A Test of integrating bookdown with Travis CI

Markus Meister

2020-03-06

Contents

1 Prerequisites **5**

2 Equations **7**

2.1 Inline equations 7

2.2 Display equations 7

2.3 Equation labels 7

2.4 Equation numbering 8

2.5 Multi-line equation with multiple labels 9

3 Python **11**

3.1 A normal R code chunk 11

3.2 Modify an R variable 11

3.3 A Python chunk 11

3.4 Modify a Python variable 12

3.5 Python graphics 12

3.6 Using the mib plotting untility 14

Chapter 1

Prerequisites

```
install.packages('tinytex')
```

```
## Installing package into '/home/travis/R/Library'
## (as 'lib' is unspecified)
```

```
tinytex::install_tinytex()
```

```
## Warning: Detected an existing tlmgr at /home/travis/bin/tlmgr. It seems TeX
## Live has been installed (check tinytex::tinytex_root()). You are recommended
## to uninstall it, although TinyTeX should work well alongside another LaTeX
## distribution if a LaTeX document is compiled through tinytex::latexmk().

## TinyTeX installed to /home/travis/.TinyTeX

## You may have to restart your system after installing TinyTeX to make sure ~/bin appears in you
```

```
projdir = rprojroot::find_rstudio_root_file() # project directory

include_svg = function(path) { # used to convert svg to pdf
  if (knitr::is_latex_output()) {
    output = xfun::with_ext(path, 'pdf')
    # you can compare the timestamp of pdf against svg to avoid conversion if necessary
    system2('rsvg-convert', c('-f', 'pdf', '-a', '-o', shQuote(c(output, path))))
  } else {
    output = path
  }
  knitr::include_graphics(output)
}
```

```r
# connect to python
library(reticulate)
matplotlib <- import("matplotlib")
matplotlib$use("Agg", force = TRUE) # backend that doesn't expect a display
```

```python
import numpy as np
import os
import sys
import matplotlib
matplotlib.rcParams['text.usetex'] = True # use Latex to draw all text
matplotlib.rcParams['text.latex.preamble'] = [r'\usepackage{amsmath}']
import matplotlib.pyplot as plt
curdir = os.getcwd()
os.chdir(r.projdir)
from mib.utils import plot, save_img # plotting routines
os.chdir(curdir)
```

Chapter 2

Equations

2.1 Inline equations

...are enclosed by simple `$ $`, like this:

```
$\tilde h(\omega) = \int_{-\infty}^{\infty}\,e^{i\omega t'} h(t') \, dt'\,$
```

which produces this output: $\tilde h(\omega) = \int_{-\infty}^{\infty}\,e^{i\omega t'} h(t') \, dt'\,$.

2.2 Display equations

...without numbers can be enclosed by double `$$ $$`, like this:

```
$$\tilde h(\omega) = \int_{-\infty}^{\infty}\,e^{i\omega t'} h(t') \, dt'\,.$$
```

which produces

$$\tilde h(\omega) = \int_{-\infty}^{\infty}\,e^{i\omega t'} h(t') \, dt'\,.$$

2.3 Equation labels

To label an equation with `name` use the format `(\#eq:name)`. To cite that equation use the format `\@ref(eq:name)`

The equation label has to appear after the body of the equation code, like this:

```
\begin{equation}
  \tilde h(\omega) = \int_{-\infty}^{\infty}\,e^{i\omega t'} h(t') \, dt'
  (\#eq:binom)
\end{equation}
```

$$\tilde{h}(\omega) = \int_{-\infty}^{\infty} e^{i\omega t'}\, h(t')\, dt' \tag{2.1}$$

```
\begin{equation}
f\left(k\right)=\binom{n}{k}p^k\left(1-p\right)^{n-k} (\#eq:binom2)
\end{equation}
```

$$f\left(k\right)=\binom{n}{k}p^k\left(1-p\right)^{n-k} \tag{2.2}$$

Then you can cite the equation, like this: (2.1).

Changed this 5:11PM.

2.4 Equation numbering

There is some weirdness about how equation numbering is handled in the PDF versus the HTML versions of the book.

In the PDF output equations are numbered by default. Every line of an equation will get numbered except if it

1. is inside `$$ $$`.
2. has `\notag` at the end of line, before the `\\`.
3. is in an `{equation*}` or `{align*}` environment where there are no labels
4. is in a `{split}` environment with a single label

But in the HTML output an equation is unnumbered by default, except if it contains an explicit equation label. For more details see here.

So, to make sure we get the same numbering in PDF and HTML we should do this:

An unnumbered display equation should be enclosed with `$$ $$` or in environments `{equation*}` or `{align*}`.

A numbered display equation should include a single label.

Here are some examples:

This equation will get a number in PDF but not in HTML, **which is a mistake!**

$$\tilde{h}(\omega) = \int_{-\infty}^{\infty} e^{i\omega t'} h(t')\, dt' \tag{2.3}$$

This gets no number in PDF or HTML:

$$\tilde{h}(\omega) = \int_{-\infty}^{\infty} e^{i\omega t'} h(t')\, dt'$$

And this gets a number in both PDF and HTML:

$$\tilde{h}(\omega) = \int_{-\infty}^{\infty} e^{i\omega t'} h(t')\, dt' \tag{2.4}$$

2.5 Multi-line equation with multiple labels

Here is a long equation stretching over several lines, first with a single number

$$\begin{aligned}
\mathrm{Var}(\hat{\beta}) &= \mathrm{Var}((X'X)^{-1}X'y) \\
&= (X'X)^{-1}X'\mathrm{Var}(y)((X'X)^{-1}X')' \\
&= (X'X)^{-1}X'\mathrm{Var}(y)X(X'X)^{-1} \\
&= (X'X)^{-1}X'\sigma^2 IX(X'X)^{-1} \\
&= (X'X)^{-1}\sigma^2
\end{aligned} \tag{2.5}$$

...then with multiple numbers

$$\begin{aligned}
\mathrm{Var}(\hat{\beta}) &= \mathrm{Var}((X'X)^{-1}X'y) \\
&= (X'X)^{-1}X'\mathrm{Var}(y)((X'X)^{-1}X')' && (2.6) \\
&= (X'X)^{-1}X'\mathrm{Var}(y)X(X'X)^{-1} && (2.7) \\
&= (X'X)^{-1}X'\sigma^2 IX(X'X)^{-1} && (2.8) \\
&= (X'X)^{-1}\sigma^2 && (2.9)
\end{aligned}$$

I wrote some grep routines that produce automatic equation labels, for example in the study guide which contains several hundred equations. Every one of those has a label, even though we rarely cite those labels.

Chapter 3

Python

3.1 A normal R code chunk

```
x = 42
print(x)
```

```
## [1] 42
```

3.2 Modify an R variable

In the following chunk, the value of `x` on the right hand side is 42, which was defined in the previous chunk.

```
x = x + 12
print(x)
```

```
## [1] 54
```

3.3 A Python chunk

This works fine and as expected.

```
print('Python version = ', sys.version)
```

```
## Python version =  3.5.2 (default, Nov 12 2018, 13:43:14)
## [GCC 5.4.0 20160609]
```

```
x = 42 * 2
print(x)
```

```
## 84
```

The value of `x` in the Python session is 84. It is not the same `x` as the one in R.

3.4 Modify a Python variable

```
x = x + 18
print(x)
```

```
## 102
```

Retrieve the value of `x` from the Python session again:

```
py$x
```

```
## [1] 102
```

Assign to a variable in the Python session from R:

```
py$y = 1:5
```

See the value of `y` in the Python session:

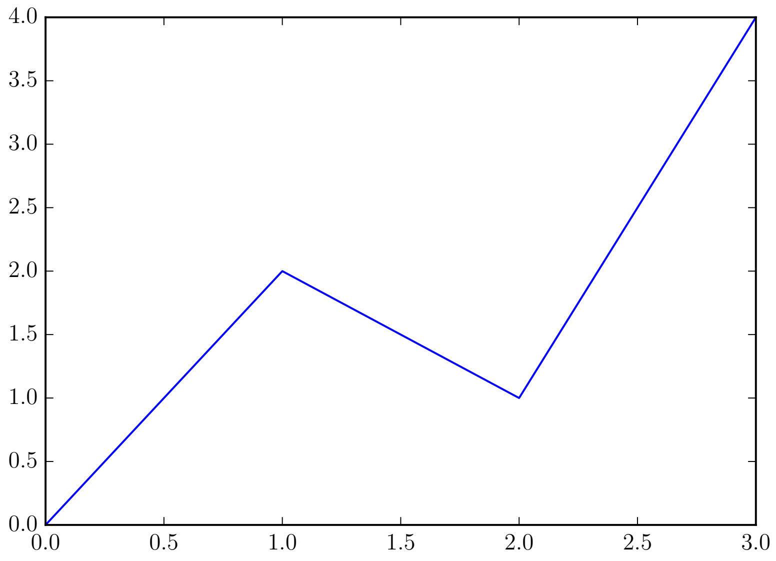
```
print(y)
```

```
## [1, 2, 3, 4, 5]
```

3.5 Python graphics

You can draw plots using the **matplotlib** package in Python.

```
import matplotlib.pyplot as plt
plt.plot([0, 2, 1, 4])
plt.show()
```



and adding a figure caption

~~~
```{python testfig1, echo=FALSE, out.width='33%', fig.align='center', fig.cap = 'A figure from Py
plt.plot([0, 2, 1, 4])
plt.show()
```
~~~

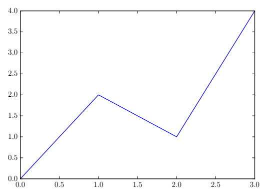


Figure 3.1: A figure from Python code

Note this last chunk fails when knitr version is knitr_1.28, but works with knitr_1.26.

Using an R chunk to insert a figure with caption:

~~~
```{r testfig2, echo=FALSE, out.width='33%', fig.align='center', fig.cap='A figure fron
knitr::include_graphics('img/testfig1.pdf')
```
~~~

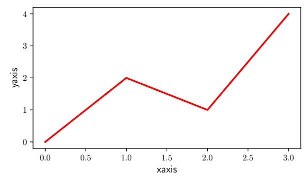


Figure 3.2: A figure from a PDF file.

Can cite both Figs. 3.1 and 3.2.

3.6 Using the mib plotting untility

```
x = np.arange(-10,10,.01)
t = [0.1,0.3,1,3,10]
y = [np.exp(-x**2/(4*ti))/np.sqrt(4*np.pi*ti) for ti in t]
plot (x,y,fmts=['g-', 'm--', 'b-.', 'r:', 'k-'], linewidth=1,
    xlabel='Distance',ylabel='Concentration',yzero=True,
    legend=['t = 0.1','0.3','1','3','10'],xticks=[-10,-5,-1,0,1,5,10])
save_img('plotP1.pdf')
plt.show()
```

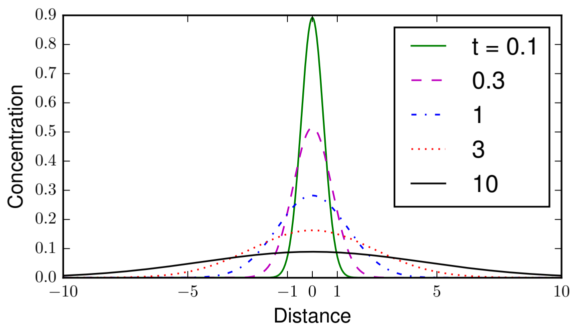


Figure 3.3: A plot using mib/utils.py

See Figure 3.3.