Reproducible Analyses with knitr and rmarkdown

Michael Sachs

April 1, 2015

Introduction

Reproducibility

NIH plans to enhance reproducibility

Francis S. Collins and **Lawrence A. Tabak** discuss initiatives that the US National Institutes of Health is exploring to restore the self-correcting nature of preclinical research.

A growing chorus of concern, from scientists and laypeople, contends that the complex system for ensuring the reproducibility of biomedical research is failing and is in need of restructuring[1,2]. As leaders of the US National Institutes of Health (NIH), we share this concern and here explore some of the significant interventions that we are planning.

Science has long been regarded as 'self-correcting', given that it is founded on the replication of earlier work. Over the long term, that principle remains true. In the shorter term, however, the checks and balances that once ensured scientific fidelity have been hobbled. This has compromised the ability of today's researchers to reproduce others' findings.

Let's be clear: with rare exceptions, we have no evidence to suggest that irreproducibility is caused by scientific misconduct. In 2011, the Office of Research Integrity of the US Department of Health and Human Services pursued only 12 such cases[3]. Even if this represents only a fraction of the actual problem, fraudulent papers are vastly

612 | NATURE | VOL 505 | 30 JANUARY 2014

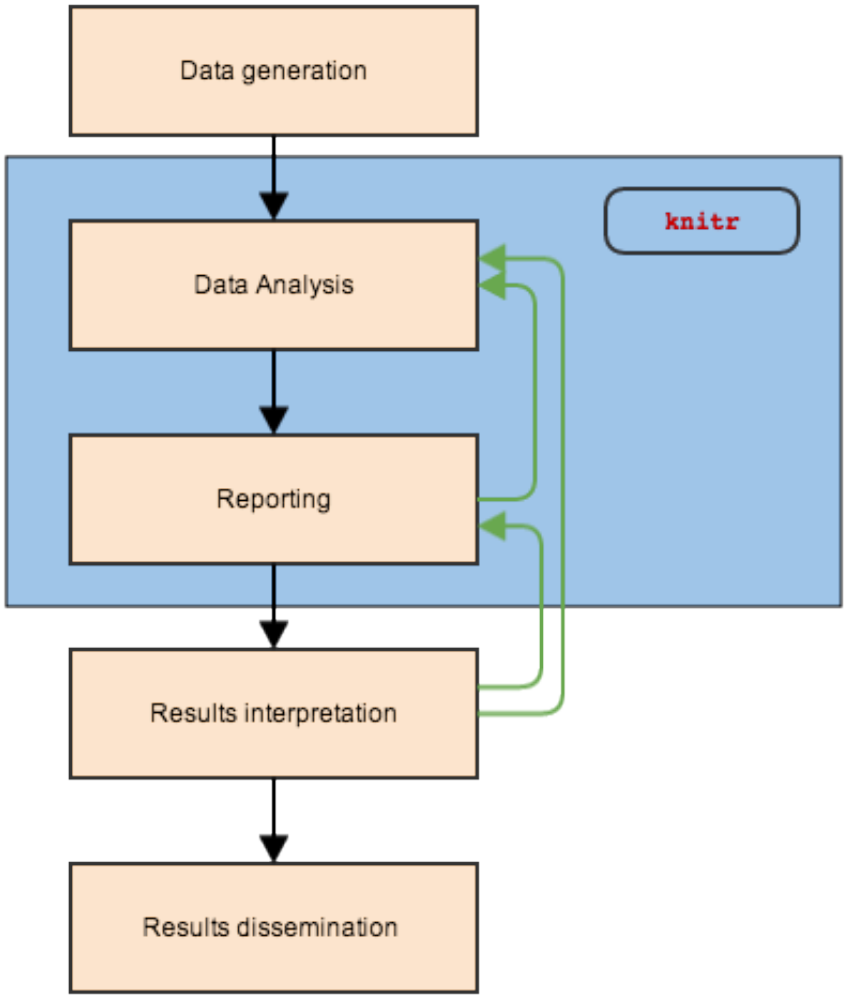
At the lab?

Trouble at the lab

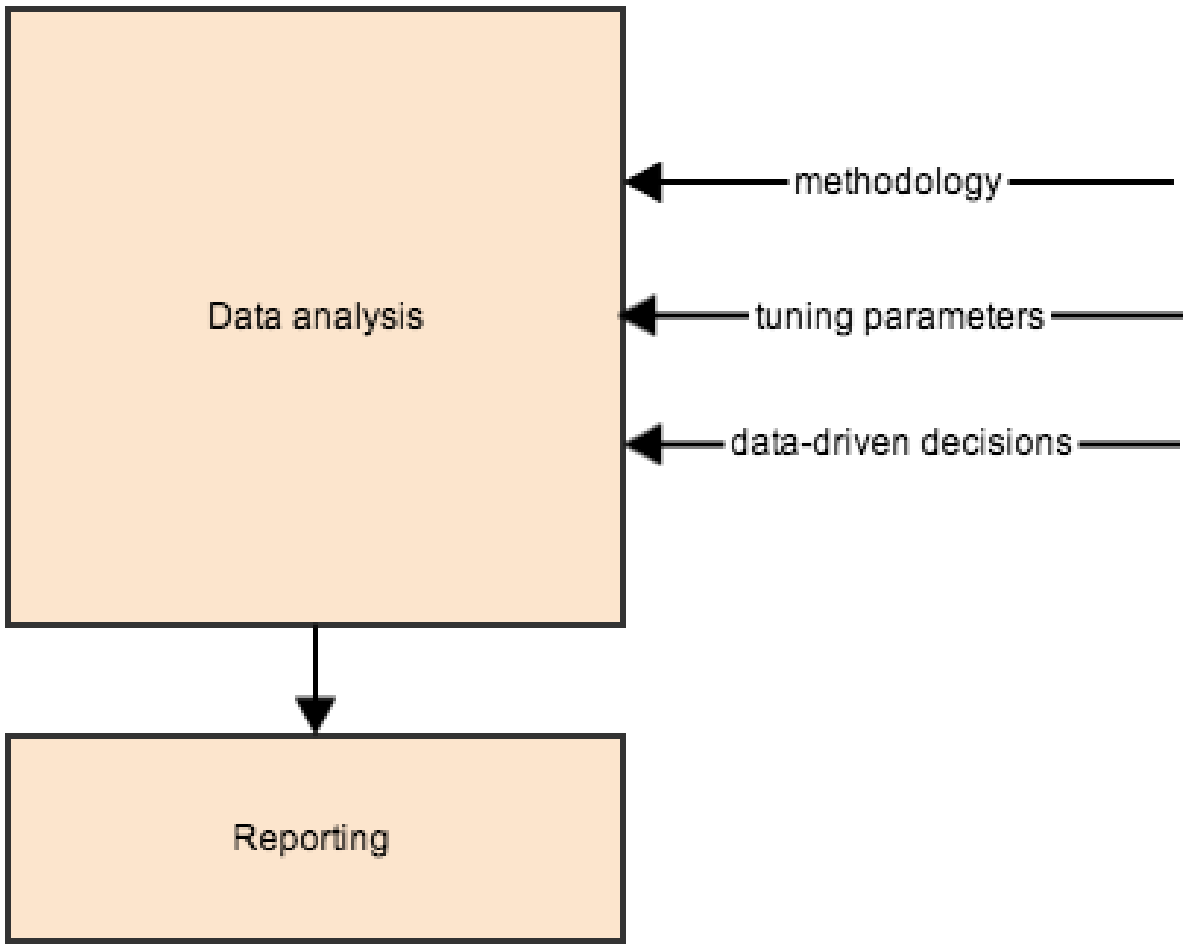
Scientists like to think of science as self-correcting. To an alarming degree, it is not

Oct 19th 2013 | From the print edition

Where do we fit in?



Data analysis



Goal: code + prose = report

Solution

- Cut and paste for report production is not a viable method
  - tedious
  - slow
  - error-prone
- Incorporate analysis code into text documents
  - `knit` to create results
  - results get incorporated into document
  - post-process to get any type of output format
- Literate documents
  - R Code creates results to inform report
  - Prose surrounding code informs analysis

How? Markdown.

```
This is a first-level heading
==============

Second level
------------

### Third

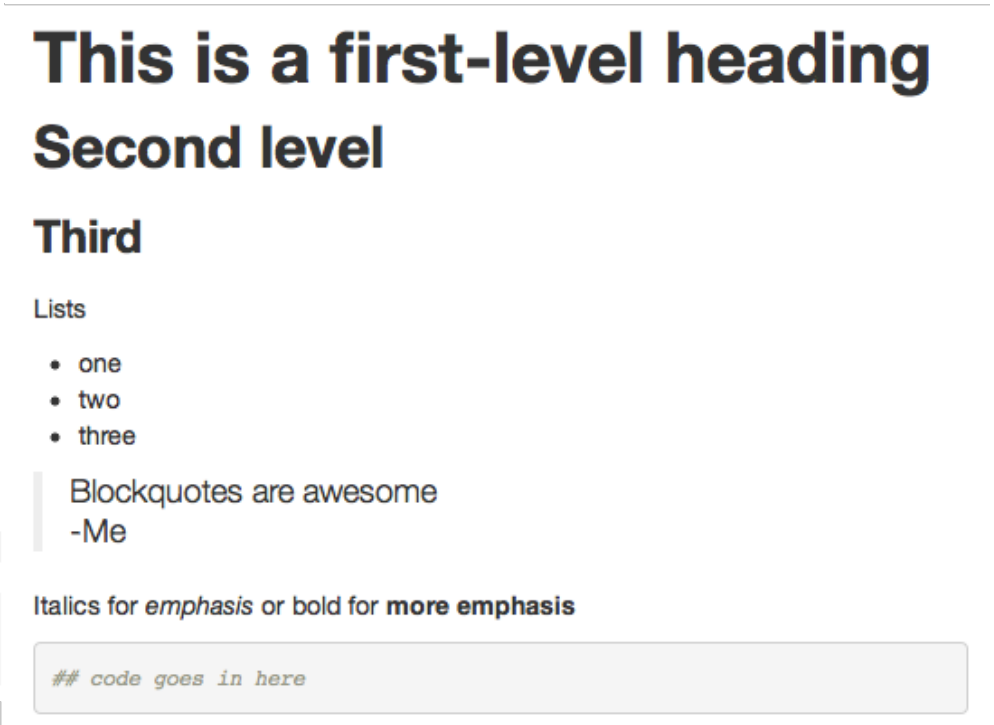
Lists

- one
- two
- three

> Blockquotes are awesome
> - Me

Italics for *emphasis* or bold for **more emphasis**

```{r my-first-chunk, results='asis'}
## code goes in here
```
```



This is a first-level heading

Second level

Third

Lists

- one
- two
- three

> Blockquotes are awesome
> -Me

Italics for *emphasis* or bold for **more emphasis**

```
## code goes in here
```

Contrast with html

```
<div id="this-is-a-first-level-heading" class="section level1">
<h1>This is a first-level heading</h1>
<div id="second-level" class="section level2">
<h2>Second level</h2>
<div id="third" class="section level3">
<h3>Third</h3>
<p>Lists</p>
<ul>
<li>one</li>
<li>two</li>
<li>three</li>
</ul>
<blockquote>
<p>Blockquotes are awesome</p>
<p>-Me</p>
</blockquote>
<p>Italics for <em>emphasis</em> or bold for <strong>more emphasis</strong></p>
<pre class="r"><code>## code goes in here</code></pre>
</div>
</div>
</div>
```

. . . and tex

```
\documentclass[]{article}
\begin{document}
\begin{center}
\normalsize
\end{center}
\section{This is a first-level heading}\label{this-is-a-first-level-heading}

\subsection{Second level}\label{second-level}

\subsubsection{Third}\label{third}

Lists

\begin{itemize}
\itemsep1pt\parskip0pt\parsep0pt
\item
 one
\item
 two
\item
 three
\end{itemize}

\begin{quote}
Blockquotes are awesome - Me
\end{quote}

Italics for \emph{emphasis} or bold for \textbf{more emphasis}

\begin{Shaded}
\begin{Highlighting}[]
\NormalTok{## code goes in here}
\end{Highlighting}
\end{Shaded}

\end{document}
```

Incorporating code chunks

Three backticks, each chunk needs a unique name:

````
```{r my-first-chunk}
## code goes in here and gets evaluated
t.test(mpg ~ vs, data = mtcars)
```
````

```
##
##  Welch Two Sample t-test
##
## data:  mpg by vs
## t = -4.6671, df = 22.716, p-value = 0.0001098
## alternative hypothesis: true difference in means is not
## 95 percent confidence interval:
##  -11.462508  -4.418445
## sample estimates:
## mean in group 0 mean in group 1
```

Inline code

Inline code uses single backticks

The mean mpg is `` `r round(mean(mtcars$mpg), 2)` ``.

The mean mpg is 20.09.

Try it

- Open Rstudio
- Click New > Rmarkdown
- Select output format

Examine the markdown prose and the code.

What do you expect the output to look like?

- Click `knit`
- What do you get?

How it works: knitr

Code chunks are evaluated sequentially the same, fresh R session:

Rmd

```
---
title: "Untitled"
author: "Michael Sachs"
date: "March 30, 2015"
output: html_document
---

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r}
summary(cars)
```

You can also embed plots, for example:

```{r, echo=FALSE}
plot(cars)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
```

md

```
---
title: "Untitled"
author: "Michael Sachs"
date: "March 30, 2015"
output: html_document
---

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:


```r
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

You can also embed plots, for example:

![plot of chunk unnamed-chunk-2](figure/unnamed-chunk-2-1.png)

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
```

How it works: pandoc

- Pandoc converts the markdown format to some other document type:
  - Word
  - Html
  - Pdf
  - ...
- Templates and output formats for specific uses:
  - Beamer slideshows
  - ioslides presentations
  - Tufte-style handouts

Note: pdf based formats require Latex. Ask your IT person to install it

Caveats

- Markdown is minimalistic
- Easy to write and read

rmarkdown + knitr is designed to *quickly* and *simply* generate analytic reports with minimal markup

- Not complex or precise enough for **complete** control over output
- How much precision do you need?

Markdown specs

- Paragraphs, `# headers`, `## subheader`, `etc`, `> blockquotes`
- Emphasis, `_italics_`, `*italics*`, `__bold__`, `**bold**`
- Images/links: `![name](pathtoimage)`, `[text](link)`
- Lists/ordered lists
- Code chunks
- Latex equations: `$\sum_{i=1}^nX_i/n$` $= \sum_{i=1}^{n} X_i/n$
- Tables
- Citations: `[@citekey]`, bibtex, endnote, others supported

Front matter

- Metadata
- Document types
- Other options
  - Default figure size, table of contents, theme
  - See `http://rmarkdown.rstudio.com/` for complete documentation

```
---
title: "Reproducible Analyses with knitr and rmarkdown"
author: "Michael Sachs"
date: "March 31, 2015"
output:
  ioslides_presentation:
    widescreen: true
---
```

Controlling R output

knitr chunk output

Results

Default: `results = 'markup'`

````
```{r markup, results = 'markup'}
head(mtcars, 4)
```
````

```
##                mpg cyl disp  hp drat    wt  qsec vs am
## Mazda RX4     21.0   6  160 110 3.90 2.620 16.46  0  1
## Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1
## Datsun 710    22.8   4  108  93 3.85 2.320 18.61  1  1
## Hornet 4 Drive 21.4  6  258 110 3.08 3.215 19.44  1  0
```

knitr chunk output

````
results = 'asis'


```{r asis, results = 'asis'}
head(mtcars, 4)
```
````

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear ca
```

Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4 Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4 Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1 Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1

knitr chunk output

Make tables pretty with `knitr::kable` and `results = 'asis'`

```
kable(head(mtcars, 4), digits = 1, caption = "Motor Trend C
```

Table 1:Motor Trend Cars, 1974

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am
Mazda RX4	21.0	6	160	110	3.9	2.6	16.5	0	1
Mazda RX4 Wag	21.0	6	160	110	3.9	2.9	17.0	0	1
Datsun 710	22.8	4	108	93	3.8	2.3	18.6	1	1
Hornet 4 Drive	21.4	6	258	110	3.1	3.2	19.4	1	0

Tables and other output

Several other packages are available to customize table output:

- `pander`: Good for printing output from regression models:

```
pander::pander(lm(mpg ~ factor(cyl), data = mtcars))
```

Table 2:Fitting linear model: mpg ~ factor(cyl)

	Estimate	Std. Error	t value	Pr(>|t|)
factor(cyl)6	-6.921	1.558	-4.441	0.0001195
factor(cyl)8	-11.56	1.299	-8.905	8.568e-10
(Intercept)	26.66	0.9718	27.44	2.688e-22

Other options

- `include = FALSE` evaluates code but doesn't include anything
- `echo = FALSE` don't display results
- `warning = FALSE` don't display warnings
- `cache = TRUE` cache results for long-running stuff
- `comment = NA` hide # from `markup` output

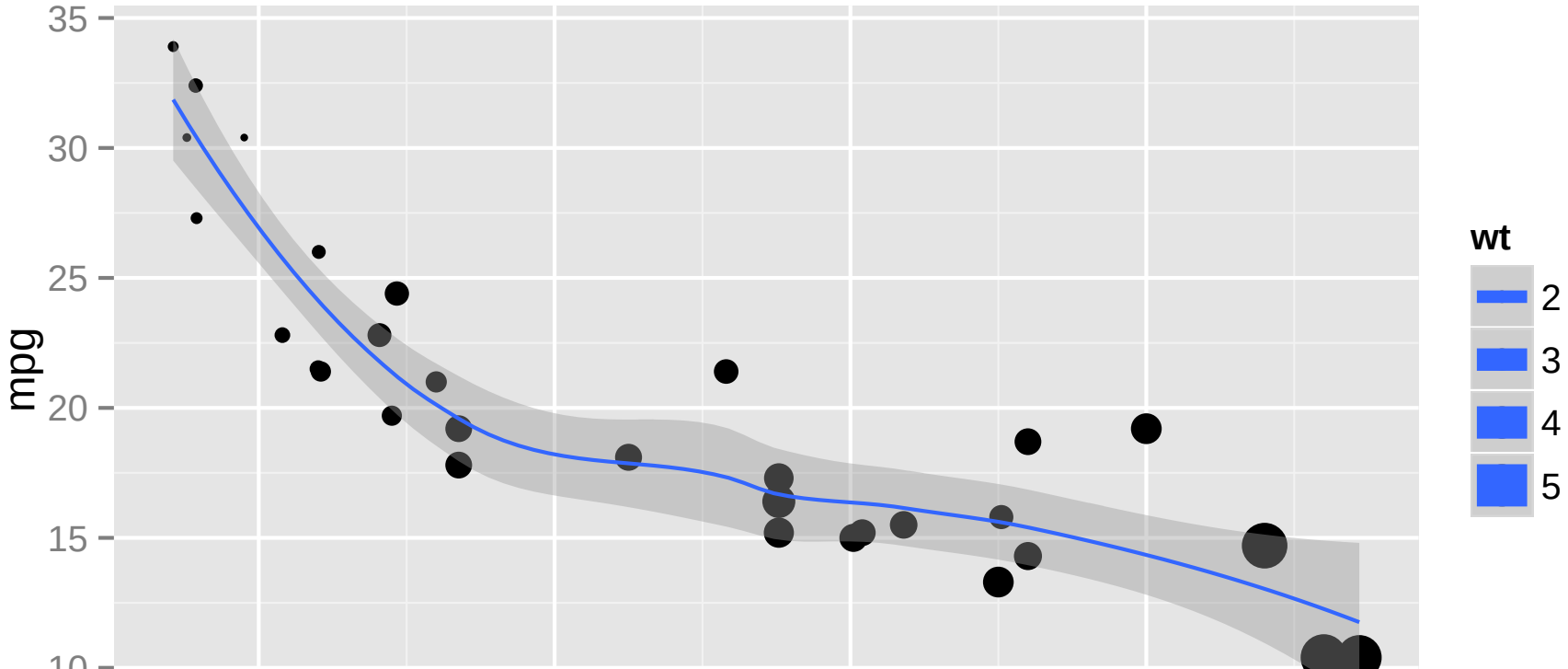
Figure options

The important ones:

- `fig.width`, `fig.height`, in inches. Can also be set globally in the header.
- `fig.align`, left, right or center
- `fig.cap = "Caption"` add caption to figure

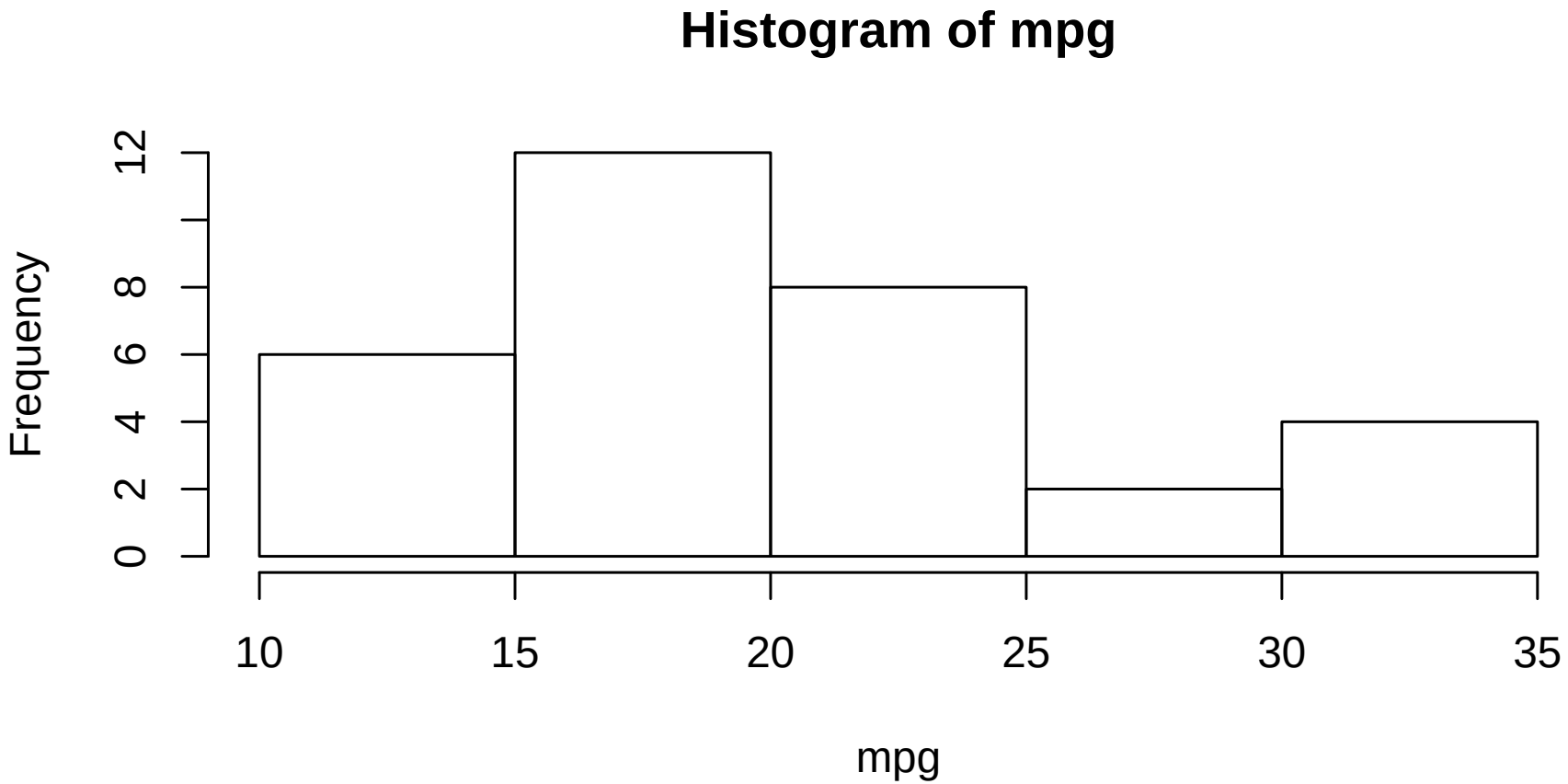
Example figure

```
```{r ggmt, fig.align = 'center', fig.height = 3.5, fig.wi
library(ggplot2)
ggplot(mtcars, aes(x = disp, y = mpg, size = wt)) +
  geom_point() + geom_smooth(method = "loess")
```
```



Example figure

```
```{r basemt, fig.align = 'center', fig.height = 3.5, fig.w
with(mtcars, hist(mpg))
```
```



Try it!

- Create a new rmarkdown document with an output format of your choice
- Use the `BMI.CSV` dataset to perform some basic analysis
- Display the data using `head`
- Summarize the data
- Do a t-test
- Make a figure using ggplot
- Make a table of regression coefficients
- *Write about what you are doing along the way*

Summary

- Analysis + report writing should be easy and integrated
- knitr + rmarkdown + pandoc all in Rstudio
- You don't have to remember everything, use the menus and help documents
- Your future self will thank you for having a reproducible analysis

Resources

Topic	Link
KBroman's UWisc Class	`https://kbroman.github.io/knitr_knu`
Knitr homepage	`http://yihui.name/knitr/`
rmarkdown documentation	`http://rmarkdown.rstudio.com/`
Another knitr tutorial	`http://sachsmc.github.io/knit-git-m`
Pandoc reference	`http://johnmacfarlane.net/pandoc/`