Example of a Document for Display in a Browser

Dr David J Scott

# Entering Text

Ordinary paragraph text can be entering using hwrite. Aspects of the text such as the font family, face, size and colour can be altered in line by using tags or styles.  
  
For example the font can be changed to sans-serif like this, italic *like this,* bold **like this,** coloured like this, or any combination of these ***like this.***  
  
Some text variations may not be rendered appropriately as can be seen in the examples above. Classes can be defined in the .css file to deal with this problem.

# Incorporating Special Symbols

Using appropriate tags and a list of symbols it is possible to output mathematical and other symbols. Superscripts and subscripts are also possible.  
  
Here are some Greek letters: α β and Γ, followed by some examples of superscripts and subscripts: α2 and βn. As examples of other symbols, here are some arrows: ← and ⇒, some set symbols: ∃ and ∩, and some operators: ∫ and ∏.  
  
More than one approach is available to produce mathematical symbols. Greek letters can be produced by using symbol font: , although this may not work in some browsers without the presence of additional fonts. Codes can be used instead of `\LaTeX`-like names: ←, ∫ and ∏  
  
Quotes can be incorporated by escaping with "\", so for example we can obtain " and ' in the middle of some text.

# Rendering Mathematics

Entering more complex mathematical expressions including displayed mathematical expressions is difficult. One approach is to insert an image produced by other means, for example by `\LaTeX` or Word's equation editor. The following expression was produced by Word's equation editor then saved as a jpeg.

DisplayedEquation.jpg

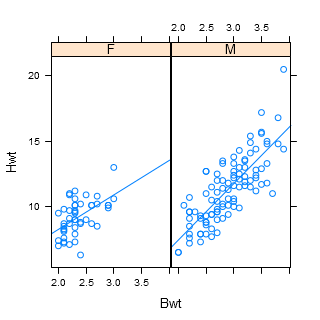
To enable the rendering of complex mathematical expressions using `\LaTeX` syntax, JavaScript is required, the JavaScript being run when the file is loaded into the browser. The package R2HTML uses MathPlayer to do the rendering. An alternative is MathJax which is used by org-babel. In this example MathPlayer is used. This is an extension to the package hwriter.  
  
Here is an inline expression:, followed by two displayed expressions, one of which is numbered and labelled. The first example also has a box around it, by assigning the value "border = '1'" to the argument table.attributes.

|  |
| --- |
|  |

Equation - equation number and label

|  |  |  |
| --- | --- | --- |
|  |  |  |

# Incorporating Images

Incorporating images is complicated by the inability of different browsers to display images in different formats. Firefox will not display .wmf images for example. The only scalable image format which can be displayed by all the common browsers (Internet Explorer, Firefox, Safari and Chrome) appears to be .svg, scalable vector graphics.  
  
Here is a windows metafile, .wmf image. Unfortunately this will only display in Internet Explorer, and there will be no output produced at all if this example is run on linux. The example is the cats data used by Leisch as an Sweave example, taken from Venables and Ripley (1987). The data frame contains measurements of heart and body weight of 144 cats (47 female, 47 male).  
  
A linear regression model of heart weight by sex and gender was fitted to this data. The graph is a scatter plot of the data including the regression lines  
  
cats.wmf  
Here is the cats data plot in svg format. This uses an extension to hwriter to produce html code which enables display in up to date versions of all common browsers.  
-->   
A further format is png (portable network graphics). This should display in up to date versions of all common browsers. It is a bitmap format however so not scalable.  
  
One problem with images is that the size of the image displayed can vary widely from browser to browser. Obtaining the right sized image for a particular browser may require a lot of trial and error.

# Vectors, Matrices and Dataframes

Here is some code producing a character vector

form <- y ~ a + b + c

example <- as.character(form)

This is the result of printing the vector

|  |  |  |
| --- | --- | --- |
| ~ | y | a + b + c |

Here is some code producing a numeric vector

y <- 3\*(1:5)

This is the result of printing the vector

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | 6 | 9 | 12 | 15 |

Here is some code producing a matrix

mdat <- matrix(c(1,2,3, 11,12,13),

nrow = 2, ncol = 3, byrow = TRUE,

dimnames = list(c('row1', 'row2'),

c('C.1', 'C.2', 'C.3')))

This is the result of printing the matrix

|  |  |  |  |
| --- | --- | --- | --- |
|  | C.1 | C.2 | C.3 |
| row1 | 1 | 2 | 3 |
| row2 | 11 | 12 | 13 |

Here is some code producing a dataframe

L3 <- LETTERS[1:3]

d <- data.frame(cbind(x = 1, y = 1:10),

fac = sample(L3, 10, replace = TRUE))

This is the result of printing the dataframe

|  |  |  |
| --- | --- | --- |
| x | y | fac |
| 1 | 1 | B |
| 1 | 2 | B |
| 1 | 3 | A |
| 1 | 4 | C |
| 1 | 5 | A |
| 1 | 6 | A |
| 1 | 7 | B |
| 1 | 8 | C |
| 1 | 9 | B |
| 1 | 10 | B |

## Tables

Here is an example taken from the documentation of xtable. It shows the output of an ANOVA table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Df** | **Sum Sq** | **Mean Sq** | **F value** | **Pr(> F)** |
| sex | 1 | 75.37 | 75.37 | 0.38 | 0.5417 |
| ethnicty | 3 | 2572.15 | 857.38 | 4.27 | 0.0072 |
| grade | 1 | 36.31 | 36.31 | 0.18 | 0.6717 |
| disadvg | 1 | 59.30 | 59.30 | 0.30 | 0.5882 |
| Residuals | 93 | 18682.87 | 200.89 |  |  |

Here is an example taken from the documentation of table. It shows the output of a two-dimensional table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Northeast** | **South** | **North Central** | **West** |
| New England | 6 | 0 | 0 | 0 |
| Middle Atlantic | 3 | 0 | 0 | 0 |
| South Atlantic | 0 | 8 | 0 | 0 |
| East South Central | 0 | 4 | 0 | 0 |
| West South Central | 0 | 4 | 0 | 0 |
| East North Central | 0 | 0 | 5 | 0 |
| West North Central | 0 | 0 | 7 | 0 |
| Mountain | 0 | 0 | 0 | 8 |
| Pacific | 0 | 0 | 0 | 5 |

## R Output

To include output from R in a file, the command capture.output is used to record the output. Then the output is included in the html file by using the command hwriteOutput, which is not in the package hwriter.  
Here is an example from the glm help.

glmOut <- capture.output(

clotting <- data.frame(

u = c(5,10,15,20,30,40,60,80,100),

lot1 = c(118,58,42,35,27,25,21,19,18),

lot2 = c(69,35,26,21,18,16,13,12,12)),

summary(glm(lot1 ~ log(u), data=clotting, family=Gamma)),

summary(glm(lot2 ~ log(u), data=clotting, family=Gamma)))

This produces the following result.

Call:

glm(formula = lot1 ~ log(u), family = Gamma, data = clotting)

Deviance Residuals:

Min 1Q Median 3Q Max

-0.04008 -0.03756 -0.02637 0.02905 0.08641

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -0.0165544 0.0009275 -17.85 4.28e-07 \*\*\*

log(u) 0.0153431 0.0004150 36.98 2.75e-09 \*\*\*

---

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

(Dispersion parameter for Gamma family taken to be 0.002446059)

Null deviance: 3.51283 on 8 degrees of freedom

Residual deviance: 0.01673 on 7 degrees of freedom

AIC: 37.99

Number of Fisher Scoring iterations: 3

Call:

glm(formula = lot2 ~ log(u), family = Gamma, data = clotting)

Deviance Residuals:

Min 1Q Median 3Q Max

-0.05574 -0.02925 0.01030 0.01714 0.06371

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -0.0239085 0.0013265 -18.02 4.00e-07 \*\*\*

log(u) 0.0235992 0.0005768 40.91 1.36e-09 \*\*\*

---

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

(Dispersion parameter for Gamma family taken to be 0.001813354)

Null deviance: 3.118557 on 8 degrees of freedom

Residual deviance: 0.012672 on 7 degrees of freedom

AIC: 27.032

Number of Fisher Scoring iterations: 3

Note the commas separating the parts of the output to be captured. See the help and examples for capture.output for details.

## An R Session

To capture an R session, or part of one, including both commands and output, the command txtStart from the package TeachingDemos can be used then hwriteOutput. This requires writing to a file, and reading the results back from the file. A temporary file can used for holding the output. See ?tempfile.  
Here is an example.

> clotting <- data.frame(u = c(5, 10, 15, 20, 30, 40, 60, 80, 100),

+ lot1 = c(118, 58, 42, 35, 27, 25, 21, 19, 18), lot2 = c(69,

+

+ 35, 26, 21, 18, 16, 13, 12, 12))

> clotting

u lot1 lot2

1 5 118 69

2 10 58 35

3 15 42 26

4 20 35 21

5 30 27 18

6 40 25 16

7 60 21 13

8 80 19 12

9 100 18 12

> summary(glm(lot1 ~ log(u), data = clotting, family = Gamma))

Call:

glm(formula = lot1 ~ log(u), family = Gamma, data = clotting)

Deviance Residuals:

Min 1Q Median 3Q Max

-0.04008 -0.03756 -0.02637 0.02905 0.08641

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -0.0165544 0.0009275 -17.85 4.28e-07 \*\*\*

log(u) 0.0153431 0.0004150 36.98 2.75e-09 \*\*\*

---

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

(Dispersion parameter for Gamma family taken to be 0.002446059)

Null deviance: 3.51283 on 8 degrees of freedom

Residual deviance: 0.01673 on 7 degrees of freedom

AIC: 37.99

Number of Fisher Scoring iterations: 3

## Creating Links

Since html is being produced, it is easy to create links to other websites. Here is an example of a link to the Statistics Department website: [The Department of Statistics.](http://www.stat.auckland.ac.nz/uoa/)  
  
Links may be created within pages using anchors. Destination anchors in HTML documents may be specified either by the a element (naming it with the name attribute), or by any other element (naming with the id attribute). Here is a link to the first section created using the a element: [Entering Text.](#intro)  
  
Here is a link to the third section created by naming with the id attribute: [Rendering Mathematics.](#mathematics)  
  
This use of named or identified elements of a documents is how cross-referencing can be implemented.  
  
The numbered equation entered previously can be cross-referenced using the link argument to hwrite. That equation is equation 1 at present. Here is the link to the equation: [Numbered Equation.](#equation1)  
  
The graph in SVG format included previously can also be cross-referenced since it was assigned an identifier using the id attribute. Here is a link to that graph: [Graph in SVG format.](#catsSVG)  
  
  
  
(Page generated on Thu Dec 01 15:11:23 2011 by [hwriter](http://www.embl.de/~gpau/hwriter/index.html) 1.3)