Advanced Sweave Techniques

Duncan Murdoch

Department of Statistical and Actuarial Sciences
University of Western Ontario

May 30, 2014

Outline

Outline

What is Sweave?

- **Sweave** is part of a *literate programming system*. It allows you to mix text with R code in a single document.
- **Sweave** will process the document. The standard output format is LaTeX, though others are available (HTML, ODF, ??).
- The **knitr** package, written by Yihui Xie, has functions similar to **Sweave**.
- **Sweave** and **knitr** are the best ways to insert R graphics and the results of computations into LaTeX documents.
- I'll assume some familiarity with **Sweave**, and concentrate on advanced techniques. But a very short intro first...

Simple example

The idea of **Sweave** is that your document corresponds to a session of R. You embed text like this:

```
<<simpleexample,fig=TRUE>>=
set.seed(123)
x <- 1:10
y <- rnorm(10)
y
plot(x, y)
@
```

into a mostly LaTeX file with the **.Rnw** extension . . .

Simple example

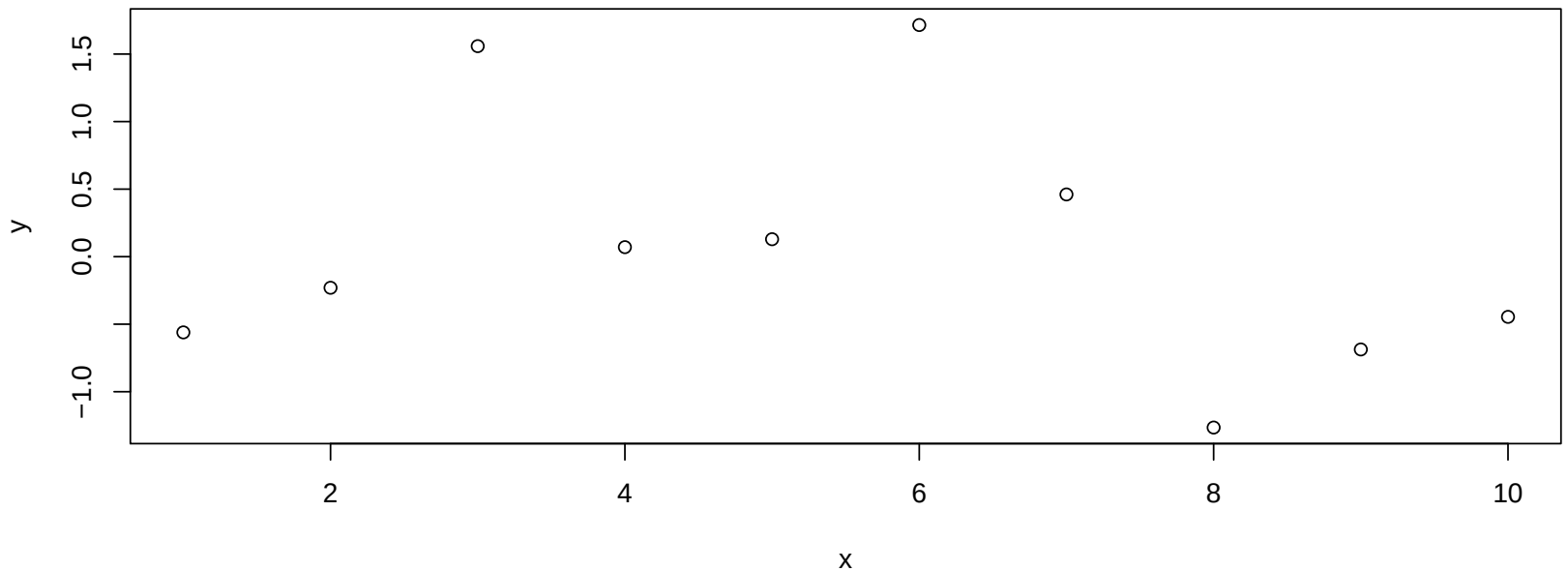
and **Sweave** converts it into pure LaTeX

```
\begin{Schunk}
\begin{Sinput}
> set.seed(123)
> x <- 1:10
> y <- rnorm(10)
> y
\end{Sinput}
\begin{Soutput}
 [1] -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
 [6]  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197
\end{Soutput}
\begin{Sinput}
> plot(x, y)
\end{Sinput}
\end{Schunk}
\includegraphics{figs/part2-simpleexample}
```

Simple example

which LaTeX processes into this:

```
> set.seed(123)
> x <- 1:10
> y <- rnorm(10)
> y
 [1] -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
 [6]  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197
> plot(x, y)
```



What's wrong with Sweave?

I use `Sweave` all the time, but in some ways it's hard to use:

- Do you run `Sweave()` within an R session (and possibly get unreproducible results, due to objects in your workspace)?

What's wrong with Sweave?

I use `Sweave` all the time, but in some ways it's hard to use:

- Do you run `Sweave()` within an R session (and possibly get unreproducible results, due to objects in your workspace)?
- Do you run it from a command line? (What's a command line?)

What's wrong with Sweave?

I use **`Sweave`** all the time, but in some ways it's hard to use:

- Do you run **`Sweave()`** within an R session (and possibly get unreproducible results, due to objects in your workspace)?
- Do you run it from a command line? (What's a command line?)
- How do you correct errors in what you write? (You see the preview, but where did it come from in the **`.Rnw`** file?)

What's wrong with Sweave?

I use **`Sweave`** all the time, but in some ways it's hard to use:

- Do you run **`Sweave()`** within an R session (and possibly get unreproducible results, due to objects in your workspace)?
- Do you run it from a command line? (What's a command line?)
- How do you correct errors in what you write? (You see the preview, but where did it come from in the **`.Rnw`** file?)
- How do you manage big projects, e.g. multi-chapter books?

Sweave: synchronizing

Modern PDF viewers (though not Acrobat!?!) support synchronization:

- You click on the preview, and your editor jumps to the source code.
- You tell your editor to jump to a particular location in the preview, and it does.
- But LaTeX thinks the `.tex` file is the source!

Sweave supports the `concordance` option, which outputs LaTeX code to record the correspondence between the `.tex` file and the original `.Rnw` file. This needs postprocessing.

Sweave: big projects

When writing a large document (a thesis, or a book) it makes sense to split the source text into multiple files, e.g. one per chapter.

- Speed up processing.
- Make it easier to edit.

Some LaTeX editors (e.g. TeXWorks on Windows, TeXShop on MacOS) allow you to record the "root" file in a subordinate `.tex` file. Some people put together "makefiles" describing big projects. Sweave doesn't have built-in support for this.

Outline

The `patchDVI` package

- I wrote `patchDVI` (Murdoch, 2013a) a few years ago to provide synchronization for `.dvi` files using the concordances.
- Since then I've added support for PDFs, and project management functions.

Concordances

The three lines of Sweave input

```
<<>>=
y
@
```

would generate 8 lines of LaTeX output:

```
\begin{Schunk}
\begin{Sinput}
> y
\end{Sinput}
\begin{Soutput}
[1] 2
\end{Soutput}
\end{Schunk}
```

The “concordance” describes how those output lines correspond to input lines.

Encoding of concordances

The concordance is stored in text in a file; for example, for this talk, the file could be **`Sweave-concordance.tex`**. Typical contents would be

```
\Sconcordance{concordance:Sweave.tex:Sweave.Rnw:%
1 93 1 2 5 71 1 1 2 1 0 3 1 8 0 1 1 4 0 %
1 2 82 1}
```

The `\Sconcordance` macro embeds this in the output. The filenames show output and input files; the numbers say the following:

Encoding of concordances

The concordance is stored in text in a file; for example, for this talk, the file could be **`Sweave-concordance.tex`**. Typical contents would be

```
\Sconcordance{concordance:Sweave.tex:Sweave.Rnw:%
1 93 1 2 5 71 1 1 2 1 0 3 1 8 0 1 1 4 0 %
1 2 82 1}
```

The `\Sconcordance` macro embeds this in the output. The filenames show output and input files; the numbers say the following:

- The first line of **`Sweave.tex`** is line 1 of **`Sweave.Rnw`**.

Encoding of concordances

The concordance is stored in text in a file; for example, for this talk, the file could be **`Sweave-concordance.tex`**. Typical contents would be

```
\Sconcordance{concordance:Sweave.tex:Sweave.Rnw:%
1 93 1 2 5 71 1 1 2 1 0 3 1 8 0 1 1 4 0 %
1 2 82 1}
```

The `\Sconcordance` macro embeds this in the output. The filenames show output and input files; the numbers say the following:

- The first line of **`Sweave.tex`** is line 1 of **`Sweave.Rnw`**.
- The next 93 lines each increase the position by 1.

Encoding of concordances

The concordance is stored in text in a file; for example, for this talk, the file could be **`Sweave-concordance.tex`**. Typical contents would be

```
\Sconcordance{concordance:Sweave.tex:Sweave.Rnw:%
1 93 1 2 5 71 1 1 2 1 0 3 1 8 0 1 1 4 0 %
1 2 82 1}
```

The `\Sconcordance` macro embeds this in the output. The filenames show output and input files; the numbers say the following:

- The first line of **`Sweave.tex`** is line 1 of **`Sweave.Rnw`**.
- The next 93 lines each increase the position by 1.
- The next 2 lines each increase the position by 5.

Encoding of concordances

The concordance is stored in text in a file; for example, for this talk, the file could be **`Sweave-concordance.tex`**. Typical contents would be

```
\Sconcordance{concordance:Sweave.tex:Sweave.Rnw:%
1 93 1 2 5 71 1 1 2 1 0 3 1 8 0 1 1 4 0 %
1 2 82 1}
```

The `\Sconcordance` macro embeds this in the output. The filenames show output and input files; the numbers say the following:

- The first line of **`Sweave.tex`** is line 1 of **`Sweave.Rnw`**.
- The next 93 lines each increase the position by 1.
- The next 2 lines each increase the position by 5.
- The next 71 lines each increase the position by 1.

Encoding of concordances

The concordance is stored in text in a file; for example, for this talk, the file could be **`Sweave-concordance.tex`**. Typical contents would be

```
\Sconcordance{concordance:Sweave.tex:Sweave.Rnw:%
1 93 1 2 5 71 1 1 2 1 0 3 1 8 0 1 1 4 0 %
1 2 82 1}
```

The `\Sconcordance` macro embeds this in the output. The filenames show output and input files; the numbers say the following:

- The first line of **`Sweave.tex`** is line 1 of **`Sweave.Rnw`**.
- The next 93 lines each increase the position by 1.
- The next 2 lines each increase the position by 5.
- The next 71 lines each increase the position by 1.
- etc.

Using concordances

- When LaTeX processes the file, the concordance is embedded in the output **`.pdf`** or **`.dvi`** file.
- The ***`pdflatex`*** program also creates a **`.synctex`** file relating the displayed output to the **`.tex`** file.
- **`patchDVI`** has code to read the output file, and patch the embedded links to source lines, so that they now refer to lines in the **`.Rnw`** source.
- Your previewer will now tell your editor to jump to the real source.

Project management

A large Sweave project may have multiple input files:

- A main **`.tex`** or **`.Rnw`** file containing `\input` commands to read the other **`.tex`** files.
- Some **`.Rnw`** files that need to be converted to **`.tex`** files.
- Some **`.tex`** files that don't need Sweave processing.
- Some static figures to include.

Building the document

The problem comes in keeping everything up to date: if I change **`Chapter2.Rnw`**, I need to

1. Convert **`Chapter2.Rnw`** to **`Chapter2.tex`**.
2. Convert any other changed Sweave files.
3. Run ***`pdflatex`*** on **`Main.tex`**.

Usually this will happen while I'm editing **`Chapter2.Rnw`**, so I want the rest to be automatic.

The `SweaveAll` function

The **`SweaveAll`** function in **`patchDVI`** runs Sweave on a file, then checks for the presence of some variables:

`.TexRoot` The name of the main **`.tex`** file.

`.SweaveFiles` A vector of other filenames that might need conversion by Sweave.

(The variables **`.SweaveMake`** and **`.PostSweaveHook`** also control the process.)

Sample file: Chapter2.Rnw

Chapter2.Rnw:

```
<<echo=FALSE>>=
.SweaveFiles <- c("Main.Rnw", "SubChapter.Rnw")
.TexRoot <- "Main.tex"
@
  ...
```

- The first line says that **patchDVI** should look at **Main.Rnw** and **SubChapter.Rnw** and see if they are newer than their corresponding **.tex** files. If so, process them as well.
- The second line says that when all the **.tex** files are up to date, it should run **Main.tex** through ***pdflatex***.

Sample file: Main.Rnw

Main.Rnw:

```
\documentclass{article}
<<echo=FALSE>>=
.SweaveFiles <- c("Chapter1.Rnw", "Chapter2.Rnw")
@
\begin{document}
 \input{Chapter1}
 \input{Chapter2}
 ...
```

- The code chunk says to make sure both chapters are up to date.
- The `\input` commands are regular LaTeX commands to include the final versions in the document.
- All of the **`.Rnw`** files will generate concordances, and the **`.pdf`** viewer will know which file to jump to.

Book Example

A short look at a live example....

Outline

Criticism of tables

- Tables often give a poor presentation of data. Gelman (2011) gave an entertaining discussion of ways that they are badly used.

Criticism of tables

- Tables often give a poor presentation of data. Gelman (2011) gave an entertaining discussion of ways that they are badly used.
- But it doesn't have to be so!

Principles of good tables

Ehrenberg (1977) is an excellent paper about producing tables. Some advice:

- Round to two significant or effective digits.
- Display row and column averages.
- Put items to be compared in the same column, one above the other.
- Order rows and columns by size.
- Don't insert too much white space: things to be compared should be close to each other, but add gaps every 5 or so rows to help the eye travel across the table.

This advice should be considered, not followed blindly: tables are meant for communication.

How to produce good tables?

- I don't think authors want to produce bad tables, I think they don't know better, or don't know how to do better, so I wrote the R package `tables` (Murdoch, 2013b) to make it easy to produce good tables.

How to produce good tables?

- I don't think authors want to produce bad tables, I think they don't know better, or don't know how to do better, so I wrote the R package `tables` (Murdoch, 2013b) to make it easy to produce good tables.
- Many years ago, I loved SAS PROC TABULATE, which made it pretty easy to do the computations necessary to produce good tables.
- **`tables`** improves on PROC TABULATE, by working well with Sweave and LaTeX. R is a particularly natural choice for this, much more flexible than SAS.

What is a table?

- A rectangular array of numbers or text (or pictures).
- Labels on the rows and columns. These may cover multiple entries, and may be nested.
- A caption.

The formula interface in **`tables`** handles the body and the labels. LaTeX can handle the captions.

Fisher's Iris Data

My examples work with Fisher's famous iris dataset:

```
> head(iris,10)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa

Group summaries

```
> booktabs()  # Choose "booktabs" style

> latex(tabular(Species ~ (Sepal.Length
+                          + Sepal.Width)
+                         *(mean + sd),
+                 data=iris))
```

Group summaries

```
\begin{tabular}{lcccc}
\toprule
 & \multicolumn{2}{c}{Sepal.Length} & \multicolumn{2}{c}{Sepal.Width}
Species  & mean & sd & mean & \multicolumn{1}{c}{sd} \\
\midrule
setosa  & $5.006$ & $0.3525$ & $3.428$ & $0.3791$ \\
versicolor  & $5.936$ & $0.5162$ & $2.770$ & $0.3138$ \\
virginica  & $6.588$ & $0.6359$ & $2.974$ & $0.3225$ \\
\bottomrule
\end{tabular}
```

Group summaries

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.006	0.3525	3.428	0.3791
versicolor	5.936	0.5162	2.770	0.3138
virginica	6.588	0.6359	2.974	0.3225

Group summaries

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.006	0.3525	3.428	0.3791
versicolor	5.936	0.5162	2.770	0.3138
virginica	6.588	0.6359	2.974	0.3225

Fewer digits!

Fewer digits

```
> latex(tabular(Species ~ Format(digits=2)
+                              *(Sepal.Length
+                                + Sepal.Width)
+                              *(mean + sd),
+                    data=iris))
```

Fewer digits

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38
versicolor	5.94	0.52	2.77	0.31
virginica	6.59	0.64	2.97	0.32

Fewer digits

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38
versicolor	5.94	0.52	2.77	0.31
virginica	6.59	0.64	2.97	0.32

Marginal summaries!

Marginal summaries

```
> latex(tabular(Species + 1 ~ Format(digits=2)
+                             *(Sepal.Length
+                               + Sepal.Width
+                               + 1)
+                             *(mean + sd),
+                   data=iris))
```

Marginal summaries (Oops...)

	Sepal.Length		Sepal.Width		All	
Species	mean	sd	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38	*NA*	*NA*
versicolor	5.94	0.52	2.77	0.31	*NA*	*NA*
virginica	6.59	0.64	2.97	0.32	*NA*	*NA*
All	5.84	0.83	3.06	0.44	*NA*	*NA*

Marginal summaries

```
> latex(tabular(Species + 1 ~ Format(digits=2)
+                             *(Sepal.Length
+                               + Sepal.Width)
+                             *(mean + sd),
+                   data=iris))
```

Marginal summaries

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38
versicolor	5.94	0.52	2.77	0.31
virginica	6.59	0.64	2.97	0.32
All	5.84	0.83	3.06	0.44

Marginal summaries

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38
versicolor	5.94	0.52	2.77	0.31
virginica	6.59	0.64	2.97	0.32
All	5.84	0.83	3.06	0.44

Spacing!

Spacing

```
> latex(tabular(Species
+                   + Hline(2:5) + 1
+                         ~ Format(digits=2)
+                           *(Sepal.Length
+                             + Sepal.Width)
+                           *(mean + sd),
+                   data=iris))
```

Spacing

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38
versicolor	5.94	0.52	2.77	0.31
virginica	6.59	0.64	2.97	0.32
All	5.84	0.83	3.06	0.44

Spacing

	Sepal.Length		Sepal.Width	
Species	mean	sd	mean	sd
setosa	5.01	0.35	3.43	0.38
versicolor	5.94	0.52	2.77	0.31
virginica	6.59	0.64	2.97	0.32
All	5.84	0.83	3.06	0.44

Better labels!

Better labels

```
> names <- paste("\\textit{Iris",
+                levels(iris$Species), "}")
> latex(tabular(Factor(Species, levelnames=names)
+       + Hline(2:5) + 1
+     ~ Format(digits=2)
+       *(Heading("Sepal length")*Sepal.Length
+       + Heading("Sepal width")*Sepal.Width)
+       *(mean + sd),
+       data=iris))
```

Better labels

	Sepal length		Sepal width	
Species	mean	sd	mean	sd
Iris setosa	5.01	0.35	3.43	0.38
Iris versicolor	5.94	0.52	2.77	0.31
Iris virginica	6.59	0.64	2.97	0.32
All	5.84	0.83	3.06	0.44

What's in a formula?

Terms in a formula can be:

function names	Summary statistics, e.g. `mean`.
factors	Categories, e.g. `Species`.
logical vectors	Subsets.
other vectors	Values to be summarized.
"pseudo-functions"	Things that handle formatting, e.g. `Format`.
formula functions	Abbreviate formulas, e.g. `Hline`

References I

A. S. C. Ehrenberg. Rudiments of numeracy. *Journal of the Royal Statistical Society, Series A*, 140:277–297, 1977.

Andrew Gelman. Why tables are really much better than graphs. *Journal of Computational and Graphical Statistics*, 20:3–7, 2011.

Duncan Murdoch. *patchDVI: Package to patch .dvi files*, 2013a. URL `http://cran.r-project.org/web/packages/patchDVI`. R package version 1.9.

Duncan Murdoch. *tables: Formula-driven table generation*, 2013b. R package version 0.7.64, on CRAN.