`oxforddown`: An Oxford University Thesis Template for R Markdown

Author Name

Your College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2018

Abstract

This *R Markdown* template is for writing an Oxford University thesis. The template is built using Yihui Xie's `bookdown` package, with heavy inspiration from Chester Ismay's `thesisdown` and the `OxThesis` LaTeX template (most recently adapted by John McManigle).

This template's sample content include illustrations of how to write a thesis in R Markdown, and largely follows the structure from this R Markdown workshop.

Congratulations for taking a step further into the lands of open, reproducible science by writing your thesis using a tool that allows you to transparently include tables and dynamically generated plots directly from the underlying data. Hip hooray!

`oxforddown`:
An Oxford University Thesis Template for R Markdown

Author Name

Your College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2018

For Yihui Xie

Acknowledgements

This is where you will normally thank your advisor, colleagues, family and friends, as well as funding and institutional support. In our case, we will give our praises to the people who developed the ideas and tools that allow us to push open science a little step forward by writing plain-text, transparent, and reproducible theses in R Markdown.

We must be grateful to John Gruber for inventing the original version of Markdown, to John MacFarlane for creating Pandoc (`http://pandoc.org`) which converts Markdown to a large number of output formats, and to Yihui Xie for creating `knitr` which introduced R Markdown as a way of embedding code in Markdown documents, and `bookdown` which added tools for technical and longer-form writing.

Special thanks to Chester Ismay, who created the `thesisdown` package that helped many a PhD student write their theses in R Markdown. And a very special tahnks to John McManigle, whose adaption of Sam Evans' adaptation of Keith Gillow's original maths template for writing an Oxford University DPhil thesis in LaTeX provided the template that I adapted for R Markdown.

Finally, profuse thanks to JJ Allaire, the founder and CEO of RStudio, and Hadley Wickham, the mastermind of the tidyverse without whom we'd all just given up and done data science in Python instead. Thanks for making data science easier, more accessible, and more fun for us all.

Ulrik Lyngs
Linacre College, Oxford
2 December 2018

Abstract

This *R Markdown* template is for writing an Oxford University thesis. The template is built using Yihui Xie's `bookdown` package, with heavy inspiration from Chester Ismay's `thesisdown` and the `OxThesis` LaTeX template (most recently adapted by John McManigle).

This template's sample content include illustrations of how to write a thesis in R Markdown, and largely follows the structure from this R Markdown workshop.

Congratulations for taking a step further into the lands of open, reproducible science by writing your thesis using a tool that allows you to transparently include tables and dynamically generated plots directly from the underlying data. Hip hooray!

Contents

Appendices

List of Figures

List of Tables

List of Abbreviations

1-D, 2-D . . . One- or two-dimensional, referring in this thesis to spatial dimensions in an image.

Otter . . . . . One of the finest of water mammals.

Hedgehog . . . Quite a nice prickly friend.

Introduction

Welcome to the *R Markdown* Oxford University thesis template. This sample content is adapted from `thesisdown` and the formatting of PDF output is adapted from the OxThesis LaTeX template. Hopefully, writing your thesis in R Markdown will provide a nicer interface to the OxThesis template if you haven't used TeX or LaTeX before. More importantly, using *R Markdown* allows you to embed chunks of code directly into your thesis and generate plots and tables directly from the underlying data, avoiding copy-paste steps. This will get you into the habit of doing reproducible research, which benefits you long-term as a researcher, but also will greatly help anyone that is trying to reproduce or build upon your results down the road.

Using LaTeX together with *Markdown* is more consistent than the output of a word processor, much less prone to corruption or crashing, and the resulting file is smaller than a Word file. While you may never have had problems using Word in the past, your thesis is likely going to be about twice as large and complex as anything you've written before, taxing Word's capabilities.

Why use it?

R Markdown creates a simple and straightforward way to interface with the beauty of LaTeX. Packages have been written in **R** to work directly with LaTeX to produce nicely formatting tables and paragraphs. In addition to creating a user friendly interface to LaTeX, *R Markdown* allows you to read in your data, analyze it and to visualize it using **R**, **Python** or other languages, and provide documentation and commentary on the results of your project.
Further, it allows for results of code output to be passed inline to the commentary of your results. You'll see more on this later, focusing on **R**. If you are more into

Python or something else, you can still use *R Markdown* - see ‘Other language engines’ in Yihui Xie’s *R Markdown: The Definitive Guide.*

Who should use it?

Anyone who needs to use data analysis, math, tables, a lot of figures, complex cross-references, or who just cares about reproducibility in research can benefit from using *R Markdown.* If you are working in ‘softer’ fields, the user-friendly nature of the *Markdown* syntax and its ability to keep track of and easily include figures, automatically generate a table of contents, index, references, table of figures, etc. should still make it of great benefit to your thesis project.

Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit…

There is no one who loves pain itself, who seeks after it and wants to have it, simply because it is pain…

— Cicero's *de Finibus Bonorum et Malorum.*

1

R Markdown Basics: The Markdown syntax

Contents

Here is a brief introduction to using *R Markdown*. *Markdown* is a simple formatting syntax for authoring HTML, PDF, and MS Word documents and much, much more. *R Markdown* provides the flexibility of *Markdown* with the implementation of **R** input and output. For more details on using *R Markdown* see `http://rmarkdown.rstudio.com`.

Be careful with your spacing in *Markdown* documents. While whitespace largely is ignored, it does at times give *Markdown* signals as to how to proceed. As a habit, try to keep everything left aligned whenever possible, especially as you type a new paragraph. In other words, there is no need to indent basic text in the Rmd document (in fact, it might cause your text to do funny things if you do).

1.1 Markdown basic syntax

1.1.1 Italics and bold

- *Italics* are done like *this* or _this_
- **Bold** is done like **this** or __this__
- ***Bold and italics*** is done like ***this***, ___this___, or (the most transparent solution, in my opinion) **_this_**

1.1.2 Inline code

- `Inline code` is created with backticks like `` `this` ``

1.1.3 Sub and superscript

Sub$_2$ and super^2 script is created like this~2~ and this^2^

1.1.4 Strikethrough

- ~~Strikethrough~~ is done ~~like this~~

1.1.5 'Escaping' (aka "What if I need an actual asterisk?")

- To include an actual *, _ or \, add another \ in front of them: *, _, \\

1.1.6 Endash (–), emdash (—)

- – and — with -- and ---

1.1.7 Blockquotes

Do like this:

> Put a > in front of the line.

1.1.8 Headings

- are done with #'s of increasing number, i.e.
    - # First-level heading
    - ## Second-level heading
    - ### Etc.

In PDF output, a level-five heading will turn into a paragraph heading, i.e. `\paragraph{My level-five heading}`, which appears as bold text on the same line as the subsequent paragraph.

1.1.9 Lists

Unordered list by starting a line with an * or a -:

- Item 1
- Item 2

Ordered lists by starting a line with a number:

1. Item 1
2. Item 2

Notice that you can mislabel the numbers and *Markdown* will still make the order right in the output.

To create a sublist, indent the values a bit (at least four spaces or a tab):

1. Item 1
2. Item 2

3. Item 3

    - Item 3a
    - Item 3b

1.1.10 Line breaks

The official *Markdown* way to create line breaks is by ending a line with more than two spaces.

Roses are red. Violets are blue.

This appears on the same line in the output, because we didn't add spaces after red.

Roses are red.
Violets are blue.

This appears with a line break because I added spaces after red.

I find this is confusing, so I recommend the alternative way: Ending a line with a backslash will also create a linebreak:

Roses are red.
Violets are blue.

To create a new paragraph, you put a blank line.

Therefore, this line starts its own paragraph.

1.1.11 Hyperlinks

- This is a hyperlink created by writing the text you want turned into a clickable link in `[square brackets followed by a](https://hyperlink-in-parentheses)`

1.1.12 Footnotes

- Are created[1] by writing either ^[my footnote text] for supplying the footnote content inline, or something like `[^a-random-footnote-label]` and

[1] my footnote text

supplying the text elsewhere in the format shown below [2]:

```
[^a-random-footnote-label]: This is a random test.
```

1.1.13 Comments

To write comments within your text that won't actually be included in the output, use the same syntax as for writing comments in HTML. That is, <!-- this will not be included in the output -->.

1.1.14 Math

The syntax for writing math is stolen from LaTeX. To write a math expression that will be shown **inline**, enclose it in dollar signs. - This: $A = \pi*r^{2}$ Becomes: $A = \pi * r^2$

To write a math expression that will be shown in a block, enclose it in two dollar signs.
This: $$A = \pi*r^{2}$$

Becomes:

$$A = \pi * r^2$$

To create numbered equations, put them in an 'equation' environment and give them a label with the syntax `(\#eq:label)`, like this:

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

Becomes:

$$f\left(k\right) = \binom{n}{k} p^k\left(1-p\right)^{n-k} \tag{1.1}$$

For more (e.g. how to theorems), see e.g. the documentation on bookdown.org

[2] This is a random test.

1.2 Additional resources

- *R Markdown: The Definitive Guide* - `https://bookdown.org/yihui/rmarkdown/`
- *R for Data Science* - `https://r4ds.had.co.nz`

2
Adding code

Contents

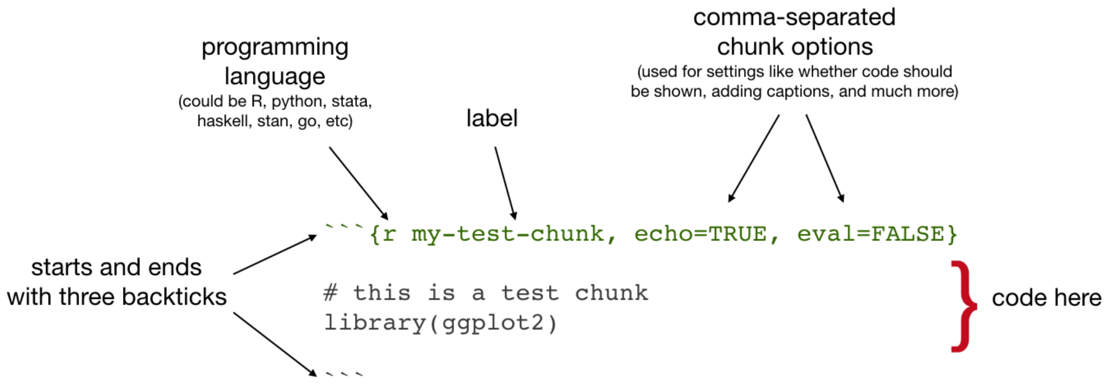
The magic of R Markdown is that we can add code within our document to make it dynamic.

We do this either as *code chunks* (generally used for loading libraries and data, performing calculations, and adding images, plots, and tables), or *inline code* (generally used for dynamically reporting results within our text).

2.1 Code chunks

The syntax of a code chunk is shown in Figure 2.1.

Common chunk options include (see e.g. bookdown.org):



~~~
```{r my-test-chunk, echo=TRUE, eval=FALSE}
# this is a test chunk
library(ggplot2)
```
~~~

Figure 2.1: Code chunk syntax

- `echo`: whether or not to display code in knitted output
- `eval`: whether or to to run the code in the chunk when knitting
- `include`: wheter to include anything from the from a code chunk in the output document
- `fig.cap`: figure caption
- `fig.scap`: short figure caption, which will be used in the 'List of Figures' in the PDF front matter

IMPORTANT: Do *not* use underscoores in your chunk labels - if you do, you are likely to get an error in PDF output saying something like "! Package caption Error: \caption outside float".

2.1.1 Setup chunks

An R Markdown document usually begins with a chunk that is used to **load libraries**, and to **set default chunk options** with `knitr::opts_chunk$set`.

In your thesis, this will probably happen in **index.Rmd** and/or as opening chunks in each of your chapters.

```
‘‘‘{r setup, include=FALSE}
# don’t show code unless we explicitly set echo = TRUE
knitr::opts_chunk$set(echo = FALSE)
```

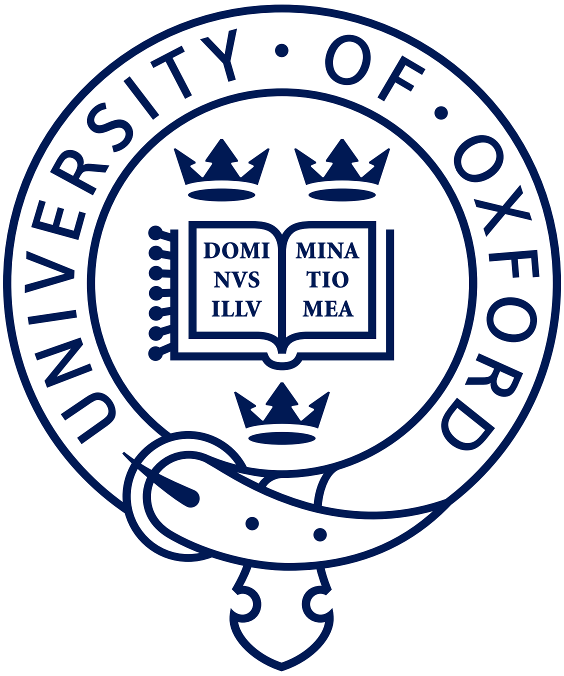


Figure 2.2: Oxford logo

```
library(tidyverse)
‘‘‘
```

2.1.2 Including images

Code chunks are also used for including images, with `include_graphics` from the `knitr` package, as in Figure 2.2

```
knitr::include_graphics("figures/beltcrest.png")
```

Useful chunk options for figures include:

- `out.width` (use with a percentage) for setting the image size
- if you've got an image that gets waaay to big in your output, it will be constrained to the page width by setting `out.width = "100%"`

Figure rotation

You can use the chunk option `out.extra` to rotate images.

The syntax is different for LaTeX and HTML, so for ease we might start by assigning the right string to a variable that depends on the format you're outputting to:

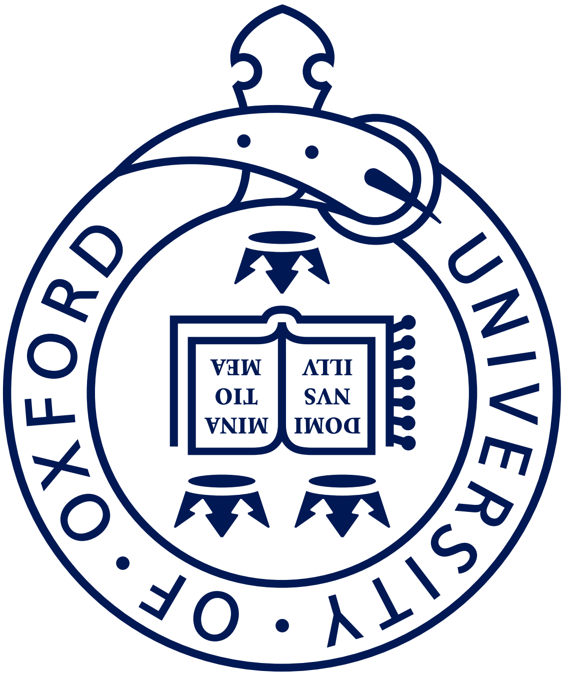


Figure 2.3: Oxford logo, rotated

```
if (knitr::opts_knit$get(’rmarkdown.pandoc.to’) == ’latex’){
  rotate180 <- "angle=180"
} else {
  rotate180 <- "style=’transform:rotate(180deg);’"
}
```

Then you can reference that variable as the value of `out.extra` to rotate images, as in Figure 2.3.

2.1.3 Including plots

Similarly, code chunks are used for including dynamically generated plots. You use ordinary code in R or other languages - Figure 2.4 shows a plot of the `cars` dataset of stopping distances for cars at various speeds (this dataset is built in to **R**).

```
cars %>%
  ggplot() +
    aes(x = speed, y = dist) +
    geom_point()
```

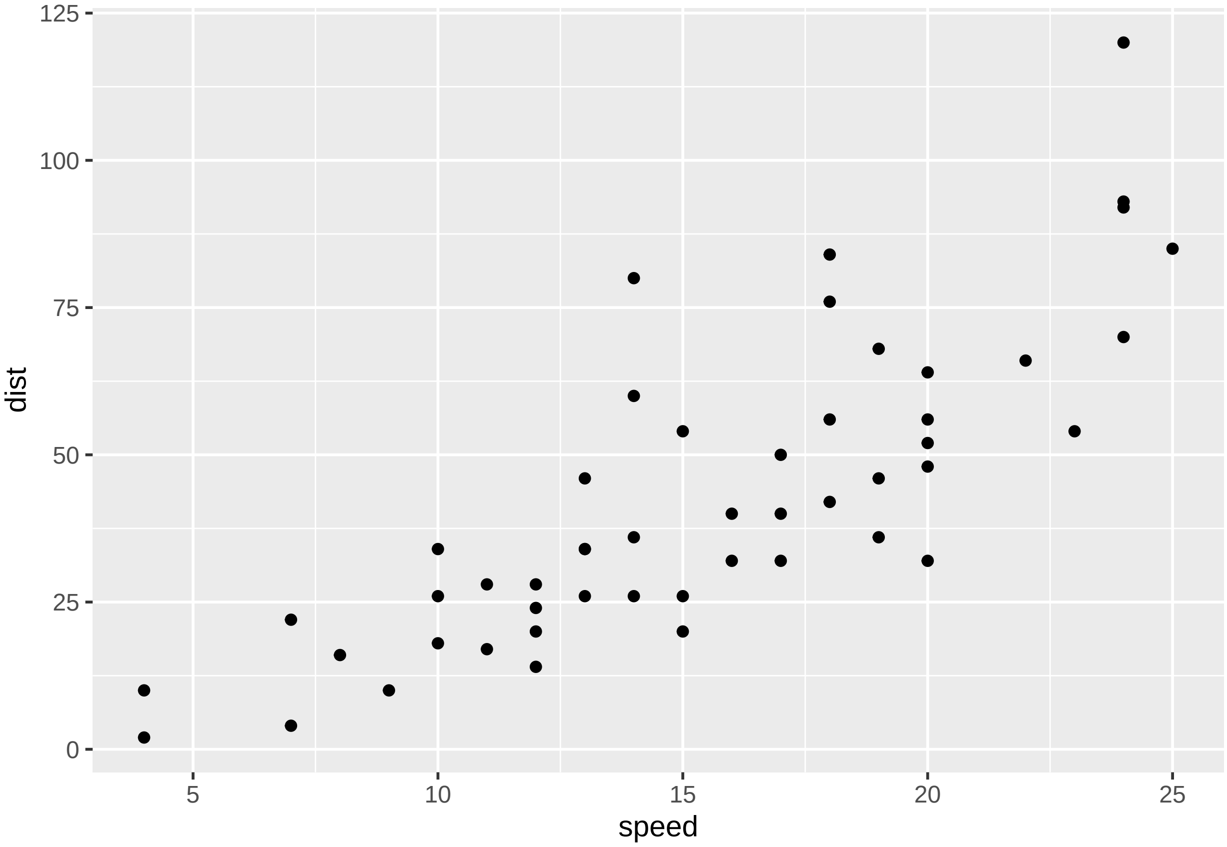


Figure 2.4: A ggplot of car stuff

Under the hood, plots are included in your document in the same way as images - when you build the book or knit a chapter, the plot is automatically generated from your code, saved as an image, then included into the output document.

2.1.4 Including tables

Tables are usually included with the `kable` function from the `knitr` package.

Table 2.1 shows the first rows of that cars data - read in your own data, then use this approach to automatically generate tables.

```
cars %>%
  head() %>%
  knitr::kable(caption = "A knitr kable table")
```

- Gotcha: when using `kable`, captions are set inside the `kable` function
- The `kable` package is often used with the `kableExtra` package

Table 2.1: A knitr kable table

speed	dist
4	2
4	10
7	4
7	22
8	16
9	10

2.1.5 A note on content positioning

One thing that may be annoying is the way *R Markdown* handles "floats" like tables and figures.

In your PDF output, LaTeX will try to find the best place to put your object based on the text around it and until you're really, truly done writing you should just leave it where it lies.

When the time comes for you to make final tweaks to content positioning, read the relevant R Markdown documentation to see if there are easy ways to do what you want.

If you have very specific needs, you might have to read up on LaTeX (`https://en.wikibooks.org/wiki/LaTeX/Floats,_Figures_and_Captions`) for your PDF output and/or on how to style HTML documents with CSS for your gitbook output.

2.2 Inline code

'Inline code' simply means inclusion of code inside text.

The syntax for doing this is `` `r R_CODE` ``

For example, `` `r 4 + 4` `` would output 8 in your text.

You will usually use this in parts of your thesis where you report results - read in data or results in a code chunk, store things you want to report in a variable, then insert the value of that variable in your text.

For example, we might assign the number of rows in the `cars` dataset to a variable:

```
num_car_observations <- nrow(cars)
```

We might then write:

“In the `cars` dataset, we have `` `r num_car_observations` `` observations.”

Which would output:

“In the `cars` dataset, we have 50 observations.”

2.2.1 Referring to results computed in other languages than R

I’ve commented the below section out, to avoid compilation errors from the `reticulate` package being unable to find a python installation (after I installed MacOS Catalina, `reticulate` was unable to select a python version on my system, and I had to set it manually with `use_python`).

If you need to use other langauges, have a look at the content I commented out by the end of the **02-rmd-basics-code.Rmd** file, which gives an example of using Python in your R Markdown file.

3
Citations and cross-references

Contents

3.1 Citations

The usual way to include citations in an *R Markdown* document is to put references in a plain text file with the extension **.bib**, in **BibTex** format.[1] Then reference the path to this file in **index.Rmd**'s YAML header with `bibliography: example.bib`.

[1] The bibliography can be in other formats as well, including EndNote (**.enl**) and RIS (**.ris**), see rmarkdown.rstudio.com/authoring_bibliographies_and_citations.

Most reference managers can create a .bib file with you references automatically. However, the **by far** best reference manager to use with *R Markdown* is Zotero with the Better BibTex plug-in, because the citr plugin for RStudio (see below) can read references directly from your Zotero library!

Here is an example of an entry in a **.bib** file:

```
@article{Shea2014,
  author =        {Shea, Nicholas and Boldt, Annika},
  journal =       {Trends in Cognitive Sciences},
  pages =         {186--193},
  title =         {{Supra-personal cognitive control}},
  volume =        {18},
  year =          {2014},
  doi =           {10.1016/j.tics.2014.01.006},
}
```

In this entry highlighed section, 'Shea2014' is the **citation identifier**. To default way to cite an entry in your text is with this syntax: [@citation-identifier].

So I might cite some things (Shea et al. 2014; Lottridge et al. 2012).

3.1.1 PDF output

In PDF output, the bibliography is handled by the OxThesis LaTeX template. If you set bib-humanities: true in **index.Rmd**, then in-text references will be formatted as author-year; otherwise references will be shown as numbers.

If you choose author-year formatting, a number of variations on the citation syntax are useful to know:

- Put author names outside the parenthesis
  - This: @Shea2014 says blah.
  - Becomes: Shea et al. (2014) says blah.
- Include only the citation-year (in parenthesis)

  - This: `Shea et al. says blah [-@Shea2014]`
  - Becomes: Shea et al. says blah (2014)

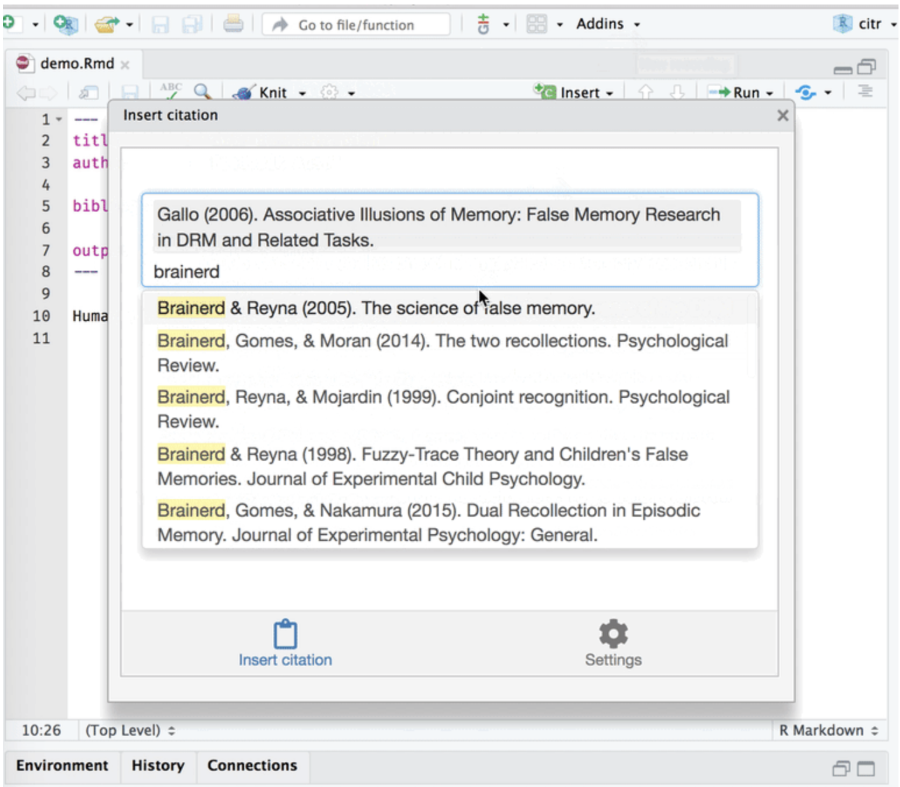
- Add text and page or chapter references to the citation

  - This: `[see @Shea2014, pp. 33-35; also @Wu2016, ch. 1]`
  - Becomes: Blah blah (see Shea et al. 2014, pp. 33-35; also Wu 2016, ch. 1).

3.1.2 Gitbook output

In gitbook output, citations are by default inserted in the Chicago author-date format.

To change the format, add `csl: some-other-style.csl` in **index.Rmd**'s YAML header. You can browse through and download styles at zotero.org/styles.



Insert citation

Gallo (2006). Associative Illusions of Memory: False Memory Research in DRM and Related Tasks.

brainerd

Brainerd & Reyna (2005). The science of false memory.

Brainerd, Gomes, & Moran (2014). The two recollections. Psychological Review.

Brainerd, Reyna, & Mojardin (1999). Conjoint recognition. Psychological Review.

Brainerd & Reyna (1998). Fuzzy-Trace Theory and Children's False Memories. Journal of Experimental Child Psychology.

Brainerd, Gomes, & Nakamura (2015). Dual Recollection in Episodic Memory. Journal of Experimental Psychology: General.

Figure 3.1: The ‘citr‘ add-in

3.1.3 Insert references easily with the citr add-in

For an easy way to insert citations, try the `citr` RStudio add-in (Figure 3.1). You can install this add-in by typing `install.packages("citr")` in the R Console.

3.2 Cross-referencing

We can make cross-references to **sections** within our document, as well as to **figures** (images and plots) and **tables**.

The general cross-referencing syntax is `\@ref(label)`

3.2.1 Section references

Headers are automatically assigned a reference label, which is the text in lower caps separated by dashes. For example, `# My header` is automatically given the label `my-header`. So `# My header` can be referenced with `\@ref(my-section)`

Remember what we wrote in section 3.1?

We can also use **hyperlink syntax** and add # before the label, though this is only guaranteed to work properly in HTML output:

- So if we write `Remember what we wrote up in [the previous section](#citations)?`
- It becomes Remember what we wrote up in the previous section?

Creating custom labels

It is a very good idea to create **custom labels** for our sections. This is because the automatically assigned labels will change when we change the titles of the sections - to avoid this, we can create the labels ourselves and leave them untouched if we change the section titles.

We create custom labels by adding `{#label}` after a header, e.g. `# My section {#my-label}`. See our chapter title for an example. That was section 3.

3.2.2 Figure (image and plot) references

- To refer to figures (i.e. images and plots) use the syntax `\@ref(fig:label)`
- **GOTCHA**: Figures and tables must have captions if you wish to cross-reference them.

Let's add an image:

```r
knitr::include_graphics("figures/captain.jpeg")
```

We refer to this image with `\@ref(fig:captain)`. So Figure 3.2 is this image.

And in Figure 2.4 we saw a cars plot.

3.2.3 Table references

- To refer to tables use the syntax `\@ref(tab:label)`

Let's include a table:

Figure 3.2: A marvel-lous meme

Table 3.1: Stopping cars

speed	dist
4	2
4	10
7	4
7	22
8	16

```r
knitr::kable(cars[1:5,],
             caption="Stopping cars")
```

We refer to this table with `\@ref(tab:cars-table2)`. So Table 3.1 is this table.

And in Table 2.1 we saw more or less the same cars table.

3.2.4 Including page numbers

Finally, in the PDF output we might also want to include the page number of a reference, so that it's easy to find in physical printed output. LaTeX has a command for this, which looks like this: `\pageref{fig/tab:label}` (note: curly braces, not parentheses)

When we output to PDF, we can use raw LaTeX directly in our .Rmd files. So if we wanted to include the page of the cars plot we could write:

- This: `Figure \@ref(fig:cars-plot) on page \pageref(fig:cars-plot)`
- Becomes: Figure 2.4 on page 13

Include page numbers only in PDF output

A problem here is that LaTeX commands don't display in HTML output, so in the gitbook output we'd see simply "Figure 2.4 on page".

One way to get around this is to use inline R code to insert the text, and use an `ifelse` statement to check the output format and then insert the appropriate text.

- So this: `` `r ifelse(knitr::is_latex_output(), "Figure \\@ref(fig:cars-plot) on page \\pageref{fig:cars-plot}", "")` ``
- Inserts this (check this on both PDF and gitbook): Figure 2.4 on page 13

Note that we need to escape the backslash with another backslash here to get the correct output.

3.3 Customising your thesis' front matter 'n stuff

3.3.1 Shorten captions shown in the list of figures (PDF)

You might want your list of figures (which follows the table of contents) to have shorter (or just different) figure descriptions than the actual figure captions.

Do this using the chunk option `fig.scap` ('short caption'), for example `{r captain-image, fig.cap="A very long and descriptive (and potentially boring) caption that doesn't fit in the list of figures, but helps the reader understand what the figure communicates.", fig.scap="A concise description for the list of figures"`

3.3.2 Shorten captions shown in the list of tables (PDF)

You might want your list of tables (which follows the list of figures in your thesis front matter) to have shorter (or just different) table descriptions than the actual table captions.

If you are using `knitr::kable` to generate a table, you can do this with the argument `caption.short`, e.g.:

```
knitr::kable(mtcars,
              caption = "A very long and descriptive (and potentially
              boring) caption that doesn't fit in the list of figures,
              but helps the reader understand what the figure
              communicates.",
              caption.short = "A concise description for the list of tables")
```

3.3.3 Shorting the running header (PDF)

You might want a chapter's running header (i.e. the header showing the title of the current chapter at the top of page) to be shorter (or just different) to the actual chapter title.

Do this by adding the latex command `\chaptermark{My shorter version}` after your chapter title.

For example, this chapter's running header is simply 'Cites and cross-refs', because it begins like this:

```
# Citations and cross-references {#cites-and-refs}
\chaptermark{Cites and cross-refs}
```

There is grandeur in this view of life, with its several powers, having been originally breathed into a few forms or into one; and that, whilst this planet has gone cycling on according to the fixed law of gravity, from so simple a beginning endless forms most beautiful and most wonderful have been, and are being, evolved.

— Charles Darwin (Darwin 1859)

4

Final Notes on The OxThesis template and on collaboration

Contents

4.1 Beginning chapters with quotes

The OxThesis LaTeX template lets you inject some wittiness into your thesis by including a block of type `savequote` at the beginning of chapters. To do this, use the syntax ` ```{block type='savequote'} `.[1]

Add the reference for the quote with the chunk option `quote_author="my author name"`. You will also want to add the chunk option `include=knitr::is_latex_output()` so that quotes are only included in PDF output.

[1] For more on custom block types, see the relevant section in *Authoring Books with R Markdown*.

It's not possible to use markdown syntax inside chunk options, so if you want to e.g. italicise a book name in the reference use a 'text reference': Create a named piece of text with '(ref:label-name) My text', then point to this in the chunk option with `quote_author='(ref:label-name)'`.

4.2 Highlighting corrections

For when it comes time to do corrections, you may want to highlight changes made when you submit a post-viva, corrected copy to your examiners so they can quickly verify you've completed the task. You can do so like this:

4.2.1 Short, inline corrections

Highlight **short, inline corrections** by doing `[like this]{.correction}` — the text between the square brackets will then be highlighted in blue in the output.

4.2.2 Blocks of added or changed material

Highlight entire **blocks of added or changed material** by putting them in a block of type `correction`, using the syntax ` ```{block type='correction'}`.[2] Like so:

For larger chunks, like this paragraph or indeed entire figures, you can use the `correction` block type. This environment **highlights paragraph-sized and larger blocks** with the same blue colour.

4.2.3 Stopping corrections from being highlighted in the output

For **PDF** output, go to **index.Rmd** and (i) set `corrections: false` under `params` in the YAML header (stops block of corrections from being highlighted), (ii) comment out `pandoc_args: ["--lua-filter=scripts_and_filters/correction_filter.lua"]` (stops inline corrections from being highlighted).

[2] In the **.tex** file for PDF output, this will put the content between `\begin{correction}` and `\end{correction}`; in gitbook output it will be put between `<div class="correction">` and `</div>`.

For **gitbook** output, go to **style.css** and comment out the styling for `.correction`.

4.3 Diving in to the OxThesis LaTeX template

For LaTeX minded people, you can read through **templates/template.tex** to see which additional customisation options are available as well as **templates/ociamthesis.cls** which supplies the base class. For example, **template.tex** provides an option for master's degree submissions, which changes identifying information to candidate number and includes a word count. At the time of writing, you must set this directly in **template.tex** rather than from the YAML header in **index.Rmd**.

4.4 Collaborative writing

Best practices for collaboration and change tracking when using R Markdown are still an open question. In the blog post **One year to dissertate** by Lucy D'Agostino, which I highly recommend, the author notes that she knits .Rmd files to a `word_document`, then uses the `googledrive` R package to send this to Google Drive for comments / revisions from co-authors, then incorporates Google Drive suggestions *by hand* into the .Rmd source files. This is a bit clunky, and there are ongoing discussions among the *R Markdown* developers about what the best way is to handle collaborative writing (see issue #1463 on GitHub, where CriticMarkup is among the suggestions).

For now, this is an open question in the community of R Markdown users. I often knit to a format that can easily be imported to Google Docs for comments, then go over suggested revisions and manually incorporate them back in to the .Rmd source files. For articles, I sometimes upload a near-final draft to Overleaf, then collaboratively make final edits to the LaTeX file there. I suspect some great solution will be developed in the not-to-distant future, probably by the RStudio team.

5 Customisations and extensions

This chapter describes a number of possible customizations to the `oxforddown` thesis.

5.1 Embedding PDF documents as chapters

You may want to embed existing PDF documents into the thesis, for example if your department allows a ‘portfolio’ style thesis and you need to include an existing typeset publication as a chapter.

In gitbook output, you can simply use `knitr::include_graphics` and it should include a scrollable (and downloadable) PDF. You will probably want to set the chunk options `out.width='100%'` and `out.height='1000px'`:

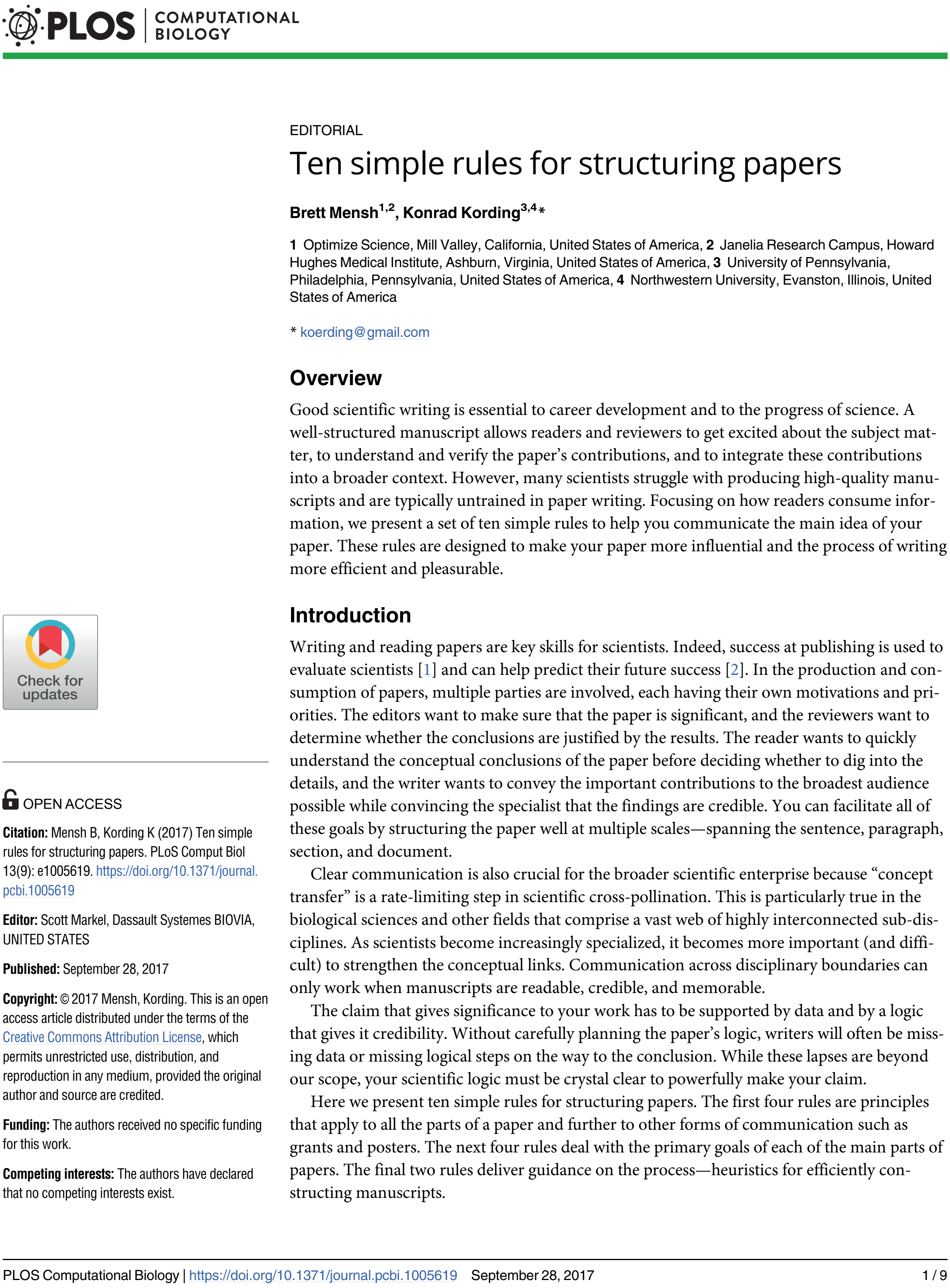
```
knitr::include_graphics("pdf_example/Mensh_Kording_2017.pdf")
```

In LaTeX output, however, this approach can cause odd behaviour. Therefore, when you build your thesis to PDF, split the pdf into pages (you can do this with the package `pdftools`) and then insert the appropriate LaTeX command, as shown below. *Note that the chunk option* `results='asis'` *must be used.*

```
# install.packages(pdftools)
# #split PDF in pages stored in pdf_example/split/
# pdftools::pdf_split("pdf_example/Mensh_Kording_2017.pdf",
# output = "pdf_example/split/")

pages <- list.files("pdf_example/split", full.names = TRUE)

# for each page, insert the latex command to insert it nicely
cat(paste0("\\newpage \\begin{center}
\\makebox[\\linewidth][c]{\\includegraphics[width=1.2\\linewidth]{",
pages, "}} \\end{center}"))
```



PLOS | COMPUTATIONAL BIOLOGY

EDITORIAL

Ten simple rules for structuring papers

Brett Mensh[1,2], **Konrad Kording**[3,4]*

1 Optimize Science, Mill Valley, California, United States of America, 2 Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, Virginia, United States of America, 3 University of Pennsylvania, Philadelphia, Pennsylvania, United States of America, 4 Northwestern University, Evanston, Illinois, United States of America

* koerding@gmail.com

Overview

Good scientific writing is essential to career development and to the progress of science. A well-structured manuscript allows readers and reviewers to get excited about the subject matter, to understand and verify the paper's contributions, and to integrate these contributions into a broader context. However, many scientists struggle with producing high-quality manuscripts and are typically untrained in paper writing. Focusing on how readers consume information, we present a set of ten simple rules to help you communicate the main idea of your paper. These rules are designed to make your paper more influential and the process of writing more efficient and pleasurable.

Introduction

Writing and reading papers are key skills for scientists. Indeed, success at publishing is used to evaluate scientists [1] and can help predict their future success [2]. In the production and consumption of papers, multiple parties are involved, each having their own motivations and priorities. The editors want to make sure that the paper is significant, and the reviewers want to determine whether the conclusions are justified by the results. The reader wants to quickly understand the conceptual conclusions of the paper before deciding whether to dig into the details, and the writer wants to convey the important contributions to the broadest audience possible while convincing the specialist that the findings are credible. You can facilitate all of these goals by structuring the paper well at multiple scales—spanning the sentence, paragraph, section, and document.

Clear communication is also crucial for the broader scientific enterprise because "concept transfer" is a rate-limiting step in scientific cross-pollination. This is particularly true in the biological sciences and other fields that comprise a vast web of highly interconnected sub-disciplines. As scientists become increasingly specialized, it becomes more important (and difficult) to strengthen the conceptual links. Communication across disciplinary boundaries can only work when manuscripts are readable, credible, and memorable.

The claim that gives significance to your work has to be supported by data and by a logic that gives it credibility. Without carefully planning the paper's logic, writers will often be missing data or missing logical steps on the way to the conclusion. While these lapses are beyond our scope, your scientific logic must be crystal clear to powerfully make your claim.

Here we present ten simple rules for structuring papers. The first four rules are principles that apply to all the parts of a paper and further to other forms of communication such as grants and posters. The next four rules deal with the primary goals of each of the main parts of papers. The final two rules deliver guidance on the process—heuristics for efficiently constructing manuscripts.

Check for updates

OPEN ACCESS

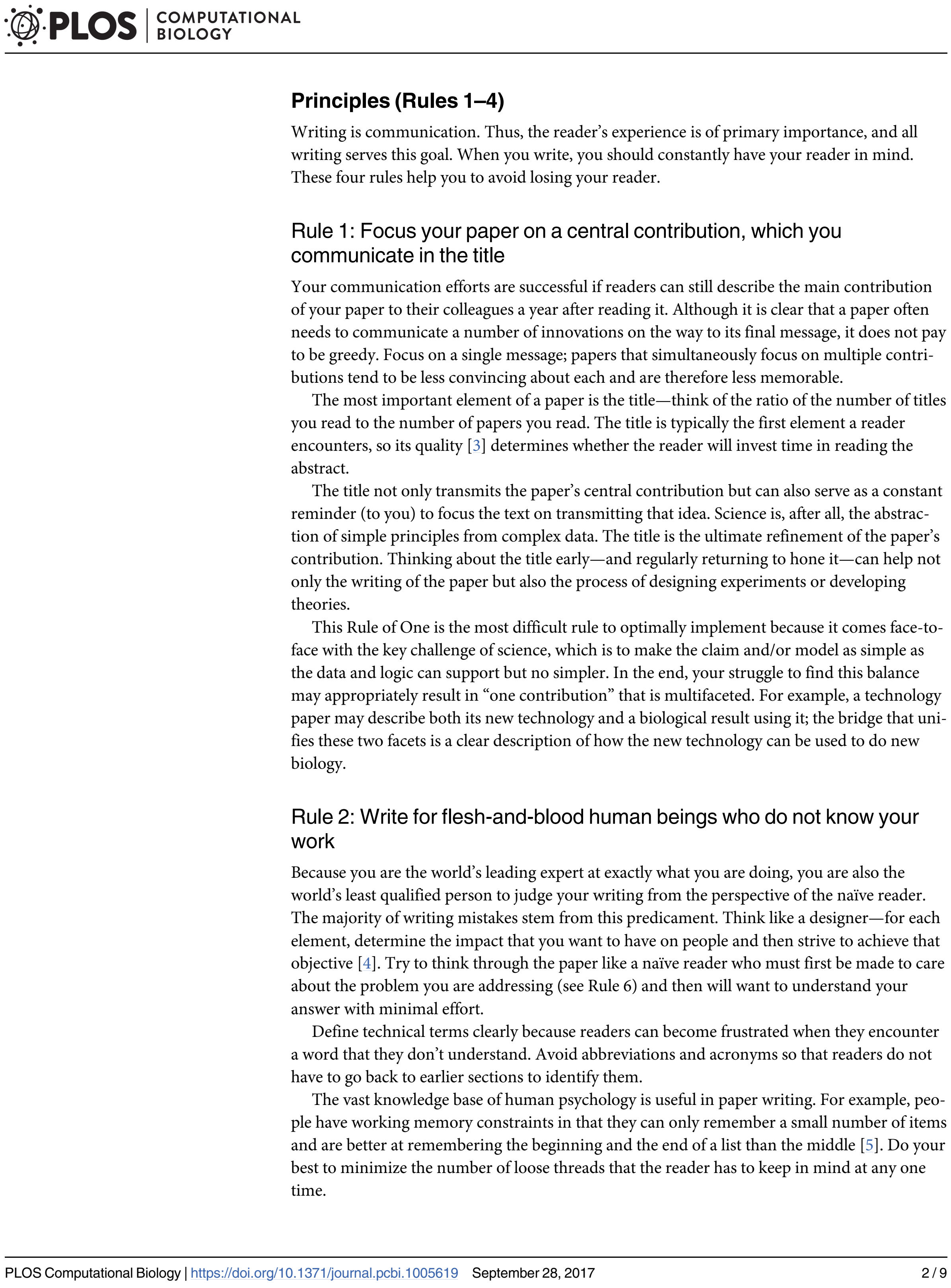
Citation: Mensh B, Kording K (2017) Ten simple rules for structuring papers. PLoS Comput Biol 13(9): e1005619. https://doi.org/10.1371/journal.pcbi.1005619

Editor: Scott Markel, Dassault Systemes BIOVIA, UNITED STATES

Published: September 28, 2017

Funding: The authors received no specific funding for this work.

Competing interests: The authors have declared that no competing interests exist.



Principles (Rules 1–4)

Writing is communication. Thus, the reader's experience is of primary importance, and all writing serves this goal. When you write, you should constantly have your reader in mind. These four rules help you to avoid losing your reader.

Rule 1: Focus your paper on a central contribution, which you communicate in the title

Your communication efforts are successful if readers can still describe the main contribution of your paper to their colleagues a year after reading it. Although it is clear that a paper often needs to communicate a number of innovations on the way to its final message, it does not pay to be greedy. Focus on a single message; papers that simultaneously focus on multiple contributions tend to be less convincing about each and are therefore less memorable.

The most important element of a paper is the title—think of the ratio of the number of titles you read to the number of papers you read. The title is typically the first element a reader encounters, so its quality [3] determines whether the reader will invest time in reading the abstract.

The title not only transmits the paper's central contribution but can also serve as a constant reminder (to you) to focus the text on transmitting that idea. Science is, after all, the abstraction of simple principles from complex data. The title is the ultimate refinement of the paper's contribution. Thinking about the title early—and regularly returning to hone it—can help not only the writing of the paper but also the process of designing experiments or developing theories.

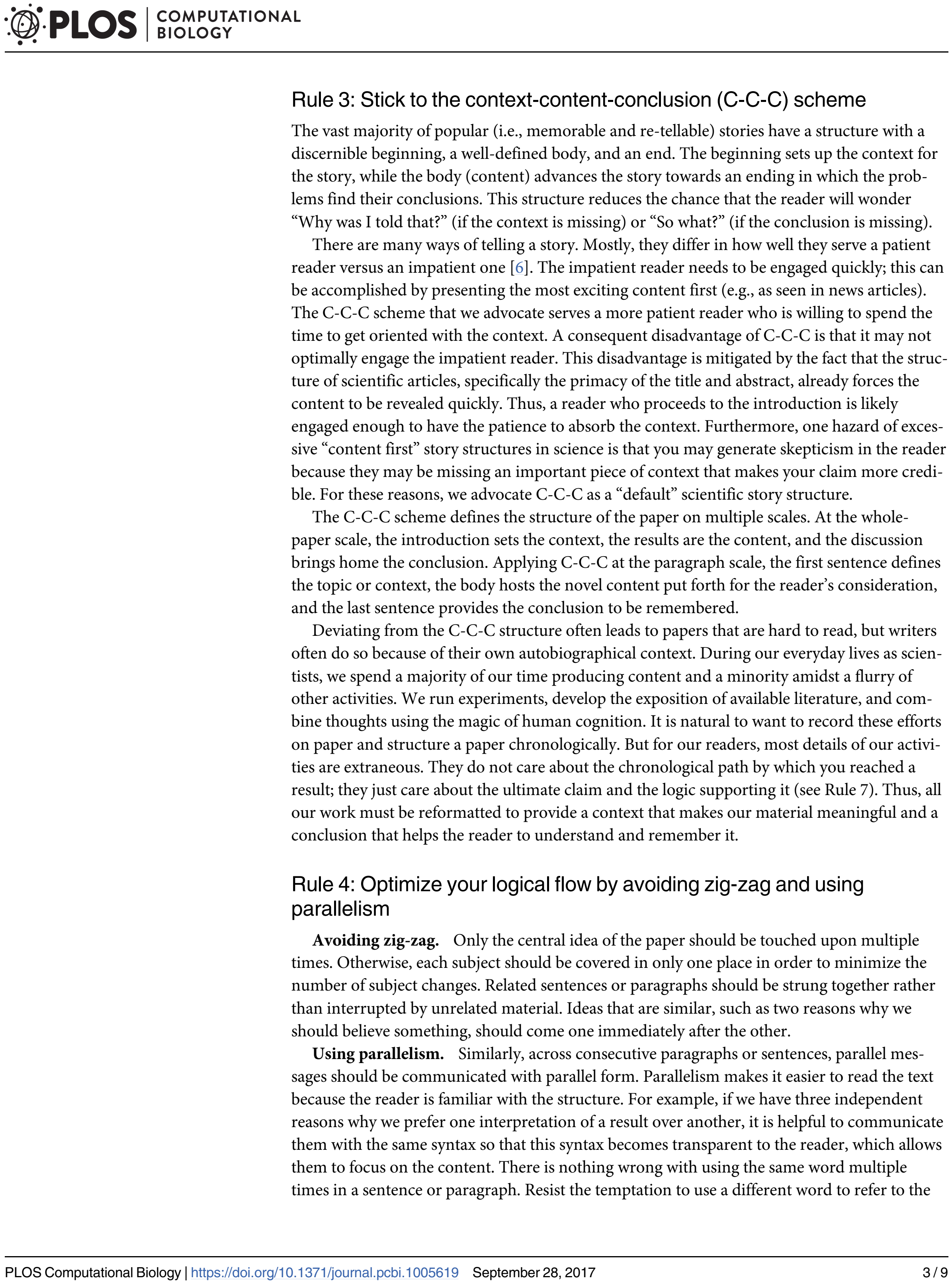
This Rule of One is the most difficult rule to optimally implement because it comes face-to-face with the key challenge of science, which is to make the claim and/or model as simple as the data and logic can support but no simpler. In the end, your struggle to find this balance may appropriately result in "one contribution" that is multifaceted. For example, a technology paper may describe both its new technology and a biological result using it; the bridge that unifies these two facets is a clear description of how the new technology can be used to do new biology.

Rule 2: Write for flesh-and-blood human beings who do not know your work

Because you are the world's leading expert at exactly what you are doing, you are also the world's least qualified person to judge your writing from the perspective of the naïve reader. The majority of writing mistakes stem from this predicament. Think like a designer—for each element, determine the impact that you want to have on people and then strive to achieve that objective [4]. Try to think through the paper like a naïve reader who must first be made to care about the problem you are addressing (see Rule 6) and then will want to understand your answer with minimal effort.

Define technical terms clearly because readers can become frustrated when they encounter a word that they don't understand. Avoid abbreviations and acronyms so that readers do not have to go back to earlier sections to identify them.

The vast knowledge base of human psychology is useful in paper writing. For example, people have working memory constraints in that they can only remember a small number of items and are better at remembering the beginning and the end of a list than the middle [5]. Do your best to minimize the number of loose threads that the reader has to keep in mind at any one time.



Rule 3: Stick to the context-content-conclusion (C-C-C) scheme

The vast majority of popular (i.e., memorable and re-tellable) stories have a structure with a discernible beginning, a well-defined body, and an end. The beginning sets up the context for the story, while the body (content) advances the story towards an ending in which the problems find their conclusions. This structure reduces the chance that the reader will wonder "Why was I told that?" (if the context is missing) or "So what?" (if the conclusion is missing).

There are many ways of telling a story. Mostly, they differ in how well they serve a patient reader versus an impatient one [6]. The impatient reader needs to be engaged quickly; this can be accomplished by presenting the most exciting content first (e.g., as seen in news articles). The C-C-C scheme that we advocate serves a more patient reader who is willing to spend the time to get oriented with the context. A consequent disadvantage of C-C-C is that it may not optimally engage the impatient reader. This disadvantage is mitigated by the fact that the structure of scientific articles, specifically the primacy of the title and abstract, already forces the content to be revealed quickly. Thus, a reader who proceeds to the introduction is likely engaged enough to have the patience to absorb the context. Furthermore, one hazard of excessive "content first" story structures in science is that you may generate skepticism in the reader because they may be missing an important piece of context that makes your claim more credible. For these reasons, we advocate C-C-C as a "default" scientific story structure.

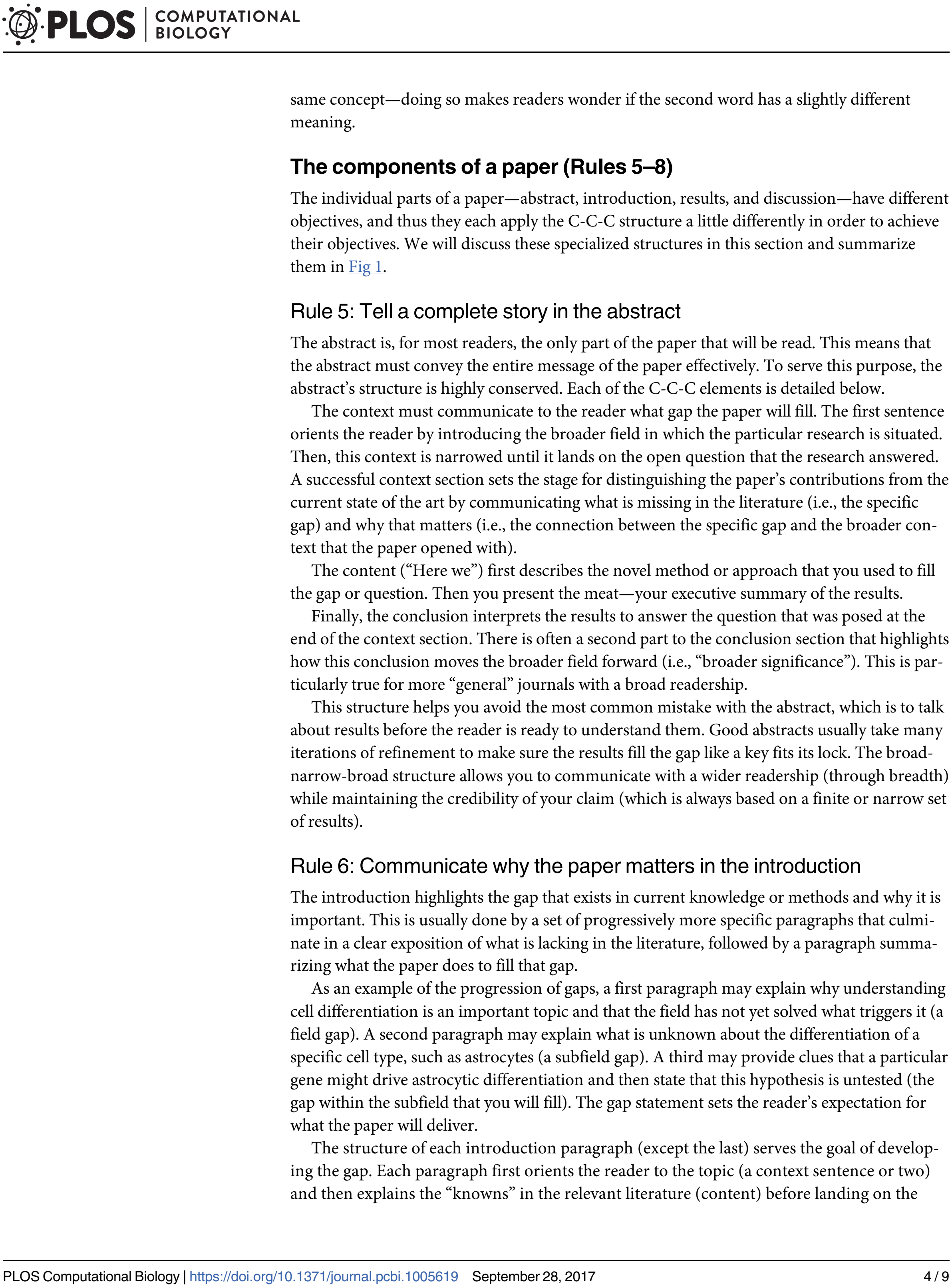
The C-C-C scheme defines the structure of the paper on multiple scales. At the whole-paper scale, the introduction sets the context, the results are the content, and the discussion brings home the conclusion. Applying C-C-C at the paragraph scale, the first sentence defines the topic or context, the body hosts the novel content put forth for the reader's consideration, and the last sentence provides the conclusion to be remembered.

Deviating from the C-C-C structure often leads to papers that are hard to read, but writers often do so because of their own autobiographical context. During our everyday lives as scientists, we spend a majority of our time producing content and a minority amidst a flurry of other activities. We run experiments, develop the exposition of available literature, and combine thoughts using the magic of human cognition. It is natural to want to record these efforts on paper and structure a paper chronologically. But for our readers, most details of our activities are extraneous. They do not care about the chronological path by which you reached a result; they just care about the ultimate claim and the logic supporting it (see Rule 7). Thus, all our work must be reformatted to provide a context that makes our material meaningful and a conclusion that helps the reader to understand and remember it.

Rule 4: Optimize your logical flow by avoiding zig-zag and using parallelism

Avoiding zig-zag. Only the central idea of the paper should be touched upon multiple times. Otherwise, each subject should be covered in only one place in order to minimize the number of subject changes. Related sentences or paragraphs should be strung together rather than interrupted by unrelated material. Ideas that are similar, such as two reasons why we should believe something, should come one immediately after the other.

Using parallelism. Similarly, across consecutive paragraphs or sentences, parallel messages should be communicated with parallel form. Parallelism makes it easier to read the text because the reader is familiar with the structure. For example, if we have three independent reasons why we prefer one interpretation of a result over another, it is helpful to communicate them with the same syntax so that this syntax becomes transparent to the reader, which allows them to focus on the content. There is nothing wrong with using the same word multiple times in a sentence or paragraph. Resist the temptation to use a different word to refer to the



same concept—doing so makes readers wonder if the second word has a slightly different meaning.

The components of a paper (Rules 5–8)

The individual parts of a paper—abstract, introduction, results, and discussion—have different objectives, and thus they each apply the C-C-C structure a little differently in order to achieve their objectives. We will discuss these specialized structures in this section and summarize them in Fig 1.

Rule 5: Tell a complete story in the abstract

The abstract is, for most readers, the only part of the paper that will be read. This means that the abstract must convey the entire message of the paper effectively. To serve this purpose, the abstract's structure is highly conserved. Each of the C-C-C elements is detailed below.

The context must communicate to the reader what gap the paper will fill. The first sentence orients the reader by introducing the broader field in which the particular research is situated. Then, this context is narrowed until it lands on the open question that the research answered. A successful context section sets the stage for distinguishing the paper's contributions from the current state of the art by communicating what is missing in the literature (i.e., the specific gap) and why that matters (i.e., the connection between the specific gap and the broader context that the paper opened with).

The content ("Here we") first describes the novel method or approach that you used to fill the gap or question. Then you present the meat—your executive summary of the results.

Finally, the conclusion interprets the results to answer the question that was posed at the end of the context section. There is often a second part to the conclusion section that highlights how this conclusion moves the broader field forward (i.e., "broader significance"). This is particularly true for more "general" journals with a broad readership.

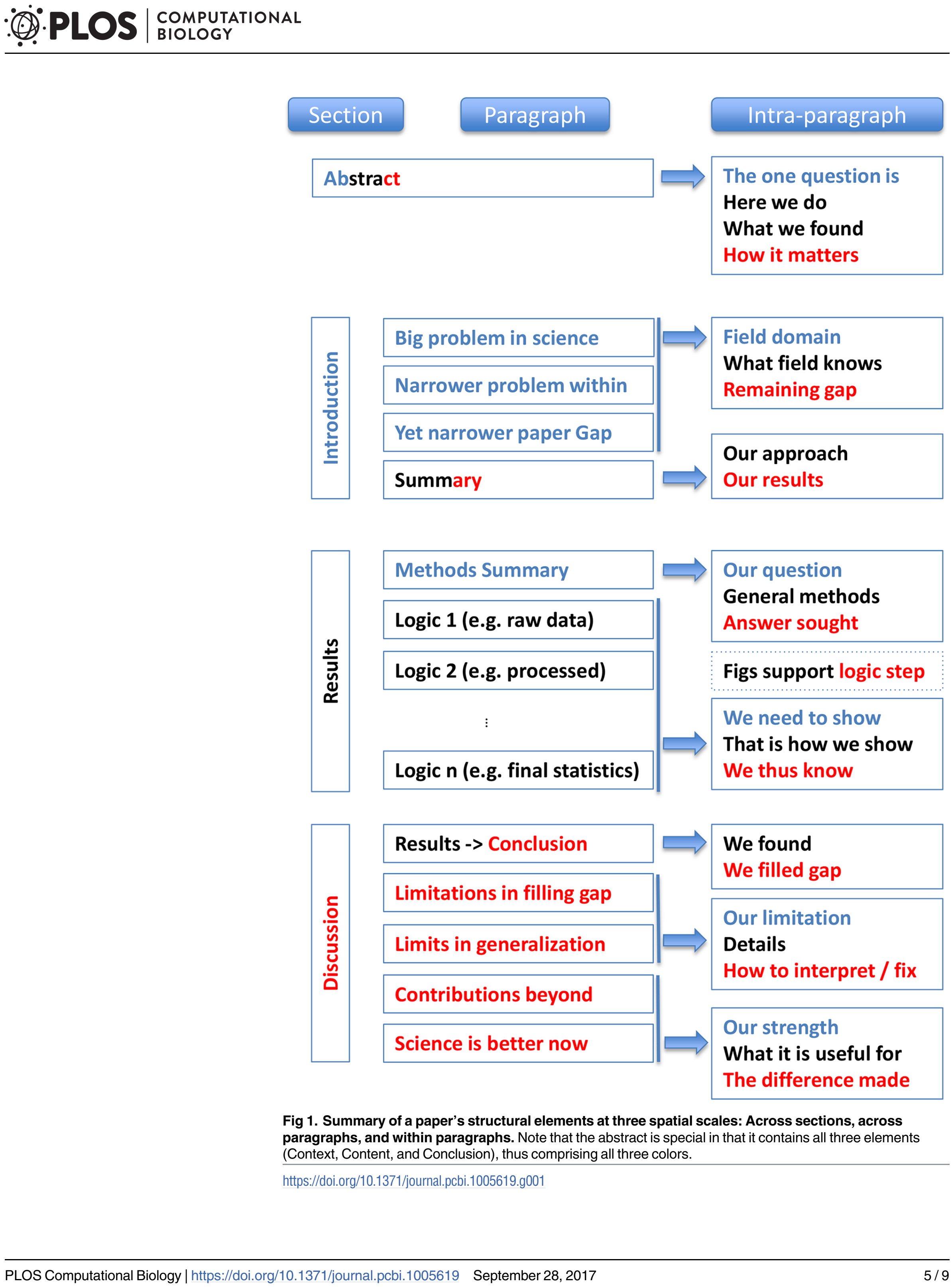
This structure helps you avoid the most common mistake with the abstract, which is to talk about results before the reader is ready to understand them. Good abstracts usually take many iterations of refinement to make sure the results fill the gap like a key fits its lock. The broad-narrow-broad structure allows you to communicate with a wider readership (through breadth) while maintaining the credibility of your claim (which is always based on a finite or narrow set of results).

Rule 6: Communicate why the paper matters in the introduction

The introduction highlights the gap that exists in current knowledge or methods and why it is important. This is usually done by a set of progressively more specific paragraphs that culminate in a clear exposition of what is lacking in the literature, followed by a paragraph summarizing what the paper does to fill that gap.

As an example of the progression of gaps, a first paragraph may explain why understanding cell differentiation is an important topic and that the field has not yet solved what triggers it (a field gap). A second paragraph may explain what is unknown about the differentiation of a specific cell type, such as astrocytes (a subfield gap). A third may provide clues that a particular gene might drive astrocytic differentiation and then state that this hypothesis is untested (the gap within the subfield that you will fill). The gap statement sets the reader's expectation for what the paper will deliver.

The structure of each introduction paragraph (except the last) serves the goal of developing the gap. Each paragraph first orients the reader to the topic (a context sentence or two) and then explains the "knowns" in the relevant literature (content) before landing on the



PLOS | COMPUTATIONAL BIOLOGY

Section

Paragraph

Intra-paragraph

Abstract

The one question is
Here we do
What we found
How it matters

Introduction

Big problem in science
Narrower problem within
Yet narrower paper Gap
Summary

Field domain
What field knows
Remaining gap

Our approach
Our results

Results

Methods Summary
Logic 1 (e.g. raw data)
Logic 2 (e.g. processed)
⋮
Logic n (e.g. final statistics)

Our question
General methods
Answer sought

Figs support logic step

We need to show
That is how we show
We thus know

Discussion

Results -> Conclusion
Limitations in filling gap
Limits in generalization
Contributions beyond
Science is better now

We found
We filled gap

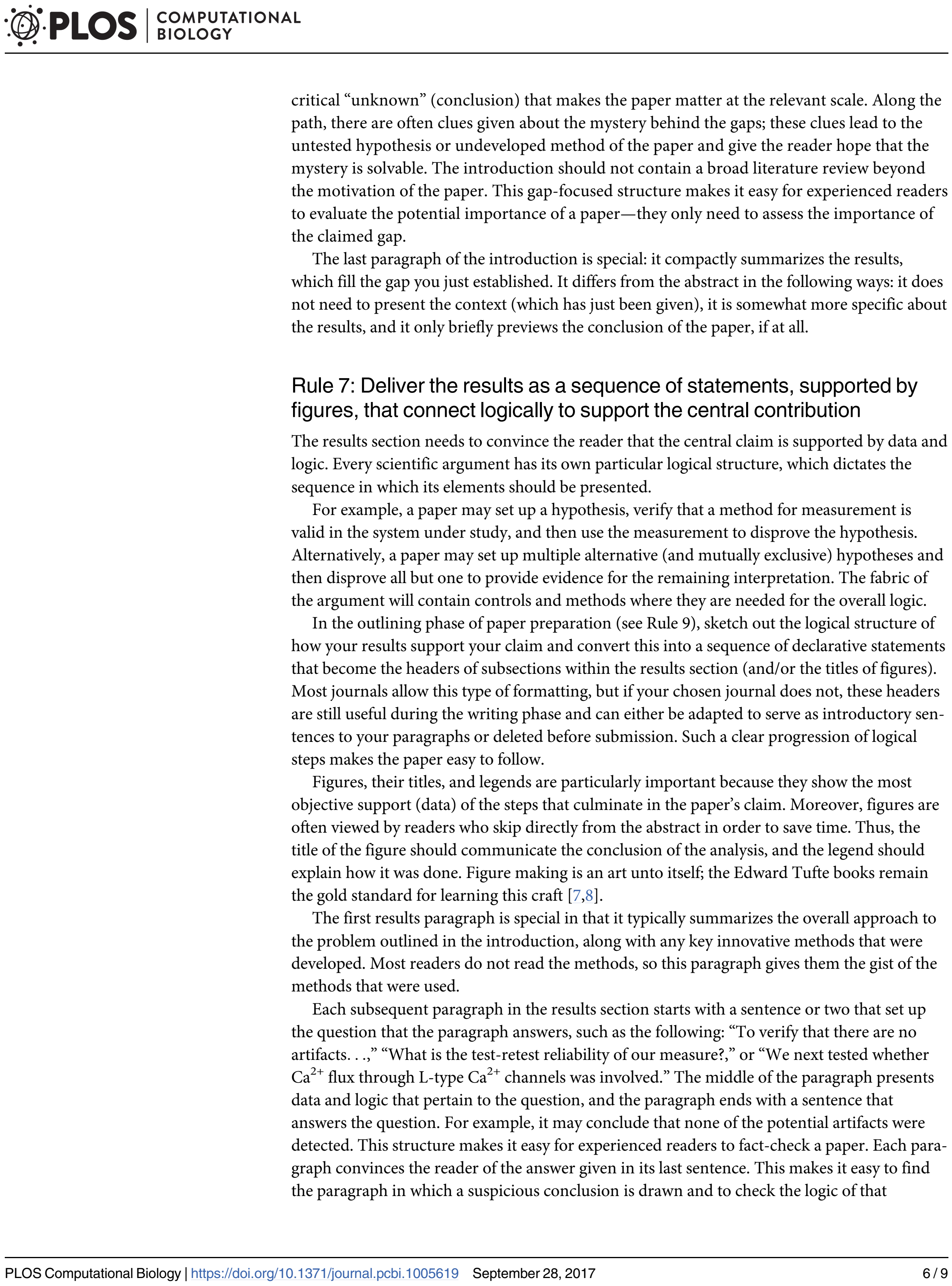
Our limitation
Details
How to interpret / fix

Our strength
What it is useful for
The difference made

Fig 1. Summary of a paper's structural elements at three spatial scales: Across sections, across paragraphs, and within paragraphs. Note that the abstract is special in that it contains all three elements (Context, Content, and Conclusion), thus comprising all three colors.

https://doi.org/10.1371/journal.pcbi.1005619.g001

PLOS Computational Biology | https://doi.org/10.1371/journal.pcbi.1005619 September 28, 2017 5 / 9



critical "unknown" (conclusion) that makes the paper matter at the relevant scale. Along the path, there are often clues given about the mystery behind the gaps; these clues lead to the untested hypothesis or undeveloped method of the paper and give the reader hope that the mystery is solvable. The introduction should not contain a broad literature review beyond the motivation of the paper. This gap-focused structure makes it easy for experienced readers to evaluate the potential importance of a paper—they only need to assess the importance of the claimed gap.

The last paragraph of the introduction is special: it compactly summarizes the results, which fill the gap you just established. It differs from the abstract in the following ways: it does not need to present the context (which has just been given), it is somewhat more specific about the results, and it only briefly previews the conclusion of the paper, if at all.

Rule 7: Deliver the results as a sequence of statements, supported by figures, that connect logically to support the central contribution

The results section needs to convince the reader that the central claim is supported by data and logic. Every scientific argument has its own particular logical structure, which dictates the sequence in which its elements should be presented.

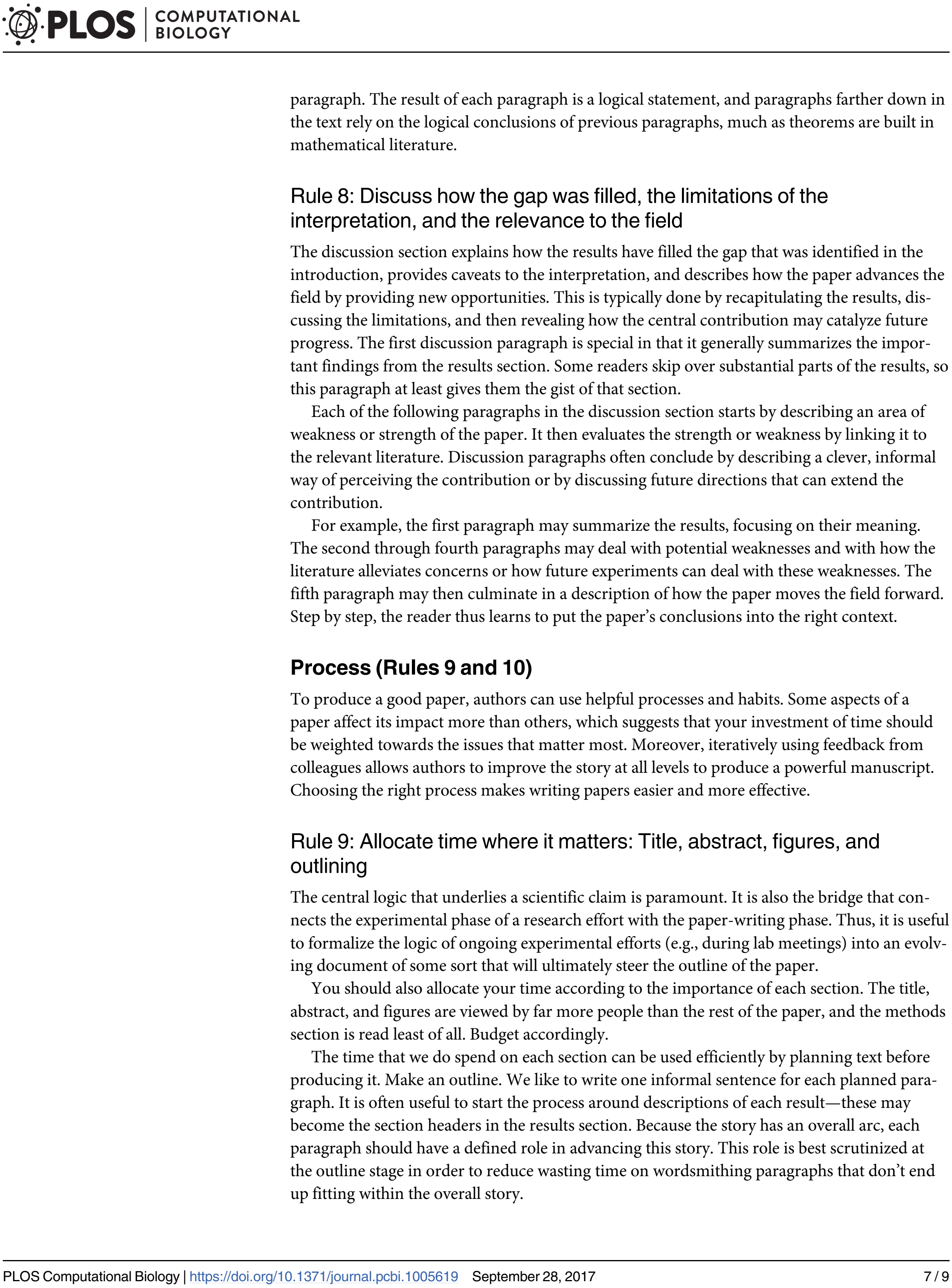
For example, a paper may set up a hypothesis, verify that a method for measurement is valid in the system under study, and then use the measurement to disprove the hypothesis. Alternatively, a paper may set up multiple alternative (and mutually exclusive) hypotheses and then disprove all but one to provide evidence for the remaining interpretation. The fabric of the argument will contain controls and methods where they are needed for the overall logic.

In the outlining phase of paper preparation (see Rule 9), sketch out the logical structure of how your results support your claim and convert this into a sequence of declarative statements that become the headers of subsections within the results section (and/or the titles of figures). Most journals allow this type of formatting, but if your chosen journal does not, these headers are still useful during the writing phase and can either be adapted to serve as introductory sentences to your paragraphs or deleted before submission. Such a clear progression of logical steps makes the paper easy to follow.

Figures, their titles, and legends are particularly important because they show the most objective support (data) of the steps that culminate in the paper's claim. Moreover, figures are often viewed by readers who skip directly from the abstract in order to save time. Thus, the title of the figure should communicate the conclusion of the analysis, and the legend should explain how it was done. Figure making is an art unto itself; the Edward Tufte books remain the gold standard for learning this craft [7,8].

The first results paragraph is special in that it typically summarizes the overall approach to the problem outlined in the introduction, along with any key innovative methods that were developed. Most readers do not read the methods, so this paragraph gives them the gist of the methods that were used.

Each subsequent paragraph in the results section starts with a sentence or two that set up the question that the paragraph answers, such as the following: "To verify that there are no artifacts. . .," "What is the test-retest reliability of our measure?," or "We next tested whether Ca^{2+} flux through L-type Ca^{2+} channels was involved." The middle of the paragraph presents data and logic that pertain to the question, and the paragraph ends with a sentence that answers the question. For example, it may conclude that none of the potential artifacts were detected. This structure makes it easy for experienced readers to fact-check a paper. Each paragraph convinces the reader of the answer given in its last sentence. This makes it easy to find the paragraph in which a suspicious conclusion is drawn and to check the logic of that



paragraph. The result of each paragraph is a logical statement, and paragraphs farther down in the text rely on the logical conclusions of previous paragraphs, much as theorems are built in mathematical literature.

Rule 8: Discuss how the gap was filled, the limitations of the interpretation, and the relevance to the field

The discussion section explains how the results have filled the gap that was identified in the introduction, provides caveats to the interpretation, and describes how the paper advances the field by providing new opportunities. This is typically done by recapitulating the results, discussing the limitations, and then revealing how the central contribution may catalyze future progress. The first discussion paragraph is special in that it generally summarizes the important findings from the results section. Some readers skip over substantial parts of the results, so this paragraph at least gives them the gist of that section.

Each of the following paragraphs in the discussion section starts by describing an area of weakness or strength of the paper. It then evaluates the strength or weakness by linking it to the relevant literature. Discussion paragraphs often conclude by describing a clever, informal way of perceiving the contribution or by discussing future directions that can extend the contribution.

For example, the first paragraph may summarize the results, focusing on their meaning. The second through fourth paragraphs may deal with potential weaknesses and with how the literature alleviates concerns or how future experiments can deal with these weaknesses. The fifth paragraph may then culminate in a description of how the paper moves the field forward. Step by step, the reader thus learns to put the paper's conclusions into the right context.

Process (Rules 9 and 10)

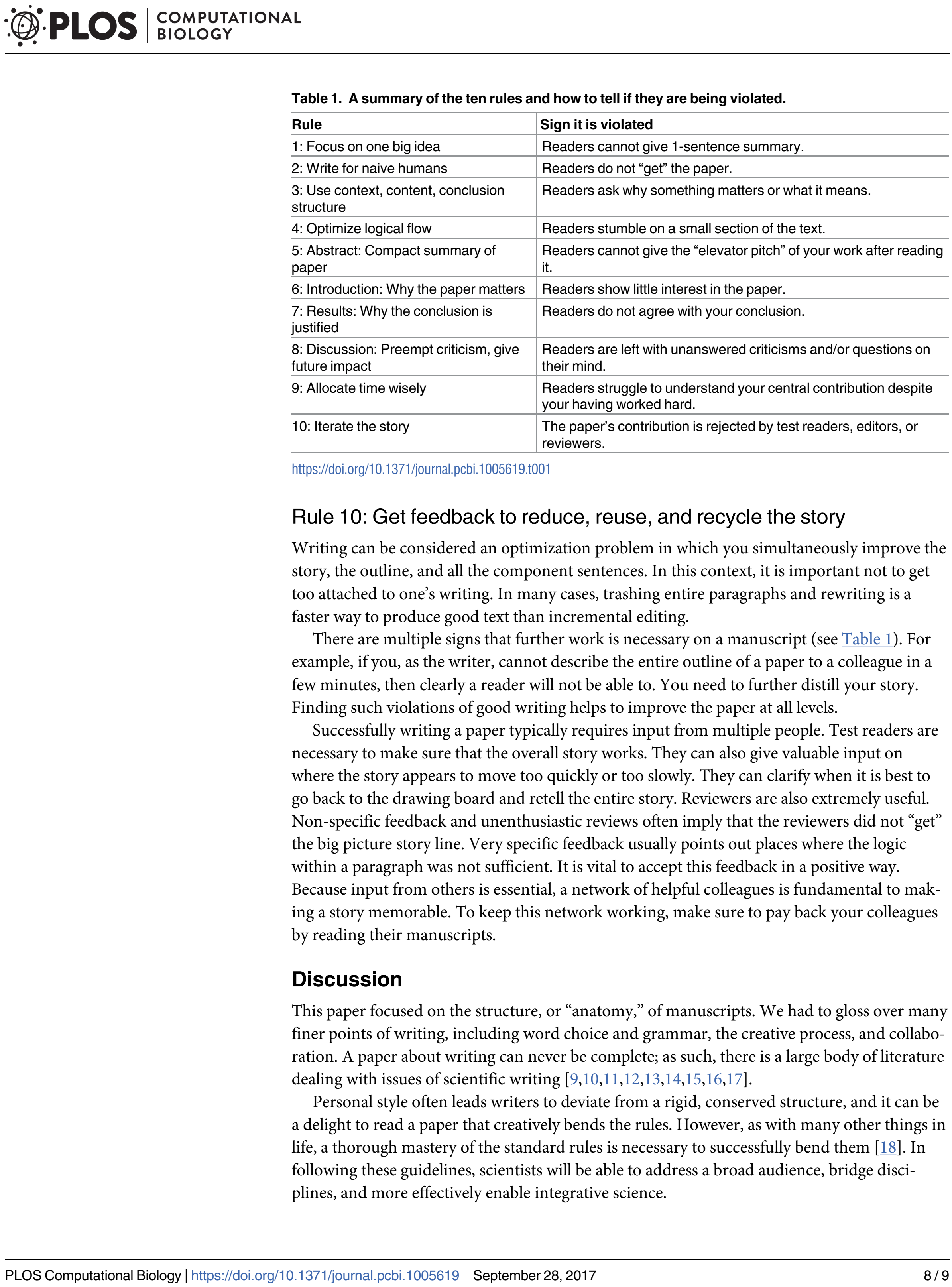
To produce a good paper, authors can use helpful processes and habits. Some aspects of a paper affect its impact more than others, which suggests that your investment of time should be weighted towards the issues that matter most. Moreover, iteratively using feedback from colleagues allows authors to improve the story at all levels to produce a powerful manuscript. Choosing the right process makes writing papers easier and more effective.

Rule 9: Allocate time where it matters: Title, abstract, figures, and outlining

The central logic that underlies a scientific claim is paramount. It is also the bridge that connects the experimental phase of a research effort with the paper-writing phase. Thus, it is useful to formalize the logic of ongoing experimental efforts (e.g., during lab meetings) into an evolving document of some sort that will ultimately steer the outline of the paper.

You should also allocate your time according to the importance of each section. The title, abstract, and figures are viewed by far more people than the rest of the paper, and the methods section is read least of all. Budget accordingly.

The time that we do spend on each section can be used efficiently by planning text before producing it. Make an outline. We like to write one informal sentence for each planned paragraph. It is often useful to start the process around descriptions of each result—these may become the section headers in the results section. Because the story has an overall arc, each paragraph should have a defined role in advancing this story. This role is best scrutinized at the outline stage in order to reduce wasting time on wordsmithing paragraphs that don't end up fitting within the overall story.



PLOS | COMPUTATIONAL BIOLOGY

Table 1. A summary of the ten rules and how to tell if they are being violated.

Rule	Sign it is violated
1: Focus on one big idea	Readers cannot give 1-sentence summary.
2: Write for naive humans	Readers do not "get" the paper.
3: Use context, content, conclusion structure	Readers ask why something matters or what it means.
4: Optimize logical flow	Readers stumble on a small section of the text.
5: Abstract: Compact summary of paper	Readers cannot give the "elevator pitch" of your work after reading it.
6: Introduction: Why the paper matters	Readers show little interest in the paper.
7: Results: Why the conclusion is justified	Readers do not agree with your conclusion.
8: Discussion: Preempt criticism, give future impact	Readers are left with unanswered criticisms and/or questions on their mind.
9: Allocate time wisely	Readers struggle to understand your central contribution despite your having worked hard.
10: Iterate the story	The paper's contribution is rejected by test readers, editors, or reviewers.

https://doi.org/10.1371/journal.pcbi.1005619.t001

Rule 10: Get feedback to reduce, reuse, and recycle the story

Writing can be considered an optimization problem in which you simultaneously improve the story, the outline, and all the component sentences. In this context, it is important not to get too attached to one's writing. In many cases, trashing entire paragraphs and rewriting is a faster way to produce good text than incremental editing.

There are multiple signs that further work is necessary on a manuscript (see Table 1). For example, if you, as the writer, cannot describe the entire outline of a paper to a colleague in a few minutes, then clearly a reader will not be able to. You need to further distill your story. Finding such violations of good writing helps to improve the paper at all levels.

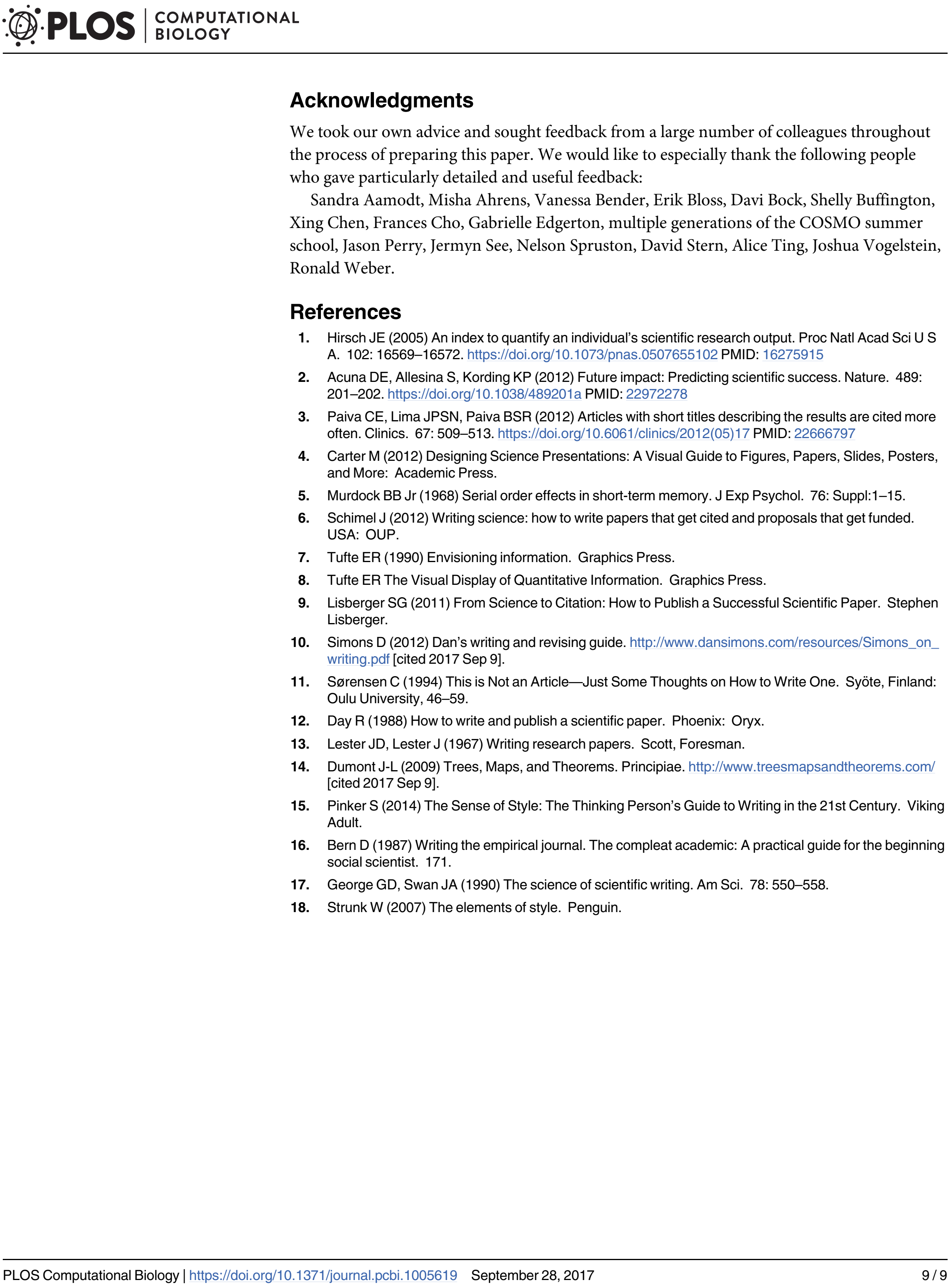
Successfully writing a paper typically requires input from multiple people. Test readers are necessary to make sure that the overall story works. They can also give valuable input on where the story appears to move too quickly or too slowly. They can clarify when it is best to go back to the drawing board and retell the entire story. Reviewers are also extremely useful. Non-specific feedback and unenthusiastic reviews often imply that the reviewers did not "get" the big picture story line. Very specific feedback usually points out places where the logic within a paragraph was not sufficient. It is vital to accept this feedback in a positive way. Because input from others is essential, a network of helpful colleagues is fundamental to making a story memorable. To keep this network working, make sure to pay back your colleagues by reading their manuscripts.

Discussion

This paper focused on the structure, or "anatomy," of manuscripts. We had to gloss over many finer points of writing, including word choice and grammar, the creative process, and collaboration. A paper about writing can never be complete; as such, there is a large body of literature dealing with issues of scientific writing [9,10,11,12,13,14,15,16,17].

Personal style often leads writers to deviate from a rigid, conserved structure, and it can be a delight to read a paper that creatively bends the rules. However, as with many other things in life, a thorough mastery of the standard rules is necessary to successfully bend them [18]. In following these guidelines, scientists will be able to address a broad audience, bridge disciplines, and more effectively enable integrative science.

PLOS Computational Biology | https://doi.org/10.1371/journal.pcbi.1005619 September 28, 2017 8 / 9



Acknowledgments

We took our own advice and sought feedback from a large number of colleagues throughout the process of preparing this paper. We would like to especially thank the following people who gave particularly detailed and useful feedback:

Sandra Aamodt, Misha Ahrens, Vanessa Bender, Erik Bloss, Davi Bock, Shelly Buffington, Xing Chen, Frances Cho, Gabrielle Edgerton, multiple generations of the COSMO summer school, Jason Perry, Jermyn See, Nelson Spruston, David Stern, Alice Ting, Joshua Vogelstein, Ronald Weber.

References

1. Hirsch JE (2005) An index to quantify an individual's scientific research output. Proc Natl Acad Sci U S A. 102: 16569–16572. https://doi.org/10.1073/pnas.0507655102 PMID: 16275915
2. Acuna DE, Allesina S, Kording KP (2012) Future impact: Predicting scientific success. Nature. 489: 201–202. https://doi.org/10.1038/489201a PMID: 22972278
3. Paiva CE, Lima JPSN, Paiva BSR (2012) Articles with short titles describing the results are cited more often. Clinics. 67: 509–513. https://doi.org/10.6061/clinics/2012(05)17 PMID: 22666797
4. Carter M (2012) Designing Science Presentations: A Visual Guide to Figures, Papers, Slides, Posters, and More: Academic Press.
5. Murdock BB Jr (1968) Serial order effects in short-term memory. J Exp Psychol. 76: Suppl:1–15.
6. Schimel J (2012) Writing science: how to write papers that get cited and proposals that get funded. USA: OUP.
7. Tufte ER (1990) Envisioning information. Graphics Press.
8. Tufte ER The Visual Display of Quantitative Information. Graphics Press.
9. Lisberger SG (2011) From Science to Citation: How to Publish a Successful Scientific Paper. Stephen Lisberger.
10. Simons D (2012) Dan's writing and revising guide. http://www.dansimons.com/resources/Simons_on_writing.pdf [cited 2017 Sep 9].
11. Sørensen C (1994) This is Not an Article—Just Some Thoughts on How to Write One. Syöte, Finland: Oulu University, 46–59.
12. Day R (1988) How to write and publish a scientific paper. Phoenix: Oryx.
13. Lester JD, Lester J (1967) Writing research papers. Scott, Foresman.
14. Dumont J-L (2009) Trees, Maps, and Theorems. Principiae. http://www.treesmapsandtheorems.com/ [cited 2017 Sep 9].
15. Pinker S (2014) The Sense of Style: The Thinking Person's Guide to Writing in the 21st Century. Viking Adult.
16. Bern D (1987) Writing the empirical journal. The compleat academic: A practical guide for the beginning social scientist. 171.
17. George GD, Swan JA (1990) The science of scientific writing. Am Sci. 78: 550–558.
18. Strunk W (2007) The elements of style. Penguin.

5.2 Customizing referencing

5.2.1 Using a .csl file with pandoc instead of biblatex

The `oxforddown` package uses biblatex in latex for referencing. It is also possible to use pandec for referencing by providing a .csl file in the YAML header of **index.Rmd** (likely requiring commenting out the biblatex code in **template.Rmd**). This may be helpful for those who have a .csl file describing the referencing format for a particular journal. However, note that this approach does not support chapter bibliographies (see Section 5.2.2).

```
csl: ecology.csl
```

5.2.2 Customizing biblatex and adding chapter bibliographies

This section provides one example of customizing biblatex. Much of this code was combined from searches on Stack Exchange and other sources (e.g. here).

In **template.tex**, one can replace the existing biblatex calls with the following to achieve referencing that looks like this:

(Charmantier and Gienapp 2014)

Charmantier, A. and P. Gienapp (2014). Climate change and timing of avian breeding and migration: evolutionary versus plastic changes. Evolutionary Applications 7(1):15–28. doi: 10.1111/eva.12126.

```
\usepackage[backend=biber,
    bibencoding=utf8,
    refsection=chapter, % referencing by chapter
    style=authoryear,
    firstinits=true,
    isbn=false,
    doi=true,
    url=false,
    eprint=false,
    related=false,
    dashed=false,
    clearlang=true,
    maxcitenames=2,
    mincitenames=1,
    maxbibnames=10,
    abbreviate=false,
    minbibnames=3,
    uniquelist=minyear,
    sortcites=true,
    date=year
]{biblatex}
\AtEveryBibitem{%
  \clearlist{language}%
  \clearfield{note}
}

\DeclareFieldFormat{titlecase}{\MakeTitleCase{#1}}

\newrobustcmd{\MakeTitleCase}[1]{%
  \ifthenelse{\ifcurrentfield{booktitle}\OR\ifcurrentfield{booksubtitle}%
    \OR\ifcurrentfield{maintitle}\OR\ifcurrentfield{mainsubtitle}%
    \OR\ifcurrentfield{journaltitle}\OR\ifcurrentfield{journalsubtitle}%
    \OR\ifcurrentfield{issuetitle}\OR\ifcurrentfield{issuesubtitle}%
    \OR\ifentrytype{book}\OR\ifentrytype{mvbook}\OR\ifentrytype{bookinbook}%
    \OR\ifentrytype{booklet}\OR\ifentrytype{suppbook}%
    \OR\ifentrytype{collection}\OR\ifentrytype{mvcollection}%
    \OR\ifentrytype{suppcollection}\OR\ifentrytype{manual}%
    \OR\ifentrytype{periodical}\OR\ifentrytype{suppperiodical}%
    \OR\ifentrytype{proceedings}\OR\ifentrytype{mvproceedings}%
    \OR\ifentrytype{reference}\OR\ifentrytype{mvreference}%
    \OR\ifentrytype{report}\OR\ifentrytype{thesis}}
    {#1}
    {\MakeSentenceCase{#1}}}

% \renewbibmacro{in:}{}
% suppress "in" for articles
%
\renewbibmacro{in:}{%
  \ifentrytype{article}{}{\printtext{\bibstring{in}\intitlepunct}}}
%-- no "quotes" around titles of chapters/article titles
\DeclareFieldFormat[article, inbook, incollection, inproceedings, misc, thesis, unp
{title}{#1}
%-- no punctuation after volume
\DeclareFieldFormat[article]
{volume}{{#1}}
%-- puts number/issue between brackets
\DeclareFieldFormat[article, inbook, incollection, inproceedings, misc, thesis, unp
{number}{\mkbibparens{#1}}
%-- and then for articles directly the pages w/o any "pages" or "pp."
\DeclareFieldFormat[article]
{pages}{#1}
%-- for some types replace "pages" by "p."
\DeclareFieldFormat[inproceedings, incollection, inbook]
{pages}{p. #1}
%-- format 16(4):224--225 for articles
\renewbibmacro*{volume+number+eid}{
  \printfield{volume}%
  \printfield{number}%
  \printunit{\addcolon}
}
```

If you would like chapter bibliographies, in addition insert the following code at the end of each chapter, and comment out the entire REFERENCES section at the end of template.tex.

```
\printbibliography[segment=\therefsection,heading=subbibliography]
```

5.3 Customizing the page headers and footers

The following code, when it replaces the existing correpsonding code block in **ociamthesis.cls**, puts chapter number and title centered in the header and page number in the footer, centered. This may be desirable particularly when inserting PDF chapters, as the margins of the PDF may not exactly align with the left and right margins of the page, demarcated by the existing header and footer text. In the following code block, the original code is commented out where replaced.

```
\usepackage{fancyhdr}
\setlength{\headheight}{15pt}
\fancyhf{} % clear the header and footers
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{\markboth{\thechapter. #1}{\thechapter. #1}}
% \renewcommand{\chaptermark}[1]{\markboth{\thechapter. #1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection. #1}}
\renewcommand{\headrulewidth}{0pt}
\fancyhead[CO]{\emph{\leftmark}}
\fancyhead[CE]{\emph{\rightmark}}
% \fancyhead[LO,RE]{}
% \fancyhead[LE,RO]{}
\fancyfoot[CO,CE]{\emph{\thepage}}

\fancypagestyle{plain}{\fancyhf{}\fancyfoot[C]{\emph{\thepage}}}
```

```
% JEM fix header on cleared pages for openright
\def\cleardoublepage{\clearpage\if@twoside \ifodd\c@page\else
   \hbox{}
   % \fancyhead[RE,LO]{}
   \fancyhead[CE,CO]{}
   \newpage
   \if@twocolumn\hbox{}\newpage
   \fi
   % \fancyhead[LO]{\emph{\leftmark}}
   % \fancyhead[RE]{\emph{\rightmark}}
   \fancyhead[CO]{\emph{\leftmark}}
   \fancyhead[CE]{\emph{\rightmark}}
   \fi\fi}
```

Alles Gescheite ist schon gedacht worden.
Man muss nur versuchen, es noch einmal zu denken.

All intelligent thoughts have already been thought;
what is necessary is only to try to think them again.

— Johann Wolfgang von Goethe (von Goethe 1829)

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{-}` attribute.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Appendices

A
The First Appendix

This first appendix includes an R chunk that was hidden in the document (using `echo = FALSE`) to help with readibility:

In 02-rmd-basics-code.Rmd

```r
library(tidyverse)
knitr::include_graphics("figures/chunk-parts.png")
```

And here's another one from the same chapter, i.e. Chapter 2:

```r
knitr::include_graphics("figures/beltcrest.png")
```

B
The Second Appendix, for Fun

Works Cited

Darwin, Charles (1859). *On the Origin of Species by Means of Natural Selection or the Preservation of Favoured Races in the Struggle for Life*. London: John Murray.

Lottridge, Danielle et al. (2012). "Browser design impacts multitasking". In: *Proceedings of the Human Factors and Ergonomics Society 56th Annual Meeting*. DOI: 10.1177/1071181312561289.

Shea, Nicholas et al. (2014). "Supra-personal cognitive control and metacognition". In: *Trends in Cognitive Sciences* 18.4, pp. 186–193. DOI: 10.1016/j.tics.2014.01.006. URL: http://dx.doi.org/10.1016/j.tics.2014.01.006.

Von Goethe, Johann Wolfgang (1829). *Wilhelm Meisters Wanderjahre oder die Entsagenden*. de. Cotta.

Wu, Tim (2016). *The Attention Merchants: The Epic Scramble to Get Inside Our Heads*. Knopf Publishing Group.