

The `HTMLreport()` function in the `doBy` package

Søren Højsgaard

January 3, 2011

Contents

1	Introduction	1
2	Usage	2
3	Text markup	2
3.1	Text beautifiers	3
3.2	Headings	3
3.3	Miscellaneous	4
3.4	R code	4
4	Implementation of <code>HTMLreport()</code>	4

Design library by Frank E Harrell Jr

Type `library(help='Design'), ?DesignOverview`, or `?Design.Overview'`
to see overall documentation.

1 Introduction

The `HTMLreport()` function in the `doBy` package provides facilities for translating an R-script (a file with R commands and text comments) into an HTML document. This HTML document contains the text and the R-code along with the results from executing the R-code (i.e. tables, graphics etc).

A small example is shown in Figure 1. This R-script contains R code and text comments (in the lines starting with `##`). The result from processing this R-script file is an HTML document which is shown in Figures 2, 3 and 4.

`HTMLreport()` is nowhere as flexible as using `Sweave()` with \LaTeX or using `odfWeave()` with `OpenOffice`. The facilities of `HTMLreport()` may be summarized as follows (details are provided in Section 3):

- There are a few markup facilities for the text. These are inspired by `txt2tags` markups (see <http://txt2tags.org/>).
- The specification of R-code follows the `noweb` syntax also employed by `Sweave` (see <http://www.stat.uni-muenchen.de/~leisch/Sweave/Sweave-manual.pdf>).

```
##      = HTMLreport Example 1 =
##      == The Puromycin data ==
##      === Søren Højsgaard ===
##      %%date

## === The &&Puromycin&& data ===
## The first lines of data are:

## <<>>=
head(Puromycin,3)
## @

## Transformation almost gives linearity
## <<fig=T,HTMLheight=300,HTMLwidth=600>>=
par(mfrow=c(1,2))
plot(rate~conc,          data=Puromycin, col=as.numeric(state))
plot(1/rate~I(1/conc), data=Puromycin, col=as.numeric(state))
## @

## Fit a model to transformed data
## <<>>=
ml <- lm(1/rate~state + I(1/conc) + state*I(1/conc), data=Puromycin)
summary(ml)
## @

### TODO: Maybe more could be done...
```

Figure 1: An R-script file with a few markups of text.

2 Usage

Suppose the text in Figure 1 is in the file `Example1-Puromycin.R`. Then the HTML file is created with

```
HTMLreport("Example1-Puromycin.R")

Preprocessing...
source file      : Example1-Puromycin.R
filename        : Example1-Puromycin
temp source file : file4dce21a9
Writing to file file4dce21a9.html
Processing code chunks ...
1 : term Robj
2 : echo term Robj
3 : echo term Robj png
4 : echo term Robj
file file4dce21a9.html is completed
Postprocessing...
target file     : ./Example1-Puromycin-REPORT.html
```

This creates the file `Example1-Puromycin-REPORT.html` which (by default) is located in the working directory of R.

3 Text markup

All text lines start with one or more hashes (#).

- Lines starting with one or two hashes are regarded as text which are transferred (possibly after some additional processing; see below) to the resulting HTML document.
- Lines starting with three hashes are not transferred to the HTML document. This is useful e.g. for TODOs.

HTMLreport Example 1

The Puromycin data

Søren Højsgaard

2010-12-29 22:33:33 CET

The Puromycin data

The first lines of data are:

```
> head(Puromycin, 3)
```

	conc	rate	state
1	0.02	76	treated
2	0.02	47	treated
3	0.06	97	treated

Transformation almost gives linearity

```
> par(mfrow = c(1, 2))  
  
> plot(rate ~ conc, data = Puromycin, col = as.numeric(state))  
  
> plot(1/rate ~ I(1/conc), data = Puromycin, col = as.numeric(state))
```

Figure 2: The resulting HTML document produced by `HTMLreport()`.

3.1 Text beautifiers

- Beautifiers: **boldface**, *italics*, underline, `monospace` :

These are produced with: `**boldface**`, `//italics//`, `__underline__`, `&&monospace&&`

- The beautifiers can be combined in any way, e.g. `**__some text__**`.

3.2 Headings

- Headings at different font sizes are produced with:

`= Title level 1 =`, `== Title level 2 ==`, `=== Title level 3 ===`

- The text beautifiers can be used in the headings.

3.3 Miscellaneous

- The time of creation of the HTML document is produced by `%%date`.
- All text markups must appear on one line; that one may write

```
## = HERE COMES A TITLE =
```

whereas is it is not allowed to write

```
## =  
##  HERE COMES A TITLE  
## =
```

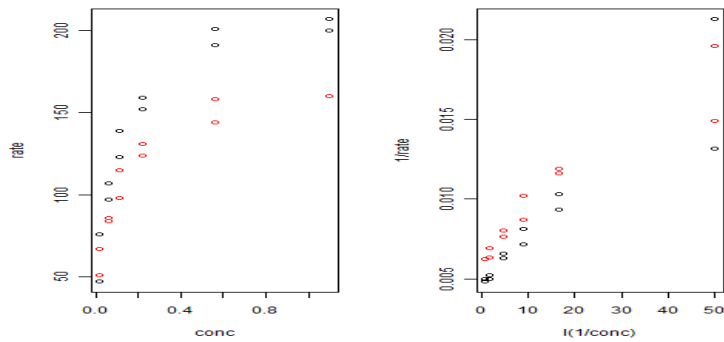
3.4 R code

- A chunk of R-code lines start with `##<<>=` and ends with `##@`.
- Various options to code chunks can be specified between `<<` and `>>=`; see the example.

4 Implementation of `HTMLreport()`

A major design goal of `HTMLreport()` has been that no additional software must be installed. `HTMLreport()` is based processing the source file line-by-line (using `gsub()`) and therefore all text markups must not be split over several lines.

The workhorse of `HTMLreport()` is the `Sweave()` function using the `RweaveHTML` driver of the `R2HTML` package.



Fit a model to **transformed** data

```
> m1 <- lm(1/rate ~ state + I(1/conc) + state * I(1/conc), data = Puromycin)
> summary(m1)
```

- Call: `lm(formula = 1/rate ~ state + I(1/conc) + state * I(1/conc), Call: data = Puromycin)`
- Residuals

Min	1Q	Median	3Q	Max
-0.0043	-0.0005	-0.0002	0.0009	0.0038

- Coefficients

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	5.11e-03	6.27e-04	8.14	1.3e-07 ***
stateuntreated	1.86e-03	9.20e-04	2.03	0.057 .
I(1/conc)	2.47e-04	2.86e-05	8.64	5.2e-08 ***
stateuntreated:I(1/conc)	-3.22e-05	4.10e-05	-0.79	0.442

Figure 3: The resulting HTML document produced by `HTMLreport()`.

--- Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

- Residuals standard error: 0.002 on 19 degrees of freedom
- Multiple R-Squared:**0.876**
- Adjusted R-Squared:**0.856**
- F-statistics: **44.614** on 3 and 19 DF. P-value:**0**.

Figure 4: The resulting HTML document produced by `HTMLreport()`.