makesweave 0.3.0: Literate Programming with Make and Sweave

Charlotte Maia

November 22, 2010

1 Introduction

This vignette introduces the **makesweave** package, a (Linux-based) R package for using Make with Sweave efficiently.

The main technical goal is to start R once, then use the same R process for each use of Sweave. This is trivial when using R as a command line interface (with a user typing in commands), however is potentially very difficult, if we wish to use Make to build Sweave documents.

The approach taken here, is that we (after opening a shell), create a background process, which in turn, creates a child R process. Then, in the Makefile, instead of calling R CMD Sweave, we call a utility application (also called makesweave), which communicates with the background process, via a pipe, which in turn, communicates with R.

This can be improved further by using a Make variable, that defaults to "R CMD Sweave", however can be replaced by "makesweave". This allows use of our Makefile on systems, that do not have the makesweave package installed.

Note that this vignette was produced using the makesweave package. The Makefile and the main vignette source file (makesweavevignette.rnw) can be found in the inst/doc directory, plus the Makefile's contents are reproduced in Appendix A.

Appendix B contains some trivial R examples, just to prove that it works...

2 Current Requirements

1. A (typical) Linux system, along with:
   Bash, Make and GCC.
2. Latex.
3. R (with Sweave installed properly).

3 Installation

This approach, is slightly cumbersome, and will hopefully be simplified in a later version (excuse, the author is not familiar with writing R packages that contain C code...).

Firstly, install the R package in the regular way.

Note that at present, regular installation does not install the executable file. So secondly, unzip the package source file. Navigate to the src directory (the directory containing the file makesweave.c). Then as root, run: `make install`.

An alternative approach, is just to run `make`, then put the executable file that's created, in an appropriate place.

4 Using Make with Sweave

- Firstly, create a Makefile (similar to the one used here, editing the .rnw, .tex and .pdf targets), along with any Sweave source files.
- Secondly, open a terminal/shell, in the directory with both the Makefile and the Sweave source files.
- Thirdly, run the init target (in the shell): `make init`
  (Note that if we close the shell, we must re-run the init target).
- Fourthly, to build documents in the standard way, use: `make`
  Or, to build documents, in the enhanced way, use: `make enhanced`
- Last, if we want, run the clean target.

5 Issues

General issues:

- It is possible for problems to occur with the pipe. In which case, open a new shell, then re-run the init target.
- Sweave will create a .tex file, even if there's a problem with the Sweave source file. As of makesweave 0.2.0, Make will stop, if Sweave encounters and error. However, we then need to either make the clean target (or remove the corrupt .tex file some other way), or preferably, fix the Sweave source file. If we do neither, Make will assume we have a valid and up to date .tex file. Hence re-running Make, will cause Make to try and build the .pdf file from a corrupt .tex file).

Issues working with package vignettes:

- If we change an R package, that is loaded in R via the library function (e.g. library (mypackage)). We need to either re-run the init target (to restart R), or add to our Sweave source file, the appropriate R commands

to re-load the library.
Note that simply calling library (mypackage), does not re-load the package.

Issues working with multiple documents:

- In general, any sweave options (e.g. `\SweaveOpts{keep.source=TRUE}`), need to be included in every Sweave source file.
- Furthermore, in general, any R code that loads libraries, datasets, sets Sweave hook functions, or performs any other initialisation task, should also be included in every Sweave source file, unless we are certain that they are not necessary.
  Remember that the default target, restarts R for each file (hence a fresh R session), however the enhanced target, uses the same R session (so objects carry over).
- If an Sweave source file, defines a prefix string, then each file, must have a distinct prefix string. For files, a.rnw and b.rnw, we might do something like:
  `\SweaveOpts{prefix.string=tmp-a}` (in a.rnw)
  `\SweaveOpts{prefix.string=tmp-b}` (in b.rnw)

Appendix A: Makefile

```
bldcmd=R CMD Sweave

default:
    make makesweavevignette.pdf
    make clean

enhanced:
    make draft bldcmd=makesweave

draft:
    make makesweavevignette.pdf

makesweavevignette.pdf: makesweavevignette.tex trivial1.tex trivial2.tex
    pdflatex makesweavevignette.tex

makesweavevignette.tex: makesweavevignette.rnw
    $(bldcmd) makesweavevignette.rnw

trivial1.tex: ext/trivial1.rnw
    $(bldcmd) ext/trivial1.rnw

trivial2.tex: ext/trivial2.rnw
    $(bldcmd) ext/trivial2.rnw

init:
    rm -f /tmp/makesweavepipe
    mkfifo /tmp/makesweavepipe
    echo -e "0\n" > /tmp/makesweavestatus
    makesweave -i &

clean:
    rm *.tex
    rm *.aux
    rm *.log
```

Appendix B: Proof of The Pudding

Trivial example 1:
(R source in trivial1.rnw).

```
> x = 1:10
> y = x^2
> data.frame(x, y)

    x   y
1   1   1
2   2   4
3   3   9
4   4  16
5   5  25
6   6  36
7   7  49
8   8  64
9   9  81
10 10 100
```

Trivial example 2:
(R source in trivial2.rnw).

```
> suit = c("Hrt", "Dmd", "Spd", "Clb")
> value = c("A", 2:10, "J", "Q", "K")
> cards = outer(suit, value, paste, sep = ".")
> sample(cards, 5)

[1] "Hrt.3"  "Dmd.7"  "Hrt.10" "Clb.6"  "Clb.A"
```