stormdown:  
A Harrisburg University Dissertation  
Template for R Markdown

Author Name

2020-12-01

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  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](https://ulyngs.github.io/rmarkdown-workshop-2019/).

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!

Table of Contents

# Introduction

Welcome to the *R Markdown* Oxford University thesis template. This sample content is adapted from [thesisdown](https://github.com/ismayc/thesisdown) and the formatting of PDF output is adapted from the [OxThesis LaTeX template](https://github.com/mcmanigle/OxThesis). 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’](https://bookdown.org/yihui/rmarkdown/language-engines.html) in Yihui Xie’s [*R Markdown: The Definitive Guide*](https://bookdown.org/yihui/rmarkdown/language-engines.html).

## 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.

# 1 R Markdown Basics: The Markdown syntax

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

Sub2 and super2 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](https://www.google.com) 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]](#footnote-43) by writing either ^[my footnote text] for supplying the footnote content inline, or something like [^a-random-footnote-label] and supplying the text elsewhere in the format shown below:[[2]](#footnote-44)

[^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:

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

Becomes:

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:

For more (e.g. how to theorems), see e.g. the documentation on [bookdown.org](https://bookdown.org/yihui/bookdown/markdown-extensions-by-bookdown.html#equations)

## 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

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.

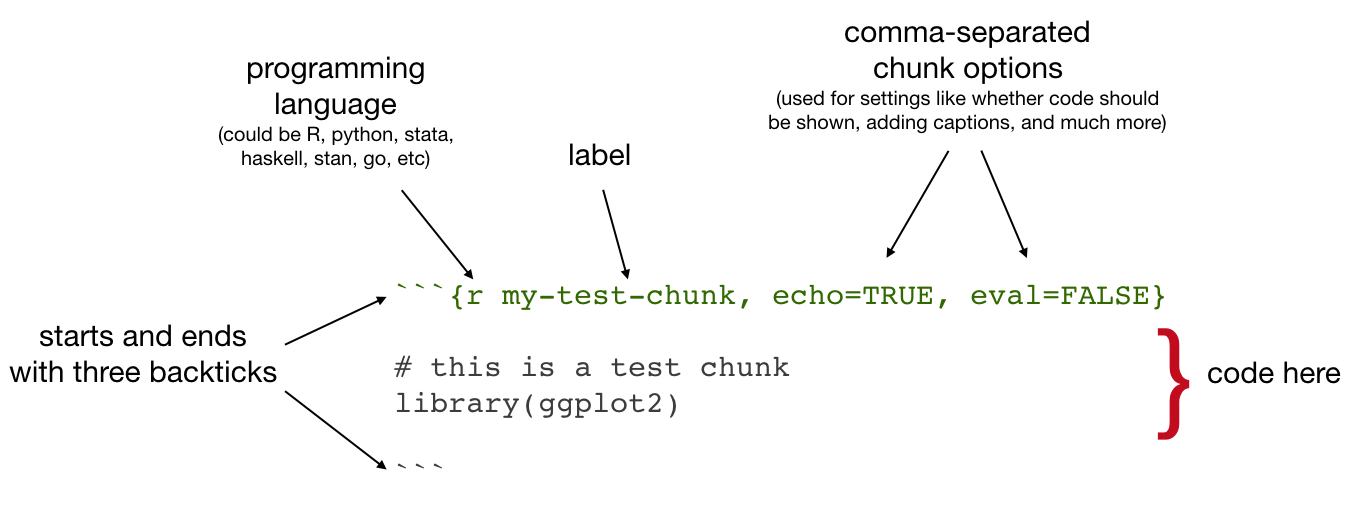


Figure 2.1: Code chunk syntax

Common chunk options include (see e.g. [bookdown.org](https://bookdown.org/yihui/rmarkdown/r-code.html)):

* 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)  
  
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")



Figure 2.2: Oxford logo

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:

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.



Figure 2.3: Oxford logo, rotated

### 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()

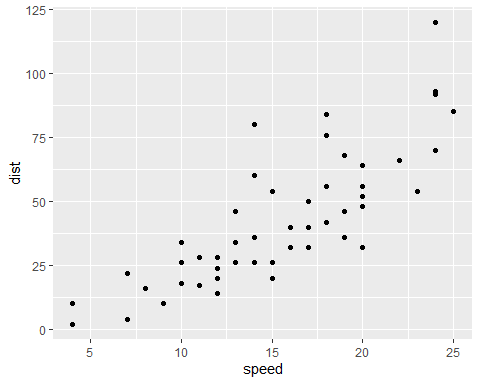


Figure 2.4: A ggplot of car stuff

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")

Table 2.1: A knitr kable table

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

* Gotcha: when using [kable](https://www.rdocumentation.org/packages/knitr/versions/1.21/topics/kable), captions are set inside the kable function
* The kable package is often used with the [kableExtra](https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html) package

### 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](https://bookdown.org/yihui/rmarkdown/) 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

## 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.[[3]](#footnote-70) Then reference the path to this file in **index.Rmd**’s YAML header with bibliography: example.bib.

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](https://www.zotero.org) with the [Better BibTex plug-in](https://retorque.re/zotero-better-bibtex/), 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](#ref-Shea2014); Lottridge et al. [2012](#ref-Lottridge2012)).

### 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](#ref-Shea2014)) says blah.
* Include only the citation-year (in parenthesis)
  + This: Shea et al. says blah [-@Shea2014]
  + Becomes: Shea et al. says blah ([2014](#ref-Shea2014))
* 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](#ref-Shea2014), 33–35; also Wu [2016](#ref-Wu2016), 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](https://www.zotero.org/styles).

### 3.1.3 Insert references easily with the citr add-in

For an easy way to insert citations, try the [citr](https://github.com/crsh/citr) RStudio add-in (Figure 3.1). You can install this add-in by typing install.packages("citr") in the R Console.

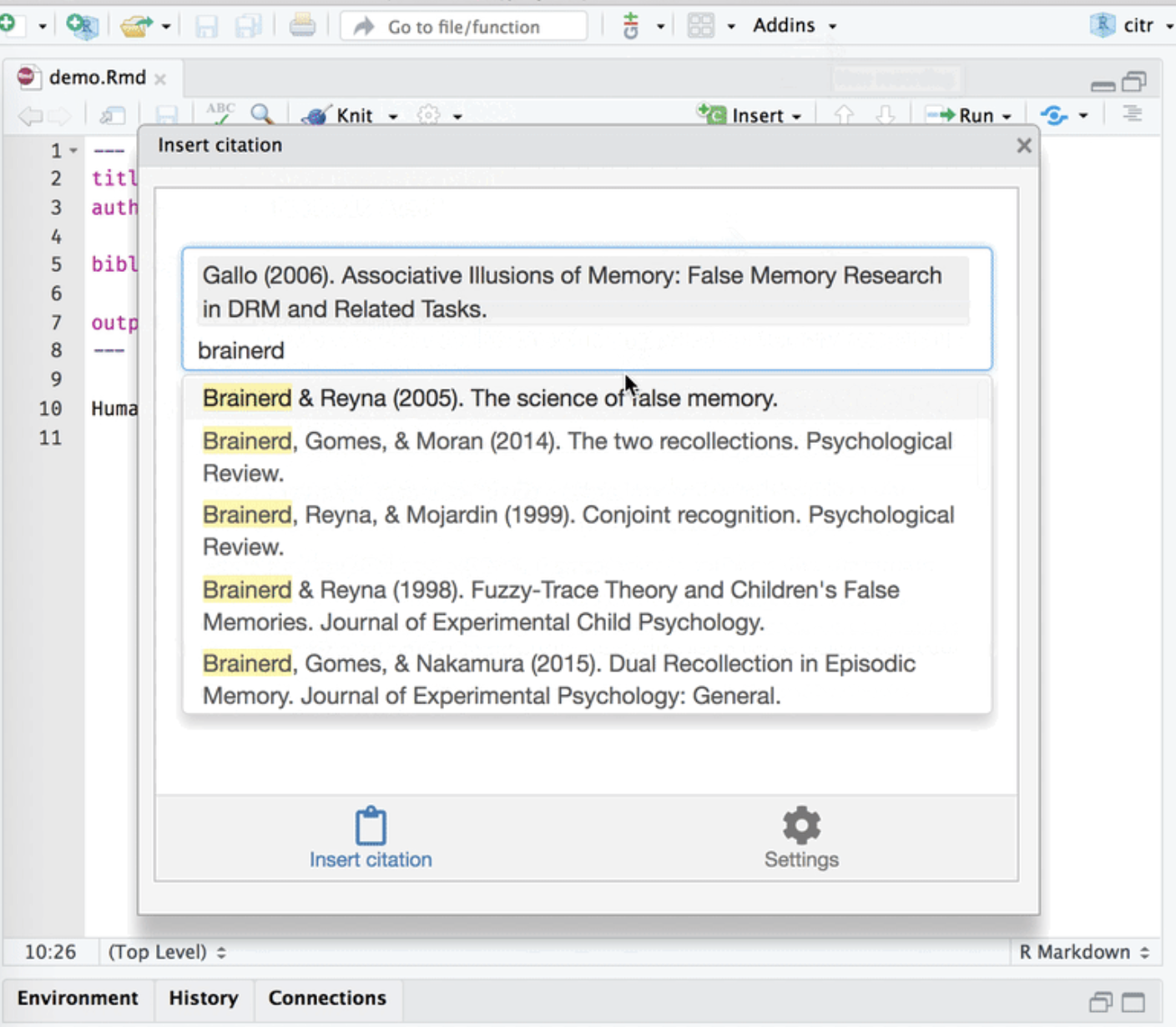


Figure 3.1: The citr add-in

## 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](#citations)?

#### 3.2.1.1 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](#cites-and-refs) 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:

knitr::include\_graphics("figures/captain.jpeg")



Figure 3.2: A marvel-lous meme

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

And in Figure 2.4 we saw a [cars plot](#fig:cars-plot).

### 3.2.3 Table references

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

Let’s include a table:

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

Table 3.1: Stopping cars

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

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

And in Table 2.1 we saw more or less [the same cars table](#tab: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 <a href="#fig:cars-plot">2.4</a> on page \pageref(fig:cars-plot)
* Becomes: Figure 2.4 on page

#### 3.2.4.1 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 \<a href="#fig:cars-plot">2.4</a> on page \\pageref{fig:cars-plot}", "")`
* Inserts this (check this on both PDF and gitbook):

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}

# 4 Final Notes on The OxThesis template and on collaboration

## 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'}.[[4]](#footnote-94)

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.

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’](https://bookdown.org/yihui/bookdown/markdown-extensions-by-bookdown.html#text-references): 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'}.[[5]](#footnote-100) Like so:

PK q

### 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).

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**](https://livefreeordichotomize.com/2018/09/14/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](https://github.com/rstudio/rmarkdown/issues/1463) on GitHub, where [CriticMarkup](http://criticmarkup.com) 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](https://www.overleaf.com/), then collaboratively make final edits to the  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("figures/pdf\_example/Lyngs2020\_FB.pdf")

In LaTeX output, however, this approach can cause odd behaviour. Therefore, when you build your thesis to PDF, split the PDF into an alphanumerically sorted sequence of **single-page** PDF files (you can do this automatically with the package pdftools). You can then use the appropriate LaTeX command to insert them, as shown below (for brevity, in the oxforddown PDF sample content we’re only including two pages) *Note that the chunk option results='asis' must be set.* You may also want to remove margins from the PDF files, which you can do with Adobe Acrobat (paid version) and likely other software.

# install.packages(pdftools)  
# split PDF into pages stored in figures/pdf\_example/split/  
# pdftools::pdf\_split("figures/pdf\_example/Lyngs2020\_FB.pdf",  
# output = "figures/pdf\_example/split/")  
  
# grab the pages  
pages <- list.files("figures/pdf\_example/split", full.names = TRUE)  
  
# set how wide you want the inserted PDFs to be:   
# 1.0 is 100 per cent of the oxforddown PDF page width;  
# you may want to make it a bit bigger  
pdf\_width <- 1.2  
  
# for each PDF page, insert it nicely and  
# end with a page break  
cat(stringr::str\_c("\\newpage \\begin{center} \\makebox[\\linewidth][c]{\\includegraphics[width=", pdf\_width, "\\linewidth]{", pages, "}} \\end{center}"))

## 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 pandoc for referencing by providing a .csl file in the YAML header of **index.Rmd** (likely requiring commenting out the biblatex code in **templates/template.tex**). 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](https://tex.stackexchange.com/questions/10682/suppress-in-biblatex)).

In **templates/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, unpublished]  
{title}{#1}  
%-- no punctuation after volume  
\DeclareFieldFormat[article]  
{volume}{{#1}}  
%-- puts number/issue between brackets  
\DeclareFieldFormat[article, inbook, incollection, inproceedings, misc, thesis, unpublished]  
{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}

# 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.

# References

Lottridge, Danielle, Eli Marschner, Ellen Wang, Maria Romanovsky, and Clifford Nass. 2012. “Browser design impacts multitasking.” In *Proceedings of the Human Factors and Ergonomics Society 56th Annual Meeting*. <https://doi.org/10.1177/1071181312561289>.

Shea, Nicholas, Annika Boldt, Dan Bang, Nick Yeung, Cecilia Heyes, and Chris D Frith. 2014. “Supra-personal cognitive control and metacognition.” *Trends in Cognitive Sciences* 18 (4): 186–93. <https://doi.org/10.1016/j.tics.2014.01.006>.

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

# (APPENDIX) Appendix

# 6 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**

library(tidyverse)  
knitr::include\_graphics("figures/chunk-parts.png")

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

knitr::include\_graphics("figures/beltcrest.png")

# 7 The Second Appendix, for Fun

1. my footnote text [↑](#footnote-ref-43)
2. This is a random test. [↑](#footnote-ref-44)
3. The bibliography can be in other formats as well, including EndNote (**.enl**) and RIS (**.ris**), see [rmarkdown.rstudio.com/authoring\_bibliographies\_and\_citations](https://rmarkdown.rstudio.com/authoring_bibliographies_and_citations.html). [↑](#footnote-ref-70)
4. For more on custom block types, see the relevant section in [*Authoring Books with R Markdown*](https://bookdown.org/yihui/bookdown/custom-blocks.html). [↑](#footnote-ref-94)
5. 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>. [↑](#footnote-ref-100)