R Markdown with LaTeX

Rob Donald

Tuesday 02 July 2019

Contents

Introduction

This set of notes shows some examples of using LaTeX with R markdown.

The idea is to give examples which help if you are reasonably comfortable with R markdown but want *some* of the customisation of using LaTeX without having to become a LaTeX guru [1]. The code that generates the .pdf output is in the .Rmd file in this repo.

What is LaTeX?

Well if you have been 'Knit'ing to PDF inside RStudio you have been using LaTeX all the time. You may have already used things like \newpage to tidy up a document. The \newpage is actually a LaTeX command which is being detected by Knitr as it processes the document.

LaTeX is used extensively in academia particularly in any subject that requires a lot of maths notation. It is a different way of thinking about documents. You write your document in plain text and add in the various commands that format the document into something that looks nice. It means you can produce a PDF document with formulas and figures and tables without ever going near Word. But it is not for everyone. If you need a WYSIWYG editor then this is not for you.

In the past you would need to use a LaTeX editor of some kind (e.g. the cross platform open source program TeXMaker). With the advent of RStudio and R Markdown (*which I assume you are at least partly familiar with*) you can now write well formatted PDF documents with very little knowledge of LaTeX.

[1] Although if you do get proficient in this stuff you may well get the guru title

This set of notes will show you how you can extend your documents by using a small amount of LaTeX code in the text portion of your .Rmd R markdown document.

NOTE: if you go down this route you can't flip back to HTML output. The Knitr package will get a bit upset and probably miss out some of the LaTeX commands. It will look like it has worked, there will just be missing bits (give it a try . . .)

Some History

LaTeX is actually a typesetting programming language and was developed by Leslie Lamport in 1983. See this link for a more detailed description LaTeX History.

It is a layer of code over the top of the typesetting language TEXwhich was developed by the famous computer scientist Donald Knuth in 1978. See this link for more details Tex History

Useful YouTube links

- LaTeX Tutorial 1 - Creating a LaTeX Document

The above link is the start of a very useful series of videos by Michelle Krummel.

Font Sizes

Some Large (with a capital 'L') text

Some large text

Some normalsize text

Compare this default R markdown line to the above 'normalsize' text.

Some small text which runs over several lines of text and allows us to put in more information in a smaller space. Which can be useful if the information is very dense.

This is footnotesize. Useful size for tables and other hard to fit in things.

Size and Position of Figures

This can be really awkward as LATEX often thinks *it* knows best.

The web is full of stuff that doesn't work for output to .pdf documents. Here are some commands to try for controlling the position and size of figures.

Size Using out.width and out.height

The technique I use all the time is to import the picture into the document with an R chunk. Inside the R chunk you use:

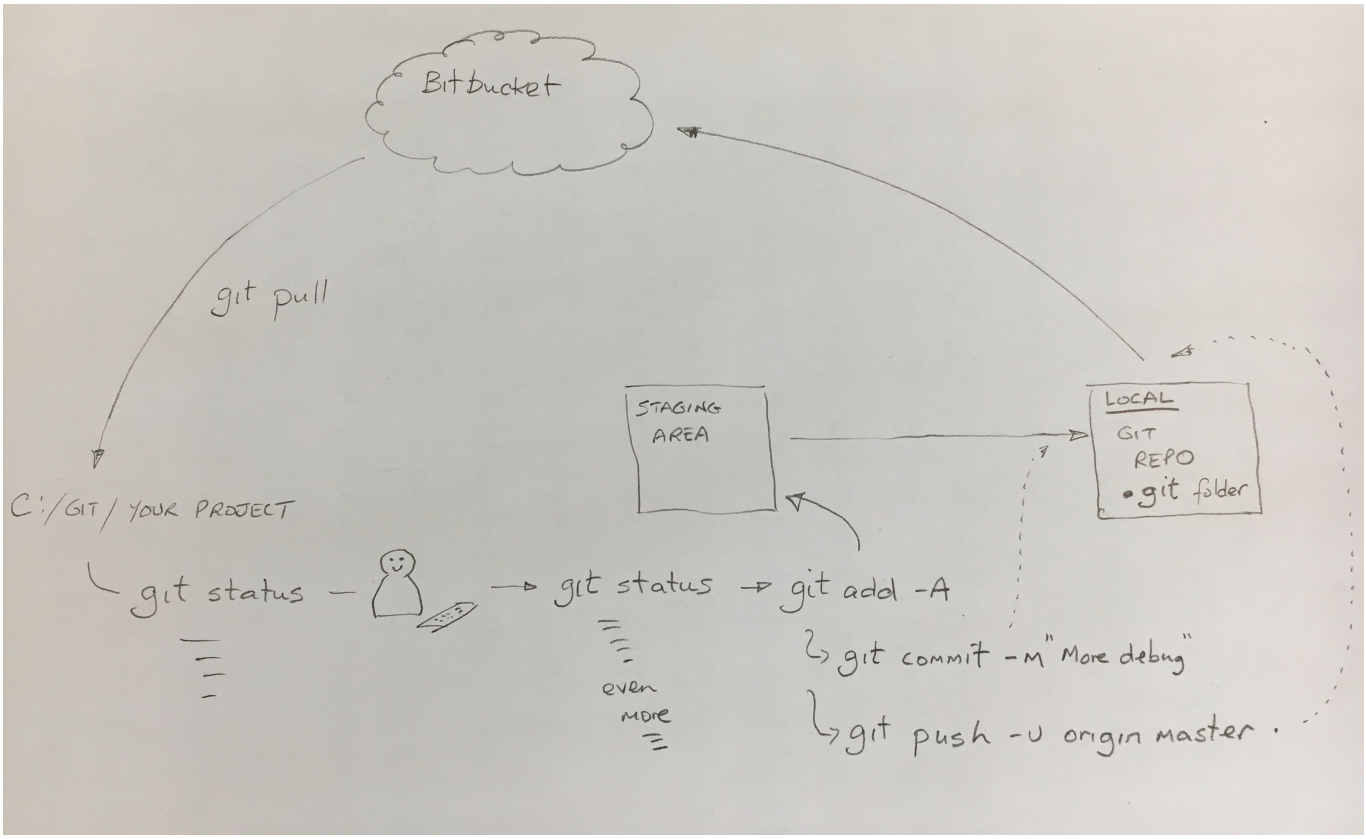
- knitr::include_graphics()

You add options to the chunk header.

```
{r, out.width='18cm',out.height='12cm'}
```

This gives:

```
image.path <- here('images','GitWorkflow.png')
knitr::include_graphics(path = image.path)
```



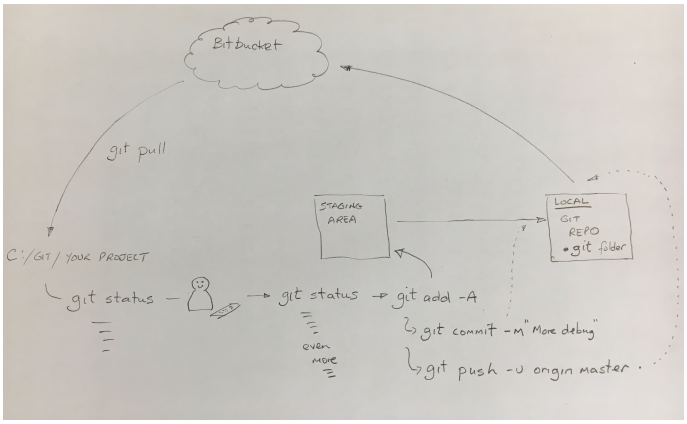
The 18 cm seems to work for an A4 page which is 21 cm wide. You can adjust the height to suit your image.

Let's half the size.

```
{r, out.width='9cm',out.height='6cm'}
```

This gives:

```
image.path <- here('images','GitWorkflow.png')
knitr::include_graphics(path = image.path)
```

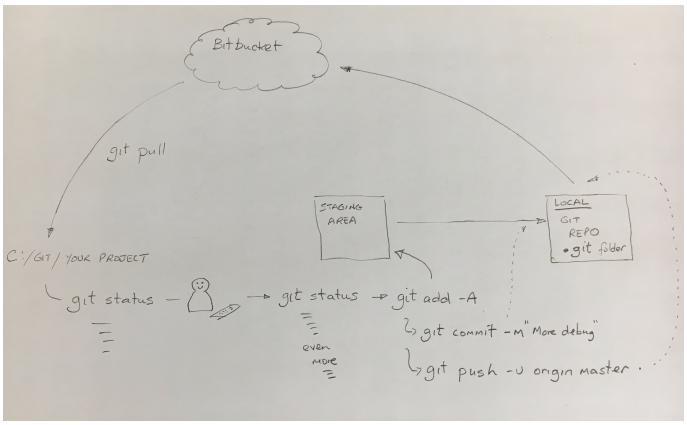


Centre the Image

Now we can try to centre the image. The next two attempts ***don't*** work for me despite lots of the internet telling you they will.

\begin{center}

```
image.path <- here('images','GitWorkflow.png')
knitr::include_graphics(path = image.path)
```



\end{center}

Directly in the text part of the R markdown.

\begin{center} \include_graphics('images/GitWorkflow.png') \end{center}

This doesn't even produce the plot!

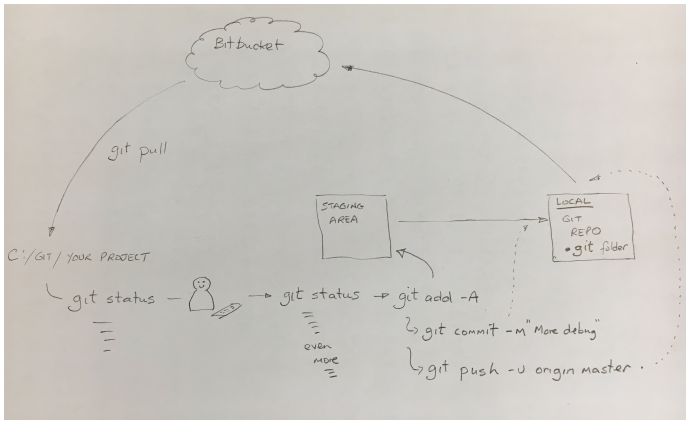
Using fig.align="center"

This technique *does* work. Add this into the chunk header.

`{r, out.width='9cm',out.height='6cm',fig.align="center"}`

This gives:

```
image.path <- here('images','GitWorkflow.png')
knitr::include_graphics(path = image.path)
```



Here is a link to a constantly referenced article on the web about this stuff. Note that it is biased towards HTML output ZevRoss Article

---

So let's add a 'Large' title and caption

Git Workflow

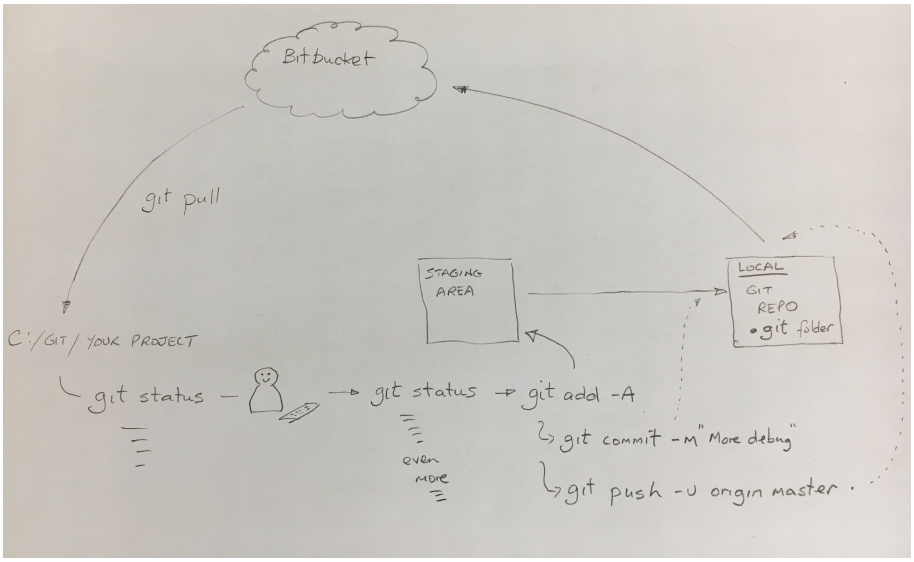


Figure1: The above diagram shows a typical Git workflow using a cloud based Git provider (e.g. Bitbucket or Github) as the central repositry for the team's work.

Centred Text

Some centred text on a line

References

This is an example of creating a references page.

Footnote Size Links

- [1] https://www.stata.com/manuals13/pss.pdf (Accessed 2019-05-02)
- [2] https://en.wikipedia.org/wiki/Sample_size_determination (Accessed 2019-05-03)
- [3] https://en.wikipedia.org/wiki/Power_(statistics) (Accessed 2019-05-03)
- [4] https://en.wikipedia.org/wiki/Effect_size (Accessed 2019-05-03)
- [5] https://en.wikipedia.org/wiki/Test_statistic (Accessed 2019-05-03)
- [6] https://blog.minitab.com/blog/adventures-in-statistics-2/understanding-analysis-of-variance-anova-and-the-f-test (Accessed 2019-05-03)