A Tour of Sweave

Max Kuhn

Pfizer Global R&D
Non–Clinical Statistics
Groton

March 14, 2011

Creating Data Analysis Reports

For most projects where we need a written record of our work, creating the report can easily consume more time that the data analysis took.

There are at least two approaches to this

- Create and format tables and plots, then add text around these elements. In this case, the analysis code has no connection to the report.
- Take a combined approach, where the analysis results and the report are creating at the same time using the same source code.

In any case, it would be good if we could decrease the report writing time.

R and Sweave

Friedrich Leisch wrote an R utility called Sweave that allows users to "weave" R code and output together within Latex documents. This was later expanded for use with HTML.

Using Sweave, a user could write a report template (say in Latex) and put Sweave "tags" in it.

An in–line Sweave tag might look like `\Sexpr{sqrt(2)}`. This would be a signal to R to insert the results of the command `sqrt(2)` in place of the tag.

For example:

`There were \Sexpr{numSamples} samples used in the analysis.`

Sweaving

The general process to create a report with R output would be:

1. Write a template of the report and put in Sweave tags wherever output is needed
2. Open R run Sweave on the report template file
3. Sweave executes the data analysis and adds the missing elements to the report
4. (optional for Latex) Run latex or pdflatex to produce the final document

The result is a nice PDF or HTML document with the analysis results.

Of course, no software can understand why the analysis was done and what it all means, so there will always be some amount of editing after the initial report.

More Complex Tags

Block tags (called "code chunks") can also be used to written to execute longer pieces of code and can produce more complex output (e.g. images):

```
<<label = boxcoxPlot, fig = TRUE, echo = FALSE, results = hide>>=
# estimate the Box-Cox transformation for a linear model
boxcox(Days+1 ~ Eth*Sex*Age*Lrn, data = quine)
@
```

The options translate to:

- Label this "chunk" of code as boxcoxPlot.
- The output is a figure, so create an image file and the/HTML code to include it
- Don't echo the commands in the report output
- No textual results of the R code will be returned to the report.

More Complex Tags - Image Example

```
<<label = boxcoxPlot, fig = TRUE, echo = FALSE, results = hide>>=
# estimate the Box-Cox transformation for a linear model
boxcox(Days+1 ~ Eth*Sex*Age*Lrn, data = quine)
@
```

produces the image below and the code to render it in the document:
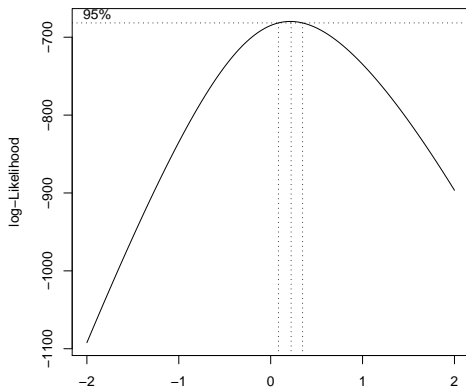
More Complex Tags - Table Example

For Fisher's iris data, let's tabulate the means of the flower measurements by each species of iris:

```
<<label = meanTable, echo = FALSE, results = tex>>=
irisMeans <- aggregate(iris[, 1:4], list(Species = iris$Species), mean)
meanTable <- latex(irisMeans,
                   file="",
                   caption = "Column means",
                   rowname=NULL,
                   ctable = TRUE)
@
```

produces the markup code that creates the table on the next page

More Complex Tags - Table Example

Table: Column means

Species	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
setosa	5.006	3.428	1.462	0.246
versicolor	5.936	2.770	4.260	1.326
virginica	6.588	2.974	5.552	2.026

More complicated tables can be produced:

More Complex Tags - Table Example

Table: Column Means

Species	Sepal		Petal	
	Length	Width	Length	Width
setosa	5.006	3.428	1.462	0.246
versicolor	5.936	2.770	4.260	1.326
virginica	6.588	2.974	5.552	2.026

More Complex Tags - Table Example

0.7

Descriptive Statistics for Boston Housing Data
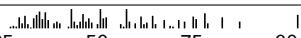### 3 Variables   506   Observations

---

chas

n	missing	unique
506	0	2

0 (471, 93%), 1 (35, 7%)

---

nox

n	missing	unique	Mean	.05	.10	.25	.50	.75	.90
506	0	81	0.5547	0.4092	0.4270	0.4490	0.5380	0.6240	0.7130

```
lowest : 0.385 0.389 0.392 0.394 0.398
highest: 0.713 0.718 0.740 0.770 0.871
```

---

rm

n	missing	unique	Mean	.05	.10	.25	.50	.75	.90	.95
506	0	446	6.285	5.314	5.594	5.886	6.208	6.623	7.152	7.588

```
lowest : 3.561 3.863 4.138 4.368 4.519
highest: 8.375 8.398 8.704 8.725 8.780
```

Advantages

In most cases, the report content and the code used to define the analysis are self–contained in one file.

This enables reproducible research: "papers with accompanying software tools that allow the reader to directly reproduce the results and employ the methods that are presented[1]."

My favorite quote: "An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship[2]."

A large number of Latex packages exist.

It can look pretty good. This presentation was completely produced using Sweave with Latex.

---

[1] Gentleman and Lange, "Statistical Analyses and Reproducible Research," *Bioconductor Working Paper*, 2004

[2] De Leeuw. Reproducible research – The Bottom Line, 2001

Disadvantages

Besides some statisticians/mathematicians/physicists/economist, who uses Latex?

The primary outputs would be pdf and html files. It is not easy for out clients to edit these documents after they have created.

With latex and HTML, markup elements are combined with the document contents and are hard to decouple. This could be important if you ever want to electronically extract any information from the document.

Extensions: odfWeave

The new Open Document Format (ODF) specification can give us the best of both worlds.

ODF files can be created using OpenOffice and other applications. ODF files are compressed archives that include text files in XML format. The ODF format covers reports, spreadsheets and presentations.

We've extended `Sweave` to work with ODF files with an R package called odfWeave.

Open Document Format

Using the Open Document Format with `odfWeave`:

- We get all the capabilities of `Sweave`
- An editor (OpenOffice) that allows user to create report templates without writing any markup. The interface looks very similar to Word and PowerPoint
- Converting to many other formats is easy: `doc`, `rtf`, `pdf`, `html` etc.
- We get a platform–independent specification that is not owned by any commercial entity
- All of the elements of the report (text, code, image files etc) are encapsulated in a single file
- ODF has many Office–like functions: time–stamping, mail–merging, spell check etc

Extensions: Caching Results

In many cases, one or more code chunks take a long time to run.
Sweaving a file then finding a bug in the last line stinks.

There are a few packages that allow the user to *cache* results; if the code
chunk (or it's dependencies) has not been changed, do not re–run it and
load the previous results.

```
<<label = BigCodeChunk, cache = TRUE>>=
(a bunch of long R code)
@
```

These packages include: cacheSweave, cacher and (my favorite) weaver.

Other Extensions

I maintain a CRAN Task View for Reproducible Research:

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

The full list of Sweave–related packages and extensions are given there.

References

R News articles:

- "Sweave, Part I: Mixing R and Latex" by Friedrich Leisch
- "Using Control Structures with Sweave" by Damian Betebenner (pg 40).
- "Using R/Sweave in everyday clinical practice" by Sven Garbade and Peter Burgard (pg 26)
- "Sweave and the Open Document Format - The odfWeave Package" by Max Kuhn (pg 2)

Also,

- http://cran.r-project.org/web/views/ReproducibleResearch.html
- The Sweave manual by Friedrich Leisch
- The Sweave FAQ by Friedrich Leisch
- An Sweave Demo by Charlie Geyer
- *Ecological Models and Data in R* by Ben Bolker (a book written using Sweave – Sweave code included)