Customizing Sweave to Produce Better Looking LaTeX Output

Ross Ihaka

May 25, 2009

1 Why Customize?

The default configuration of Sweave produces fairly ugly output. I've experimented quite a bit and have made some changes to the defaults which I feel produces better looking output. The changes I've found useful are listed below.

2 Improving the appearance of code and results

The default parameters for the layout of R code and output produce ugly looking results. The command prompt and user input are in "italic" font, the gap between input and outut is too large and the code blocks are not indented relative to the document text. Here is the default layout.

```
> 1:10

 [1]  1  2  3  4  5  6  7  8  9 10
```

I alter this by either adding the following lines at the end of the `Sweave.sty` file or at the start of my LaTeX document.

```
\DefineVerbatimEnvironment{Sinput}{Verbatim} {xleftmargin=2em}
\DefineVerbatimEnvironment{Soutput}{Verbatim}{xleftmargin=2em}
\DefineVerbatimEnvironment{Scode}{Verbatim}{xleftmargin=2em}
\fvset{listparameters={\setlength{\topsep}{0pt}}}
\renewenvironment{Schunk}{\vspace{\topsep}}{\vspace{\topsep}}
```

The result of these changes is to remove the italic font change, to indent code blocks with a 2 "em" space and to remove the vertical space between R input and output. You can see the difference below.

```
> 1:10
 [1]  1  2  3  4  5  6  7  8  9 10
```

You can find out the details on this kind of customisation by reading the documentation for the LaTeX `fancyverb` package.

3 Shortening output lines

The default output line-length produced by R is too long to be used with LaTeX. The following R expression shows the problem.

```
> 1:100
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
 [91]  91  92  93  94  95  96  97  98  99 100
```

As you can see, the result extends far beyond the right-hand margin.

This is easily fixed by changing the R option which sets the line length. This can be done by inserting the following noweb fragment into the LaTeX document (at a point before any output is produced).

```
<<echo=false>>=
options(width=60)
@
```

With this change, R ouput will fit comfortably within the margins of the standard LaTeX page.

```
> 1:100
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13
 [14]  14  15  16  17  18  19  20  21  22  23  24  25  26
 [27]  27  28  29  30  31  32  33  34  35  36  37  38  39
 [40]  40  41  42  43  44  45  46  47  48  49  50  51  52
 [53]  53  54  55  56  57  58  59  60  61  62  63  64  65
 [66]  66  67  68  69  70  71  72  73  74  75  76  77  78
 [79]  79  80  81  82  83  84  85  86  87  88  89  90  91
 [92]  92  93  94  95  96  97  98  99 100
```

4 Avoiding comment loss and code reformatting

By default, `Sweave` removes any comments present in the code and can reformat the code in ways that you may not like. For example, the input

```
## An R loop ...
for(i in 1:10)
  cat(i, "\n")
```

would be converted into the following form.

```
> for (i in 1:10) cat(i, "\n")
```

This kind of rearrangement can be very annoying when you really want code laid out in a particular way, or if you want to show comments. The behaviour can be changed by inserting the option

```
\SweaveOpts{keep.source=FALSE}
```

into the LaTeX preamble[1]

5 Removing continuation prompts

R uses a "+" prompt to indicate that the previous line was incomplete and that more text is expected to complete the input. For example, here is a simple function definition which shows the continuation prompt.

```
> fact =
+   function(n) {
+     if (n <= 1)
+       1
+     else
+       n * fact(n - 1)
+   }
```

This is useful when working interactively, but the continuation prompt does make it more difficult to read the code and also interferes with the ability to copy and paste from (PDF) documents. The prompt can be suppressed with the following noweb fragment.

```
<<echo=false>>=
options(continue="  ")
@
```

If this is placed before the function definition, the result is as follows.

```
> fact =
    function(n) {
      if (n <= 1)
        1
      else
        n * fact(n - 1)
    }
```

6 Including computed values in the running text

When preparing a report it can be useful to be able to insert values computed with R directly into the text. This can be done by enclosing the expression which computes the value inside a construction of the form `\Sexpr{...}`[2].

As an example, suppose that the variable `X` contains a χ^2 statistic and `df` the corresponding degrees of freedom. The information can be included in the text as follows.

[1] Note that the statement is not really a LaTeX statement but rather a instruction to `Sweave`. For the instruction to work, it must appear at the start of its own line.

[2] Again, this looks like LaTeX but it is actually processed by `Sweave`. This means that the entire expression must appear on a single line of LaTeX input

```
``the  $\chi^2$ statistic is \Sexpr{X} on
\Sexpr{df} degrees of freedom.''
```

This produces the result "the χ^2 statistic is 120.52 on 8 degrees of freedom."

7 Generating LaTeXwith R

It is possible to use R to generate LaTeX for direct inclusion in the output. Such LaTeX code must be generated inside a block of code of the form

```
<<results=tex,echo=false>>=
⋮
@
```

Here is an example which shows how to produce a very simple table of random normal deviates.

```
\begin{center}
\begin{tabular}{rrrrrrrr}
<<results=tex,echo=false>>=
nr = 3; nc = 8
x = matrix(format(round(rnorm(nr * nc), 2)), nc = nc)
for (i in 1:nr)
  for(j in 1:nc) {
    cat("$", x[i,j], "$")
    if(j < nc)
      cat("&")
    else
      cat("\\\\\n")
  }
@
\end{tabular}
\end{center}
```

This produces the following output.

−1.19	−0.57	1.03	0.79	−0.97	−1.11	−0.24	0.16
2.11	0.50	0.36	−1.44	1.52	−0.50	0.52	−0.11
1.07	−2.39	1.08	0.78	0.35	−0.89	−0.01	−0.61

The whole process of producing a LaTeX table can be encapsulated inside a single R function, so that tables can be produced by a single call to the function.

8 Including R graphics

Basic graphics are easy to incorporate into documents with `Sweave`. This is done with a noweb fragment of the following form.

```
<<fig=true,echo=false>>=
⋮
@
```

When `Sweave` processes this fragment, the effect is to produce the figure and to include it in-place the document. (It is usually best to include the figure within a `center` (or similar) environment.) Fragments can also included in a `figure` environment to get them floated to a later document page.

Sometimes it is useful to display the code which produces a figure as well as the figure itself. This can be done by naming the code fragment and hence the figure. This makes it possible to explcitly include the figure.

```
<<figname,include=false,fig=true>>=
plot(1:10)
@
\begin{figure}[tbp]
\centering
\includegraphics{docname-figname}
\caption{An example plot.}
\end{figure}
```

Here `docname` is the name of the document and `figname` is the name of the code fragment which produces the figure.

9 Customising graphics

Figures are produced using the default settings in place when a new device is started. In particular the standard margins are in place. The margins can be changed by resetting them at the start of the code fragment which produces a plot.

This can quickly become tedious and it can be useful to set up a graphics "hook" which does this customisation automatically. This is done as follows.

```
<<echo=false>>=
options(SweaveHooks=list(fig=function()
           par(mar=c(5.1, 4.1, 1.1, 2.1))))
@
```

The hook function (which all but eliminates the margin space above the plot) is called each time a new figure is started. You can see the effect of this in figure 1 on page 6, which is produced by the following code.

```
 > plot(1:10)
 > box("outer", "dotted")
```

The dotted box shows the outer limits of the figure.

By default, the graphics produced by Sweave are 6 inches by 6 inches. This default can be changed by using the `width` and `height` options to the code fragment which produces the plot.

```
<<fig=true,width=6,height=4>>=
⋮
@
```

The default size plots can be changed using an `\SweaveOpts` statement. For example, the statement

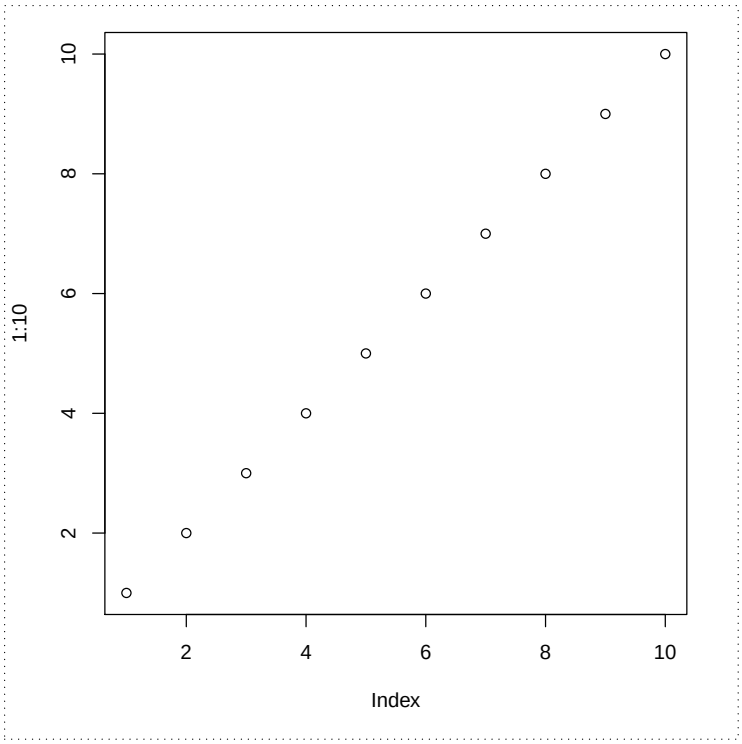

Figure 1: An example plot.

```
\SweaveOpts{width=6,height=4}
```

will make the following plots have a default width 6 inches and height 4 inches.