Doing Open, Transparent, and Reproducible Research with RMarkdown and Git (and friends)[1]

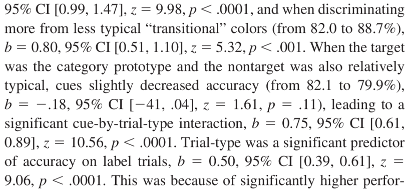
Mark Andrews
Psychology Department, Nottingham Trent University

`@xmjandrews`
`mark.andrews@ntu.ac.uk`
`https://github.com/mark-andrews/sips2019`

[1] These slides are not intended to be self-contained and comprehensive, but just aim to provide some of the workshop's content. Much more will be provided in the workshop itself.

What is reproducible research?

Or rather, what is reproducible data analysis?



95% CI [0.99, 1.47], $z = 9.98$, $p < .0001$, and when discriminating more from less typical "transitional" colors (from 82.0 to 88.7%), $b = 0.80$, 95% CI [0.51, 1.10], $z = 5.32$, $p < .001$. When the target was the category prototype and the nontarget was also relatively typical, cues slightly decreased accuracy (from 82.1 to 79.9%), $b = -.18$, 95% CI [−41, .04], $z = 1.61$, $p = .11$), leading to a significant cue-by-trial-type interaction, $b = 0.75$, 95% CI [0.61, 0.89], $z = 10.56$, $p < .0001$. Trial-type was a significant predictor of accuracy on label trials, $b = 0.50$, 95% CI [0.39, 0.61], $z = 9.06$, $p < .0001$. This was because of significantly higher perfor-

Figure 1: Part of a Results section in Forder & Lupyan (2019).

- Any research that involves statistical data analysis will usually contain many figures and tables of statistical results, and also numerous statistical results within the text.
- The goal of reproducible research — or rather, *reproducible data analysis* — is that anyone working independently could recreate all of these results exactly.

Necessary criteria for reproducible data analysis

- The following three criteria seem necessary for a given data analysis to be reproducible.

  1. The *raw* data must be available. Data that is processed and "cleaned up" is not sufficient.
  2. All the code for all the analysis must be available. All the code for all the data analysis pipeline is required, as are the scripts and build tools that execute the code.
  3. The reports of the analysis, e.g., journal articles, presentations slides, etc, must be made by *dynamic documents*.

- Gentleman & Temple Lang (2007) introduced the concept of a *research compendium*, which is a single package that contains all of the raw data, all the code for all the data analysis pipeline, and dynamic documents that generate all the final reports.

Software tools for reproducible data analysis

- There are numerous (open source, or freely available) software tools and service that facilitate creating and maintaining of a research compendium. These include the following:
  - RMarkdown (and knitr, pandoc, $\LaTeX$, etc)
  - Git & GitHub
  - Make (and other build automation tools)
  - Jupyter
  - Docker and virtual machines
  - Git LFS, Git annex, Git fat, etc
- Here, we will deal with just the first two items on this list.

What is RMarkdown?

- RMarkdown is an R based dynamic document format. It is used with knitr to generate documents in different formats that combine text with content, including figures, tables, etc., that are dynamically generated by R (or other languages).
- It can be used to create publication ready manuscripts, slides for presentations, scientific posters.
- It, and its variants `bookdown`, `blogdown`, `pkgdown`, etc, can be used to create books, websites, interactive online demos and tutorials, etc.

RMarkdown overview: Example 1

We write source code that is mixture of R code and explanatory text that optionally references the R variables.

~~~
```{r}
set.seed(101)
N <- 50
mu <- 100
sigma <- 15
x <- rnorm(N, mean=mu, sd=sigma)
```

The mean of a random sample of `r N` numbers,
drawn independently from a normal distribution
with mean `r mu` and standard deviation `r sigma`,
is `r round(mean(x), 2)`.
~~~

RMarkdown overview: Example 1 (rendered)

When we render this, we'll produce a document (in this case, LaTeX) with both the code and any output and any evaluated variables in the text.

```r
set.seed(101)
N <- 50
mu <- 100
sigma <- 15
x <- rnorm(N, mean=mu, sd=sigma)
```

The mean of a random sample of 50 numbers, drawn independently from a normal distribution with mean 100 and standard deviation 15, is 98.14.

RMarkdown overview: Example 2

We may turn off the rendering of the R source code with `echo = FALSE`.

~~~
```{r, echo=FALSE}
set.seed(101)
N <- 50
mu <- 100
sigma <- 15
x <- rnorm(N, mean=mu, sd=sigma)
```

The mean of a random sample of `r N` numbers,
drawn independently from a normal distribution
with mean `r mu` and standard deviation `r sigma`,
is `r round(mean(x), 2)`.
~~~

RMarkdown overview: Example 2 (rendered)

Then we get e.g. just the rendered text, but not the R *chunk*.

The mean of a random sample of 50 numbers, drawn independently from a normal distribution with mean 100 and standard deviation 15, is 98.14.

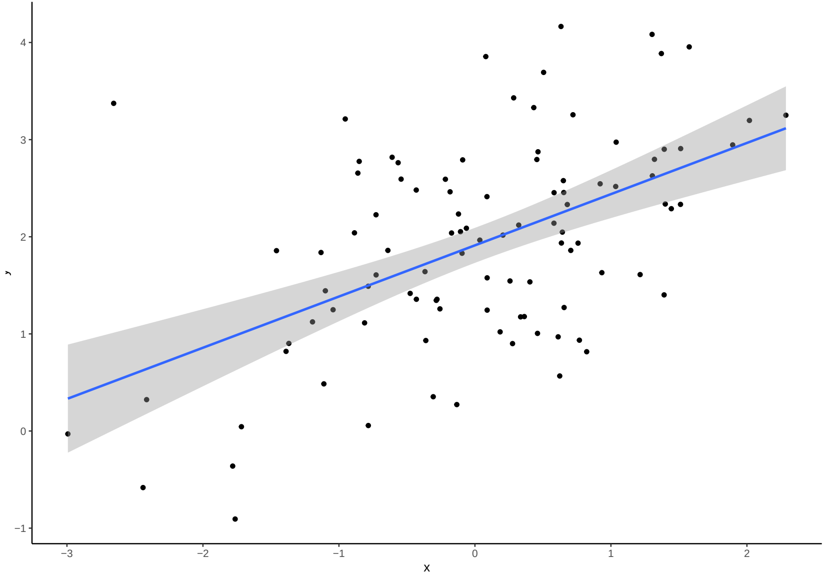
RMarkdown overview: Example 3

Figures will be rendered and inserted into the document in an identical manner.

~~~
```{r, echo=FALSE}
set.seed(42)
N <- 100
x <- rnorm(N)
Df <- data.frame(x = x,
                 y = 2 + 0.5*x + rnorm(N))
ggplot(Df,
       mapping = aes(x=x, y=y)) +
  geom_point() +
  stat_smooth(method='lm') +
  theme_classic()

```
~~~

RMarkdown overview: Example 3 (rendered)



RMarkdown overview: Example 4

Likewise, tables from statistical models can be rendered and inserted into the document.

~~~
```{r, echo=FALSE}
set.seed(42)
N <- 100
x <- rnorm(N)
Df <- data.frame(x = x,
                 y = 0.0 + 0.25*x + rnorm(N))

M <- lm(y ~ x, data=Df)
pander(summary(M))
```
~~~

RMarkdown overview: Example 4 (rendered)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-0.088	0.091	-0.972	0.333
x	0.277	0.088	3.162	0.002

Table 2: Fitting linear model: y ~ x

Observations	Residual Std. Error	R^2	Adjusted R^2
100	0.908	0.093	0.083

RMarkdown overview: Example 5

RMarkdown allows us to typeset mathematical equations, symbols, etc., just as we would do with LaTeX.

~~~
```{r, echo=FALSE}
set.seed(42)
N <- 100
x <- rnorm(N)
Df <- data.frame(x = x,
                 y = 0.0 + 0.25*x + rnorm(N))
M <- lm(y ~ x, data=Df)
```


The linear model is
$$
y_i = \alpha + \beta x_i + \epsilon_i,
\quad \text{for $i \in 1 \ldots N$}.
$$

The $R^2$ value is `r round(mean(summary(M)$r.sq),2)`.
~~~

RMarkdown overview: Example 5 (rendered)

The linear model is

$$y_i = \alpha + \beta x_i + \epsilon_i, \quad \text{for } i \in 1 \ldots N.$$

The R^2 value is 0.09.

What we'll cover in this workshop

More or less in this order too

- How to create an RMarkdown document from scratch, including the basics of the RMarkdown YAML header.
- How to *knit* rmarkdown documents with R chunks and inline R code.
- The details of the markdown syntax.
- How to include math(s) using LaTeX code.
- How to include figures and tables and control their presentation.
- How to include citations.
- How to use different document templates, include `papaja`.
- How to make LaTeX beamer[2] and html based slides.

[2] These slides are made with RMarkdown rendering to its LaTeX beamer output.

What is Git?

- Git is version control software, initially developed for version control of the Linux operating system kernel.
- It is now extremely widely used for almost all kinds of software development projects.
- Git works on a decentralized system whereby a code-base can be *cloned*, developed independently, and possible re-merged.
- For collaborating on one project, two developers use a *remote* host, clone it, develop locally, *commit* and then *push* back to and *pull* from the remote host.
- GitHub is one of the most widely used hosting sites (but there are others, e.g. BitBucket; and running your own git hosting server is simple and inexpensive).

What should we care?

- Git (and a Git server) provides a means of sharing your research compendium and its updates, and this can be done from the beginning of the research and continually throughout its entire development.
- But it's not just another "dropbox": It is distributed version control system.
    - It allows open, distributed, collaborative development of research and data analysis code, like the open source development model for software generally.
    - Even for small teams, it provides a means for efficient and organized development of source code, including (and especially) for writing the source code for our articles, slides, etc (no more of this: `http://phdcomics.com/comics/archive_print.php?comicid=1531`).
    - Even for individuals, it allows for efficient and organized development.
    - And it is (almost) endlessly fault tolerant. So long as there is one clone somewhere, the entire project and its version history is preserved entirely.

Git: Tiny tutorial

- Start by cloning a remote repository:

```
git clone https://github.com/yihui/knitr.git
cd knitr
git log # Read all the commit logs
```

- Work as normal, i.e. edit files, create new files, delete files.
- You now *stage* your changes, e.g.

```
git add foo.file.1 foo.file.2 # for edits or new files
git rm foo.file.3 # for removed files
```

- You then *commit* these:

```
git commit # Editor opens for your log msg
```

Git: Tiny tutorial (2)

- Pull down any recent changes by others from the remote:

```
git pull
git log # If new changes, read their logs
```

- Now, push your own changes to the remote

```
git push # requires permissions
```

- Undo changes:

```
git reset a381f2f # move back "head"
git revert a381f2f # applies new change to revert
```

References

Forder, L., & Lupyan, G. (2019). Hearing words changes color perception: Facilitation of color discrimination by verbal and visual cues. *Journal of Experimental Psychology: General*, *148*(7), 1105.

Gentleman, R., & Temple Lang, D. (2007). Statistical analyses and reproducible research. *Journal of Computational and Graphical Statistics*, *16*(1), 1–23.