Sweave User Manual

Friedrich Leisch

Draft, January 4, 2002

1 Introduction

Sweave provides a flexible framework for mixing text and S code for automatic document generation. A single source file contains both documentation text and S code, which are then *weaved* into a final document containing

- the documentation text together with
- the S code and/or
- the output of the code (text, graphs)

by running the S code through an S engine[1] like R (R Development Core Team, 2001). This allows to re-generate a report if the input data change and documents the code to reproduce the analysis in the same file that also contains the report. The S code of the complete analysis is embedded into a LaTeXdocument using the noweb syntax (Ramsey, 1998). Hence, the full power of LaTeX(for high-quality typesetting) and S (for data analysis) can be used simultaneously.

Many S users are also latex users, hence no new software or syntax has to be learned. The emacs text editor (Stallman, 1999) offers a perfect authoring environment for Sweave, especially for people which already use emacs for writing latex documents and interacting with an S engine. We have chosen to use noweb as basis for the Sweave system because

1. the syntax is extremely simple and hence easy to learn
2. the ESS noweb mode for emacs already provides a perfect authoring environment

The importance of 2 should not be underestimated, a document format without convenient tools for authors will almost certainly be ignored by prospective users. However, it is not necessary to use emacs, Sweave is a standalone system, the noweb source files for Sweave can be written using any text editor.

Sweave uses a modular concept using different drivers for the actual translations. Obviously different drivers are needed for different text markup languages (LaTeX, HTML, . . .). tunately we will also need different drivers for different S engines (R, Splus), because we make extensive usage of `eval()`, connections, and the graphics devices, and the various S engines have some differences there. Currently there is only the driver `RWeaveLatex` which combines R and LaTeX

2 Noweb files

Noweb (Ramsey, 1998) is a simple literate-programming tool which allows to combine program source code and the corresponding documentation into a single file. Different programs allow to extract documentation and/or source code. A noweb file is a simple text file which consists of a sequence of code and documentation segments, these segments are called *chunks*:

[1] See Becker et al. (1988) and Chambers (1998) for definitions of the S language, and Venables and Ripley (2000) for details on the term *S engine* and detailed descriptions of differences between various implementations of the S language.

Documentation chunks start with a line that has an at sign (`@`) as first character, followed by a space or newline character. The rest of this line is a comment and ignored. Typically documentation chunks will contain text in a markup language like latex.

Code chunks start with `<<name>>` at the beginning of a line; again the rest of the line is a comment and ignored.

The default for the first chunk is documentation.

In the simplest usage of noweb, the (optional) names of code chunks give the name of source code files, and the tool `notangle` can be used to extract the code chunk from the noweb file. Multiple code chunks can have the same name, the corresponding code chunks are the concatenated when the source code is extracted. Noweb has some additional mechanisms to cross-reference code chunks (the `[[...]]` operator, etc.), Sweave does currently not use or support this features, hence they are not described here.

3 Sweave files

3.1 A simple example

Sweave source files are regular noweb files with some additional syntax that allows some additional control over the final output. Traditional noweb files have the extension `.nw`, which is alos fine for Sweave files (and fully supported by the software). Additionally, Sweave currently recognises files with extensions `.rnw`, `.Rnw`, `.snw` and `.Snw` to directly indicate a noweb file with Sweave extensions. We will use `.Snw` throughout this document.

A minimal Sweave file is shown in Figure 1, which contains two code chunks embedded in a simple latex document. Sweave translates this into the latex document shown in Figures 2 and 3. The first difference between the `example-1.Snw` and `example-1.tex` is that the latex style file `Sweave.sty` is automatically loaded, which provides environments for typesetting S input and output (the latex environments `Sinput` and `Soutput`). Otherwise, the documentation chunks are copied without any modification from `example-1.Snw` to `example-1.tex`.

The real work of Sweave is done on the code chunks: The first code chunk has no name, hence the default behaviour of Seave is used, which transfers both the S commands and their respective output to the latex file, embedded in `Sinput` and `Soutput` environments, respectively. Note that the test result needs to be `print()`ed explicetly, because eval would discard it otherwise. All output that should go into the final report must be created explicitly using a `print()`, `cat()` or similar statement.

The second code chunk shows one of the Sweave extension to the noweb syntax: Code chunk names can be used to pass options to Sweave which control the final output.

- The chunk is marked as a figure chunk (`fig=TRUE`) such that Sweave creates EPS and PDF files corresponding to the plot created by the commands in the chunk. Furthermore, a `\includegraphics{example-1-002}` statement is inserted into the latex file (details on the choice of filenames for figures follow later in this manual).
- Option `echo=FALSE` indicates that the S input should not be included in the final document (no `Sinput` environment).

3.2 Sweave options

Options control how code chunks and their output (text, figures) are transfered from the `.Snw` file to the `.tex` file. All options have the form `key=value`, where `value` can be a number, string or logical value. Several options can be specied at once (seperated by commas), all options must take a value (which must not contain a comma or equal sign). Logical options can take the values `true`, `false`, t, `f` and the respective uppercase versions.

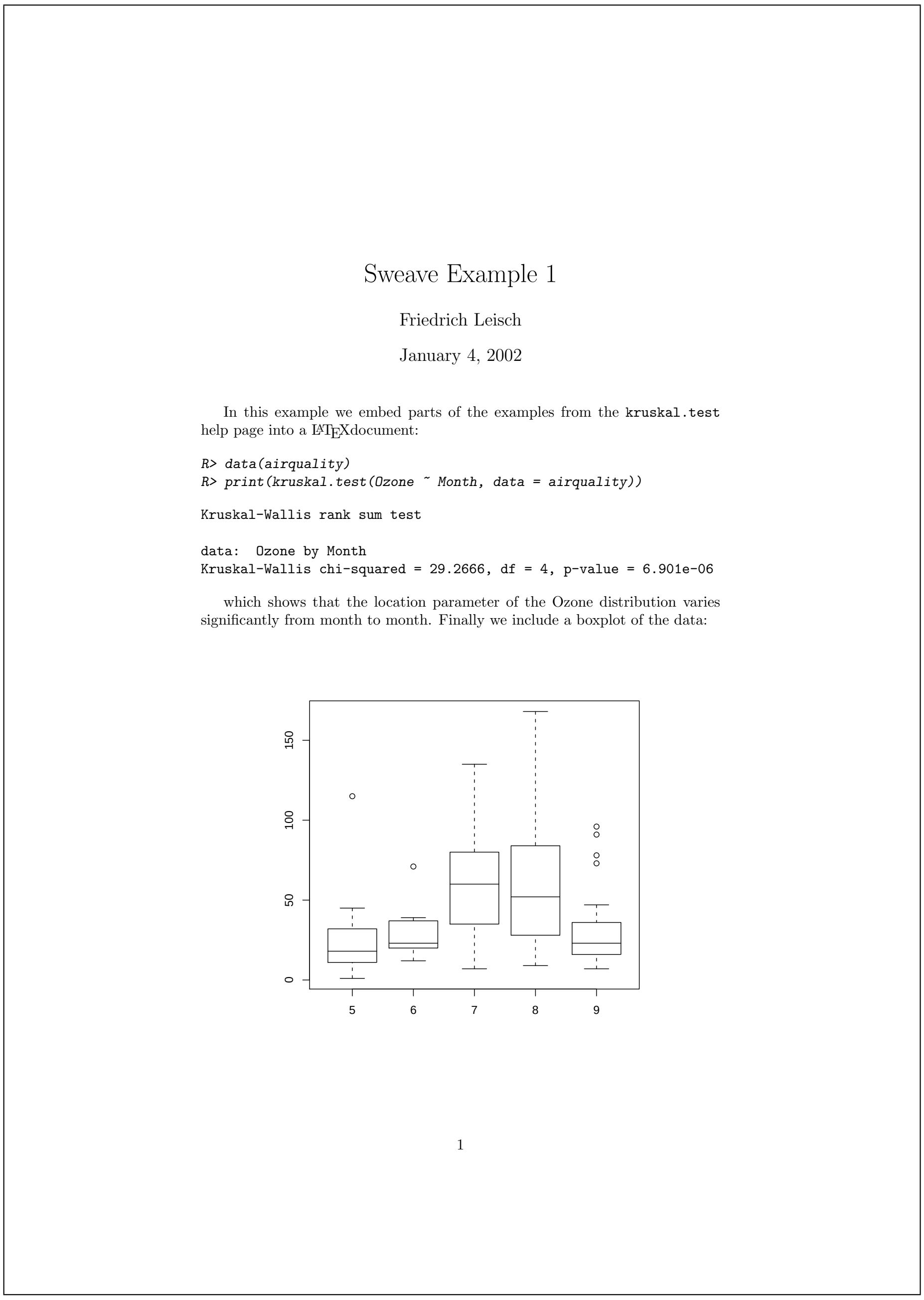
In the `.Snw` file options can be specified either

```
\documentclass[a4paper]{article}

\title{Sweave Example 1}
\author{Friedrich Leisch}

\begin{document}

\maketitle

In this example we embed parts of the examples from the
\texttt{kruskal.test} help page into a \LaTeX document:

<<>>=
data(airquality)
print(kruskal.test(Ozone ~ Month, data = airquality))
@

which shows that the location parameter of the Ozone
distribution varies significantly from month to month. Finally we
include a boxplot of the data:

\begin{center}
<<fig=TRUE,echo=FALSE>>=
boxplot(Ozone ~ Month, data = airquality)
@
\end{center}

\end{document}
```

Figure 1: A minimal Sweave file: `example-1.Snw`.

```
\documentclass[a4paper]{article}

\title{Sweave Example 1}
\author{Friedrich Leisch}

\usepackage{/home/Leisch/work/R/build-devel/library/tools/Sweave/Sweave}
\begin{document}

\maketitle

In this example we embed parts of the examples from the
\texttt{kruskal.test} help page into a \LaTeX document:

\begin{Sinput}
R> data(airquality)
R> print(kruskal.test(Ozone ~ Month, data = airquality))
\end{Sinput}
\begin{Soutput}
       Kruskal-Wallis rank sum test

data:  Ozone by Month
Kruskal-Wallis chi-squared = 29.2666, df = 4, p-value = 6.901e-06
\end{Soutput}

which shows that the location parameter of the Ozone
distribution varies significantly from month to month. Finally we
include a boxplot of the data:

\begin{center}
\includegraphics{example-1-002}
\end{center}

\end{document}
```

Figure 2: The output of `Sweave("example-1.Snw")` is the file `example-1.tex`.



Sweave Example 1

Friedrich Leisch

January 4, 2002

In this example we embed parts of the examples from the `kruskal.test` help page into a LaTeXdocument:

```
R> data(airquality)
R> print(kruskal.test(Ozone ~ Month, data = airquality))
```

```
Kruskal-Wallis rank sum test

data:  Ozone by Month
Kruskal-Wallis chi-squared = 29.2666, df = 4, p-value = 6.901e-06
```

which shows that the location parameter of the Ozone distribution varies significantly from month to month. Finally we include a boxplot of the data:

1

Figure 3: The final document is created by running `latex` on `example-1.tex`.

1. inside the angle brackets at the beginning of a code chunk, modifying the behaviour *only for this chunk*, or

2. anywhere in a documentation chunk using the command

   ```
   \SweaveOpts{opt1=value1, opt2=value2, ..., optN=valueN}
   ```

   which modifies the defaults for the rest of the document, i.e., *all code chunks after the statement*. Hence, an `SweaveOpts` statement in the preamble of the document sets defaults for all code chunks.

Which options are supported depends on the driver in use. All drivers should at least support the following options (all options appear together with their default value, if any):

engine=S: a character string describing which S engines are able to handle the respective code chunks. Possible values are, e.g., `S`, `R`, `S3` or `S4`. Each driver only processes compatible code chunks and ignores the rest.

split=FALSE: a logical value. If `TRUE`, then the output is distributed over several files, if `FALSE` all output is written to a single file. Details depend on the driver.

label: a text label for the code chunk, which is used for filename creation when `split=TRUE`. If the label is of form `label.engine`, then the extension is removed before futher usage (e.g., label `hello.S` is reduced to `hello`).

The first (and only the first) option in a code chunk name can be optionally without a name, then it is taken to be a label. I.e., starting a code chunk with

```
<<hello.S, split=FALSE>>
```

is the same as

```
<<split=FALSE, label=hello.S>>
```

but

```
<<split=FALSE, hello.S>>
```

gives a syntax error. Having an unnamed first argument for labels is needed for noweb compatibility. If only `SweaveOpts` is used for setting options, then Sweave files can be written to be fully compatible with noweb (as only filenames appear in code chunk names).

3.3 Using scalars in text

There is limited support for using the values of S objects in text chunks. Any occurrence of `\Sexpr{expr}` is replaced by the string resulting from coercing the value of the expression `expr` to a character vector; only the first element of this vector is used. E.g., `\Sexpr{sqrt(9)}` will be replaced by the string `"3"` (without any quotes).

The expression is evaluated in the same environment as the code chunks, hence one can access all objects defined in the code chunks which have appeared before the expression and were not ignored. The expression may contain any valid S code, only curly brackets are not allowed. This is not really a limitation, because more complicated computations can be easily done in a hidden code chunk and the result then be used inside a `\Sexpr`.

4 Tangling and weaving

The user frontends of the Sweave system are the two S functions `Stangle()` and `Sweave()`. Both (together with all available drivers for R) are contained in the base R package `tools` for R version 1.5.0 and later (`http://www.R-project.org`). `Stangle` can be used to extract only the code chunks from an `.Snw` file and write to one or several files. `Sweave()` runs the code chunks through an S engine and replaces them with the respective input and/or output. `Stangle` is actually just a wrapper function for Sweave, that uses a tangling instead of a weaving driver by default. The usage for both functions is shown in the appendix.

4.1 The `RweaveLatex` driver

This driver transforms `.Snw` files with latex documentation chunks and R code chunks to proper latex files (for typesetting both with standard `latex` or `pdflatex`).

Currently `RweaveLatex` supports the following options in addition to the general options:

echo=TRUE logical. If `TRUE`, the S code is included in the output file

results=verbatim: character string. If `results=verbatim`, the output of S commands is included in the verbatim-like `Soutput` environment. If `results=tex`, the output is taken to be already proper latex markup and included as is. If `results=hide` then all output is completely suppressed (but the code executed during the weave).

strip.white=TRUE: logical. If `TRUE`, blank lines at the beginning and end of output are removed.

prefix=TRUE: a logical indicating whether generated filenames of figures and output should have a common prefix.

prefix.string: a character string, default is the name of the `.Snw` source file.

include=TRUE: a logical indicating whether `\input{}` statements for text output and `\includegraphics{}` statements for figures should be auto-generated. Use `include=FALSE` if the output should appear in a different place than the code chunk (by placing the `\input{}` line manually).

fig=FALSE: a logical indicating whether the code chunk produces graphical output. Note that only one figure per code chunk can be processed this way.

eps=TRUE: logical, indicating whether EPS figures shall be generated. Ignored if `fig=FALSE`.

pdf=TRUE: logical, indicating whether PDF figures shall be generated. Ignored if `fig=FALSE`.

width: numeric, width of figures.

width: numeric, height of figures.

If `split=TRUE`, then all text corresponding to code chunks (the `Sinput` and `Soutput` environments) is written to seperate files. The filenames are of form `prefix.string-label.tex`, if several code chunks have the same label, their outputs are concatenated. If a code chunk has no label, then the number of the chunk is used instead. The same naming scheme applies to figures.

4.2 The `Rtangle` driver

This driver can be used to extract S and R code chunks from a `.Snw` file. Code chunks can either be written to one large file or seperate files (one for each label). The options `split`, `prefix`, `prefix.string` and `engine` have the same defaults and interpretation as for the `RweaveLatex` driver. Use the standard noweb command line tool `notangle` if other chunks than R or S code should be extracted.

5 Miscellaneous

5.1 Emacs support

As already mentioned above, the ESS mode (Rossini et al., 1999) for emacs (Stallman, 1999) provides a perfect authoring environment for Sweave. To use R mode as default code mode for `.Snw` files, include something like the following in your `.emacs` file:

```
(defun Rnw-mode ()
  (noweb-mode)
  (if (fboundp ’R-mode)
      (setq noweb-default-code-mode ’R-mode)))
(add-to-list ’auto-mode-alist ’("\\.Rnw\\’" . Rnw-mode))
(add-to-list ’auto-mode-alist ’("\\.Snw\\’" . Rnw-mode))
```

5.2 Resources

URLs for most software mentioned in this report can be found in the references below. The newest version of this document can always be found at the URL

`http://www.ci.tuwien.ac.at/~leisch/Sweave`

where you also find additional example files.

Acknowledgements

The author wants to thank Vince Carey, Robert Gentleman and Kurt Hornik for providing valuable comments and ideas.

References

Richard A. Becker, John M. Chambers, and Allan R. Wilks. *The New S Language.* Chapman & Hall, London, UK, 1988.

John M. Chambers. *Programming with data: A guide to the S language.* Springer Verlag, Berlin, Germany, 1998.

R Development Core Team. *R: A language and environment for statistical computing*, 2001. URL `http://www.R-project.org`. version 1.4.0.

Norman Ramsey. *Noweb man page.* University of Virginia, USA, 1998. URL `http://www.cs.virginia.edu/~nr/noweb`. version 2.9a.

Anthony J. Rossini, Richard M. Heiberger, Kurt Hornik, and Martin Maechler. *ESS: Emacs Speaks Statistics*, 1999. URL `http://ess.stat.wisc.edu`. version 5.1.x.

Richard M. Stallman. *The Emacs Editor.* Free Software Foundation, Boston, MA, USA, 1999. URL `http://www.gnu.org`. version 20.7.

William N. Venables and Brian D. Ripley. *S Programming.* Springer, 2000.