. *Science*, 327(5964): 415–416, Jan 2010. doi: 10.1126/science.1179653. 3

[7] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. URL http://www.R-project.org. ISBN 3-900051-07-0. 4

[8] D G Rossiter. *Introduction to the R Project for Statistical Computing for use at ITC*. University of Twente, Faculty ITC, 3.9 edition, May 2011. URL http://www.itc.nl/personal/rossiter/teach/R/RIntro_ITC.pdf. 20

[9] Victoria Stodden, Randall LeVeque, and Ian Mitchell. Reproducible research for scientific computing: Tools and strategies for changing the culture. *Computing in Science & Engineering*, 14(4):13–17, Jul 2012. doi: 10.1109/MCSE.2012.38. 3

[10] YiHui Xie. `knitr`: Elegant, flexible and fast dynamic report generation with R, 2011–2012. URL http://yihui.name/knitr/. Accessed 2012-12-26. 4

Index of R Concepts

A Input NoWeb files

A.1 First version of NoWeb source

This is file `test1.Rnw`, to be compiled with knitr into `test1.tex`; see below §B.1.

```
\documentclass[11pt]{article}
\title{Modelling tree volume}\author{D.\ W.\ Luo}
\date{\today}
\begin{document}
\maketitle
Here we use the \texttt{trees} dataset supplied with R to
    illustrate a simple data analysis:
\begin{enumerate}
\item describing the variables and cases;
\item investigating the inter-relation between variables; and
\item modelling tree volume as a function of tree height and/or
    tree girth.
\end{enumerate}
\par
First, load the dataset, examine its structure, and summarize the
     variables:
\par
<<>>=
data(trees)
str(trees)
summary(trees)
@

\end{document}
```

A.2 Second version of NoWeb source

```
\documentclass[11pt]{article}
\title{Modelling tree volume}\author{D.\ W.\ Luo}
\date{\today}
\begin{document}
\maketitle

<<setup, include=FALSE, cache=FALSE>>=
opts_chunk$set(fig.path='figure/test-', fig.align='center', fig.
    show='hold', fig.width=6, fig.height=6, out.width='.75\\
    linewidth')
@

Here we use the \texttt{trees} dataset supplied with R to
    illustrate a simple data analysis:
\begin{enumerate}
\item describing the variables and cases;
\item investigating the inter-relation between variables; and
\item modelling tree volume as a function of tree height and/or
    tree girth.
```

```
\end{enumerate}

\par
First, load the dataset, examine its structure, and summarize the
    variables:
\par
<<>>=
data(trees)
str(trees)
summary(trees)
@

\par
Second, look at the pairwise scatterplots of the three variables:
<<>>=
pairs(trees, pch=20, cex=1.2)
@
\par
There appears to be a very strong relation between girth and
    volume; this seems slightly non-linear (parabolic). The
    relation between height and volume is also positive but much
    weaker. Height and girth are very weakly related; this
    suggests that the trees have different morphologies.

\end{document}
```

A.3 Third version of NoWeb source

```
\documentclass[11pt]{article}
\title{Modelling tree volume}\author{D.\ W.\ Luo}
\date{\today}
\begin{document}
\maketitle

<<setup, include=FALSE, cache=FALSE>>=
opts_chunk$set(fig.path='figure/test-', fig.align='center', fig.
    show='hold', fig.width=6, fig.height=6, out.width='.75\\
    linewidth')
@

Here we use the \texttt{trees} dataset supplied with R to
    illustrate a simple data analysis:
\begin{enumerate}
\item describing the variables and cases;
\item investigating the inter-relation between variables; and
\item modelling tree volume as a function of tree height and/or
    tree girth.
\end{enumerate}

\par
First, load the dataset, examine its structure, and summarize the
    variables:
\par
```

```
<<>>=
data(trees)
str(trees)
summary(trees)
@

\par
Second, look at the pairwise scatterplots of the three variables:
<<>>=
pairs(trees, pch=20, cex=1.2)
@
\par
There appears to be a very strong relation between girth and
    volume; this seems slightly non-linear (parabolic). The
    relation between height and volume is also positive but much
    weaker. Height and girth are very weakly related; this
    suggests that the trees have different morphologies.

\par
Third, model the tree volume by a full model with the two
    possible predictors; include the interaction term:
<<>>=
# note: `*' is used to specify an interaction effect
m <- lm(Volume ~ Girth * Height, data=trees)
summary(m)
@

The success is quite good, as measured by the adjusted $R^2$ (\
    Sexpr{round(summary(m)$adj.r.squared*100,1)}\%).

\end{document}
```

B Intermediate files

B.1 First version of LaTeX file

This is the LaTeX file `test1.tex` produced by **knitr** from the source file `test1.Rnw`; see previous §A.1. This is now ready to be compiled by LaTeX into the output file `test1.pdf`.

```
\documentclass[11pt]{article}\usepackage{graphicx, color}
%% maxwidth is the original width if it is less than linewidth
%% otherwise use linewidth (to make sure the graphics do not
    exceed the margin)
\makeatletter
\def\maxwidth{ %
  \ifdim\Gin@nat@width>\linewidth
    \linewidth
  \else
    \Gin@nat@width
  \fi
}
```

```
\makeatother

\IfFileExists{upquote.sty}{\usepackage{upquote}}{}
\definecolor{fgcolor}{rgb}{0.2, 0.2, 0.2}
\newcommand{\hlnumber}[1]{\textcolor[rgb]{0,0,0}{#1}}%
\newcommand{\hlfunctioncall}[1]{\textcolor[rgb
    ]{0.501960784313725,0,0.329411764705882}{\textbf{#1}}}%
\newcommand{\hlstring}[1]{\textcolor[rgb]{0.6,0.6,1}{#1}}%
\newcommand{\hlkeyword}[1]{\textcolor[rgb]{0,0,0}{\textbf{#1}}}%
\newcommand{\hlargument}[1]{\textcolor[rgb
    ]{0.690196078431373,0.250980392156863,0.0196078431372549}{#1}}
    %
\newcommand{\hlcomment}[1]{\textcolor[rgb
    ]{0.180392156862745,0.6,0.341176470588235}{#1}}%
\newcommand{\hlroxygencomment}[1]{\textcolor[rgb
    ]{0.43921568627451,0.47843137254902,0.701960784313725}{#1}}%
\newcommand{\hlformalargs}[1]{\textcolor[rgb
    ]{0.690196078431373,0.250980392156863,0.0196078431372549}{#1}}
    %
\newcommand{\hleqformalargs}[1]{\textcolor[rgb
    ]{0.690196078431373,0.250980392156863,0.0196078431372549}{#1}}
    %
\newcommand{\hlassignement}[1]{\textcolor[rgb]{0,0,0}{\textbf
    {#1}}}%
\newcommand{\hlpackage}[1]{\textcolor[rgb
    ]{0.588235294117647,0.709803921568627,0.145098039215686}{#1}}
    %
\newcommand{\hlslot}[1]{\textit{#1}}%
\newcommand{\hlsymbol}[1]{\textcolor[rgb]{0,0,0}{#1}}%
\newcommand{\hlprompt}[1]{\textcolor[rgb]{0.2,0.2,0.2}{#1}}%

\usepackage{framed}
\makeatletter
\newenvironment{kframe}{%
 \def\at@end@of@kframe{}%
 \ifinner\ifhmode%
  \def\at@end@of@kframe{\end{minipage}}%
  \begin{minipage}{\columnwidth}%
 \fi\fi%
 \def\FrameCommand##1{\hskip\@totalleftmargin \hskip-\fboxsep
 \colorbox{shadecolor}{##1}\hskip-\fboxsep
     % There is no \\@totalrightmargin, so:
     \hskip-\linewidth \hskip-\@totalleftmargin \hskip\
         columnwidth}%
 \MakeFramed {\advance\hsize-\width
   \@totalleftmargin\z@ \linewidth\hsize
   \@setminipage}}%
 {\par\unskip\endMakeFramed%
 \at@end@of@kframe}
\makeatother

\definecolor{shadecolor}{rgb}{.97, .97, .97}
\definecolor{messagecolor}{rgb}{0, 0, 0}
\definecolor{warningcolor}{rgb}{1, 0, 1}
```

```
\definecolor{errorcolor}{rgb}{1, 0, 0}
\newenvironment{knitrout}{}{} % an empty environment to be
    redefined in TeX

\usepackage{alltt}
\title{Modelling tree volume}\author{D.\ W.\ Luo}
\date{\today}
\begin{document}
\maketitle
Here we use the \texttt{trees} dataset supplied with R to
    illustrate a simple data analysis:
\begin{enumerate}
\item describing the variables and cases;
\item investigating the inter-relation between variables; and
\item modelling tree volume as a function of tree height and/or
    tree girth.
\end{enumerate}
\par
First, load the dataset, examine its structure, and summarize the
     variables:
\par
\begin{knitrout}
\definecolor{shadecolor}{rgb}{0.969, 0.969, 0.969}\color{fgcolor
    }\begin{kframe}
\begin{alltt}
\hlfunctioncall{data}(trees)
\hlfunctioncall{str}(trees)
\end{alltt}
\begin{verbatim}
## 'data.frame':  31 obs. of  3 variables:
##  $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
##  $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
##  $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6
    19.9 ...
\end{verbatim}
\begin{alltt}
\hlfunctioncall{summary}(trees)
\end{alltt}
\begin{verbatim}
##      Girth           Height       Volume
##  Min.   : 8.3   Min.   :63   Min.   :10.2
##  1st Qu.:11.1   1st Qu.:72   1st Qu.:19.4
##  Median :12.9   Median :76   Median :24.2
##  Mean   :13.2   Mean   :76   Mean   :30.2
##  3rd Qu.:15.2   3rd Qu.:80   3rd Qu.:37.3
##  Max.   :20.6   Max.   :87   Max.   :77.0
\end{verbatim}
\end{kframe}
\end{knitrout}


\end{document}
```

C Output files

C.1 PDF

Modelling tree volume

D. W. Luo

December 26, 2012

Here we use the **trees** dataset supplied with R to illustrate a simple data analysis:

1. describing the variables and cases;
2. investigating the inter-relation between variables; and
3. modelling tree volume as a function of tree height and/or tree girth.

First, load the dataset, examine its structure, and summarize the variables:

```
data(trees)
str(trees)

## 'data.frame': 31 obs. of  3 variables:
##  $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
##  $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
##  $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 ...

summary(trees)

##      Girth           Height       Volume
##  Min.   : 8.3   Min.   :63   Min.   :10.2
##  1st Qu.:11.1   1st Qu.:72   1st Qu.:19.4
##  Median :12.9   Median :76   Median :24.2
##  Mean   :13.2   Mean   :76   Mean   :30.2
##  3rd Qu.:15.2   3rd Qu.:80   3rd Qu.:37.3
##  Max.   :20.6   Max.   :87   Max.   :77.0
```


Figure 4: First output

Modelling tree volume

D. W. Luo

December 26, 2012

Here we use the **trees** dataset supplied with R to illustrate a simple data analysis:

1. describing the variables and cases;
2. investigating the inter-relation between variables; and
3. modelling tree volume as a function of tree height and/or tree girth.

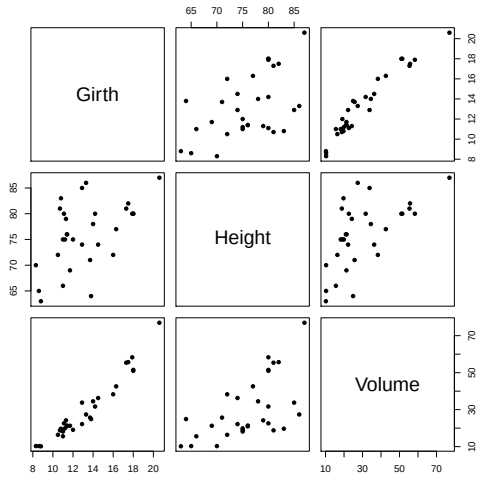
First, load the dataset, examine its structure, and summarize the variables:

```
data(trees)
str(trees)

## 'data.frame': 31 obs. of  3 variables:
##  $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
##  $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
##  $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 ...

summary(trees)

##      Girth           Height       Volume
##  Min.   : 8.3   Min.   :63   Min.   :10.2
##  1st Qu.:11.1   1st Qu.:72   1st Qu.:19.4
##  Median :12.9   Median :76   Median :24.2
##  Mean   :13.2   Mean   :76   Mean   :30.2
##  3rd Qu.:15.2   3rd Qu.:80   3rd Qu.:37.3
##  Max.   :20.6   Max.   :87   Max.   :77.0
```

Second, look at the pairwise scatterplots of the three variables:

Figure 5: Second output, with a graph (page 1 of 2)

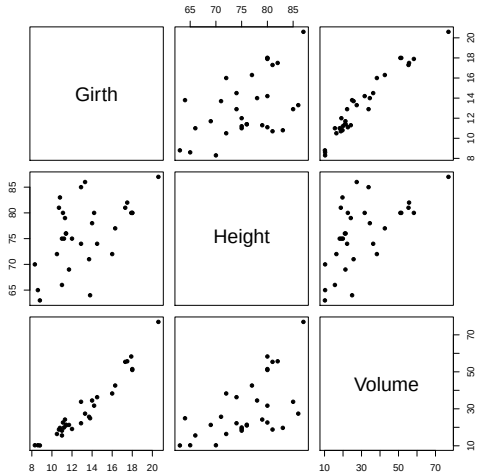
```
pairs(trees, pch = 20, cex = 1.2)
```



There appears to be a very strong relation between girth and volume; this seems slightly non-linear (parabolic). The relation between height and volume is also positive but much weaker. Height and girth are very weakly related; this suggests that the trees have different morphologies.

Figure 6: Second output, with a graph (page 2 of 2)

```
pairs(trees, pch = 20, cex = 1.2)
```



There appears to be a very strong relation between girth and volume; this seems slightly non-linear (parabolic). The relation between height and volume is also positive but much weaker. Height and girth are very weakly related; this suggests that the trees have different morphologies.

Third, model the tree volume by a full model with the two possible predictors; include the interaction term:

```
# note: `*' is used to specify an interaction effect
m <- lm(Volume ~ Girth * Height, data = trees)
summary(m)

##
## Call:
## lm(formula = Volume ~ Girth * Height, data = trees)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.582 -1.067  0.303  1.564  4.665
```


Figure 7: Third output, with a graph and in-line calculation (page 2 of 3; Page 1 is the same as Figure 5)

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   69.3963    23.8358    2.91  0.00713 **
## Girth         -5.8558     1.9213   -3.05  0.00511 **
## Height        -1.2971     0.3098   -4.19  0.00027 ***
## Girth:Height   0.1347     0.0244    5.52  7.5e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.71 on 27 degrees of freedom
## Multiple R-squared: 0.976,Adjusted R-squared: 0.973
## F-statistic:  359 on 3 and 27 DF,  p-value: <2e-16
```

The success is quite good, as measured by the adjusted R^2 (97.3%).

3

Figure 8: Third output, with a graph and in-line calculation (page 3 of 3; Page 1 is the same as Figure 5)

This example shows the effect of using the `prompt=TRUE` and `comment=NA` arguments to the `opts_chunk$set` function in the setup chunk. We show that chunk here:

```
opts_chunk$set(prompt = TRUE, comment = NA)
# options to be read by formatR
options(replace.assign = TRUE, width = 72)
```

Here is some formatted code and output. We fit a model to predict tree volume from the girth, height and their interaction two ways: (1) by least squares, (2) with a resistant ("robust") fit using the `lqs` function of the `MASS` package; we then compare the coefficients.

```
> data(trees)
> m1 <- lm(Volume ~ Girth * Height, data = trees)
> require(MASS)
> m2 <- lqs(Volume ~ Girth * Height, data = trees)
> (coefficients(m1) - coefficients(m2))

 (Intercept)        Girth       Height Girth:Height
   -18.45654      2.17256      0.28518     -0.03292
```

Figure 9: Effect of including the prompt character and excluding the comment character

C.2 R source code

This is file `test3.R`, compiled from `test3.Rnw` (§2.3) by the `purl` function..

```
## @knitr setup, include=FALSE, cache=FALSE
opts_chunk$set(fig.path='figure/test-', fig.align='center', fig.
    show='hold', fig.width=6, fig.height=6, out.width='.75\\
    linewidth')


## @knitr
data(trees)
str(trees)
summary(trees)


## @knitr
pairs(trees, pch=20, cex=1.2)


## @knitr
# note: `*' is used to specify an interaction effect
m <- lm(Volume ~ Girth * Height, data=trees)
summary(m)
```

Note the automatically-generated comments (marked with the `#` character).