Making Lecture Notes with Bookdown

Vicky Scowcroft

2024-01-18

Contents

Introduction

This guide walks through how to create html lecture notes using **R**, **Markdown** and **Bookdown**. This format is preferred as LaTeX doesn't play nicely with accessibility tools like screen readers, especially for maths heavy content.

System Requirements

To use bookdown, you will need to have R and R studio installed (perhaps R studio is not a requirement, but it makes things a lot easier). These can both be installed from the DDAT Self Service app on a Mac, or R can be downloaded directly from here and R Studio from here. You will also need **pandoc**. This may already be installed on your system as part of the Anaconda distribution.

The **bookdown** package can be installed from CRAN or Github:

```
install.packages("bookdown")
# or the development version
# devtools::install_github("rstudio/bookdown")
```

Caveat

The instructions in this guide are what worked for me, on a Mac using R version 4.2.1, R Studio version 2022.07.1 and pandoc version 2.18 (updated August 2022). This guide also assumes that you're familiar with using the command line.

Chapter 1

Getting started

The quickest way to get started is to use the bookdown demo as a template. This page gives instructions for how to do this.

The bookdown: Authoring Books and Technical Documents with R Markdown is the main documentation for bookdown. I won't repeat a lot of the information here.

This guide will walk through the steps I used to convert my LaTeX lecture notes for PH40112 to the bookdown version here. The files used to create the PH40112 notes are available on github and you're welcome to use those as a starting point.

Chapter 2

Converting from LaTeX notes

2.1 Converting tex files

If you already have LaTeX versions of your notes you can convert these to markdown using pandoc Pandoc won't create *perfect* versions of your notes - you'll most likely have to do a bit of tweaking, but it gets most of the way there.

You can convert a tex file to Rmd via the command

```
pandoc -f latex -t markdown input-tex.tex -o output-md.Rmd
```

where `input-tex.tex` is the name of your tex file and `output-md.Rmd` is the name of your output Rmd file.

Pandoc converts one tex file at a time. It doesn't understand how to process a "master" tex file with `input` or `insert` LaTeX commands. Each file will need to be processed separately.

2.2 Batch conversion

If you want to convert a batch of latex files you can script the process.

2.3 Tidying up the Rmd file

Pandoc will have done most of the work in getting your tex file converted to markdown. However, depending on how complex the original file was, you may end up with some tidying to do.

Things that typically will need fixing are:

- Section etc labels - if you used underscores in any of your latex labels you should change these to '-' in the markdown file. The underscores confuse markdown. Section 4 discusses sections, chapters, etc.
- Figures - the default pandoc conversion is quite limited, so here we'll use a more adaptable figure environment. In practice its easiest to just copy-paste the example code in Chapter 5 and edit the relavent bits.
- Equations - the equations themselves should be fine, but if you want to have equation numbers you'll need to do some tidying. See Chapter 6 for the syntax.
- Tables - tables have a different format but it's pretty easy to convert a latex table using find/replace. See Chapter **??**.
- References - generally easy to fix. See Chapter **??**.

2.4 Notes with gaps

Think carefully about whether you want to provide notes with gaps. While they may serve a purpose when the students have a hard-copy of the notes in a live lecture, they will most likely be using your notes on a screen this year.

It is possible to create notes with gaps for things like equations etc for the students to fill in. To do this you need to add some extra code to the `index.Rmd` file:

````
```
{r setup, include=FALSE}
library(knitr)
knit_engines$set(asis = function(options) {
  if (options$echo && options$eval) knit_child(text = options$code)
})
```
````

Then to "hide" e.g. an equation but still keep the same numbering, edit your equation to the following format:

````
\begin{equation}
```{asis, echo=FALSE}
\Delta s^2 = \Delta x_{1}^{2} + \Delta x_{2}^{2}
```
\label{eq:dist01}
\end{equation}
````

Here the `asis` command and back-ticks wrap around **only** the equation itself, rather than the whole `equation` object. If you wrap the `asis` around the `\begin{equation}` and `\end{equation}` the equation numbering will disappear too.

2.5 Using the `titlesec` package

If you're using the `titlesec` latex package you'll need to add an additional line to the yaml information on your index.Rmd file:

```
subparagraph: yes
```

This should fix any compilation errors related to the `titlesec` package when you're producing a pdf output.

Source: Stack Overflow

Chapter 3

A quick introduction to Markdown

3.1 What is Markdown?

> Markdown is a lightweight markup language that you can use to add formatting elements to plaintext text documents.
>
> markdown.org

You can create markdown documents in any text editor that can create plain text files. For making things in bookdown, I prefer to use the RStudio IDE so I have access to the build tools while I'm writing.

3.2 Markdown syntax

You may already be familiar with some/all of the markdown syntax from using things like Moodle. A nice markdown cheat sheet can be found here.

These are some of the features I use most frequently. In the following examples, the grey boxes with mono-spaced font show the markdown and the green boxes show the rendered output. The "\" you see in the raw markdown is an escape character to render the #, * etc. as symbols rather than a formatter.

3.2.1 Headings

You can make different levels of headings using different numbers of #'s at the start of a line

```
# One \#  for a chapter
## Two \#\# for a section
### Three \#\#\# for a subsection
```

Chapter 1 One # for a chapter

1.1 Two ## for a section

1.1.1 Three ### for a subsection

More details about chapter and section headings is in Section 4.[1]

3.2.2 Text formatting

You can make text *italic* using one * or _ at each end of the text. Bold uses two of each.

```
make italics using _one underscore_ or *one star* at each end
make bold using __two underscores__ or **two stars** at each end
```

make italics using *one underscore* or *one star* at each end

make bold using **two underscores** or **two stars** at each end

3.2.3 Lists

Making lists is easy too!

```
*   Bulleted lists
*   are made by putting
*   a \* (asterix)
-   or a dash (-)
-   at the start of the line.
-   You don't even need to be consistent.
    - You can make a sub-list by adding 4 spaces before the -
        - but I'm not sure how many levels this goes to...
```

- Bulleted lists
- are made by putting
- a * (asterix)
- or a dash (-)
- at the start of the line.
- You don't even need to be consistent.
  - You can make a sub-list by adding 4 spaces before the -
    - but I'm not sure how many levels this goes to...

[1] If you look at the source code for this section you'll see that the code to display the headings example doesn't match the markdown. This was a hacky way to get it to display correctly without messing up the section numbering If you find a solution for how to reset section numbers please let me know, because I've spent far too long on Stack Overflow today.

```
1. To make a numbered list
2. you just number things
3. i.e. 1., 2., at the start of a line
7. and it doesn't matter
4. if your numbers are in the right order
1. or if you keep using
1. the same number
```

1. To make a numbered list
2. you just number things
3. i.e. 1., 2., at the start of a line
4. and it doesn't matter
5. if your numbers are in the right order
6. or if you keep using
7. the same number

You can also do

```
a) Alphabetical lists
a) using lower case letters,
A) or using
A) upper case letters
```

a) Alphabetical lists
b) using lower case letters,

A) or using
B) upper case letters

or you can

```
1. mix all of them together
    a) if that's how
        - you like to spend
        - your time.
```

1. mix all of them together
    a) if that's how
        - you like to spend
        - your time.

Chapter 4

Chapters, sections, etc.

4.1 File Structure

Bookdown organises your notes into a book. The main page you see when you open a bookdown page is the `index.Rmd` file.

Each chapter in the book has it's own markdown file. There are two options for defining the order of your chapters: using numbers in the file name, or specifying the order in the `_bookdown.yml` file.

4.1.1 Specifying the order in `_bookdown.yml`

The easiest way to organise chapters is to specify the order in the `_bookdown.yml` file. For this method you don't need to have the numbers in the file name, and it's much easier to rearrange things. To specify the order, add an `rmd_files` section to the `_bookdown.yml` file and put the file names in the order you want them to appear:

```
rmd_files:
  - index.Rmd
  - 01-intro.Rmd
  - 02-latex-conversion.Rmd
  - intro-markdown.Rmd
  - 03-chapters.Rmd
  ...
  - 99-references.Rmd
```

You'll see here that the `intro-markdown.Rmd` chapter doesn't have a number at the start. You don't need them for this method; bookdown will ignore the numbers and render things in your listed order. More details (that you probably won't need) about how this works are here. **Your references chapter should be the last one, otherwise it doesn't work properly.**

4.1.2 Order by filename

This is the old method of ordering chapters. If you start from the bookdown template (described in Section 1), you'll see that the template files all start with a number. Bookdown will put the chapters in numerical order. For example, the following file names would order the chapters as they are in this book:

```
index.Rmd
01-intro.Rmd
02-latex-conversion.Rmd
03-intro-markdown.Rmd
04-chapters.Rmd
... etc
```

I find this works reasonably well if you know beforehand what order you're going to put things in, but it quickly gets annoying if you decide later on that you want to add a chapter between two existing ones (you'd have to rename all the subsequent chapters), so I recommend using the method in Section 4.1.1.

4.2 Section labels

The chapter title and label are given in the first line of the file.

```
# Chapters, sections, etc. {#sec:chapters}
```

The `#` at the start of the line denotes a chapter heading (see Section 3.2.1). The label is given in `{#sec:chapters}`. I think bookdown can create its own labels from the chapter/section heading, but it's easier to cross-reference if you set them yourself.

You can make sections using two # at the start of the line:

```
## Section labels {#sec:sec-lables}
```

Note that the `{#sec:label}` syntax is still the same - you don't need to say whether it's a chapter, section, subsection etc.

Subsections are done similarly, with three # at the start of the line.

You can change the section heading from "Chapter" to e.g. "Section" by editing the following in the `_bookdown.yml` file:

```
language:
  ui:
    chapter_name: "Section "
```

4.3 Cross referencing

You can reference other chapters, sections etc. throughout the book with the following syntax:

```
Section \@ref(sec:cross-ref) is this section.
```

Section 4.3 is this section.

When cross-referencing, you don't need the # before `sec`. That's only used to define the label.

Chapter 5

Figures

5.1 Adding figures

Figures and captions can be included, but the syntax is quite different to LaTeX.

Assuming that you would add a figure to a LaTeX document with the following code:

```
\begin{figure}
\begin{centering}
\includegraphics[width=0.7\textwidth]{Images/ho-tension.png}
\caption{This is a figure caption.}
\label{fig:ho-plot}
\end{centering}
\end{figure}
```

To include the figure in markdown the syntax is as follows:

````
```
{r echo=FALSE, ho-plot, out.width='70%', fig.show='hold', fig.cap="This is a figure caption."}
knitr::include_graphics("Images/ho-tension.png")
```
````

Important parts of this command:

- `echo=FALSE` prevents the code being used to display the figure being shown.
- `ho-plot` is your figure label.
- `out-width` is the output width of the figure.
- `fig.cap` is the figure caption.
- `knitr::include_graphics` gives the location of the image file.

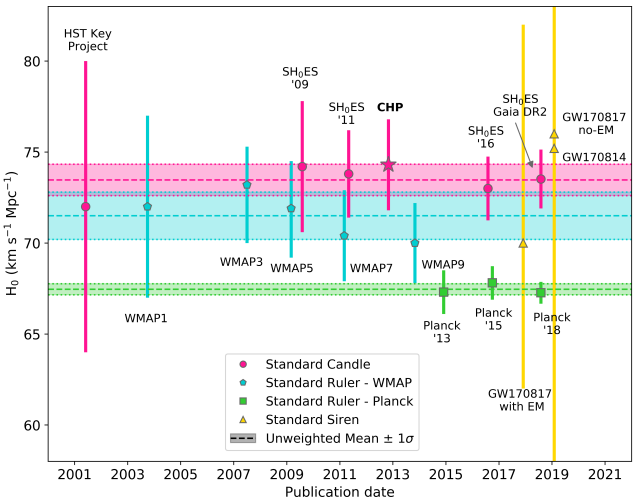

Figure 5.1: This is a figure caption.

5.2 Figure captions

Figure captions are controlled by the `fig.cap` setting. Your caption should be inside the double quotes. Remember to use \" to escape any " characters you may use inside your caption.

You can include cross-references and citations inside figure captions. See Sections 5.3, 6.3, and **??** for how to label and cross-reference figures, equations, and tables. Citations are covered in Section **??**.

5.3 Figure numbering and referencing

Figures with labels and captions are numbered according to the section (e.g. Fig 5.1 in this case).

To reference a figure use the syntax

```
\@ref(fig:label)
```

where `fig:label` would be `fig:ho-plot` in this case.

You can cross reference between sections/chapters, so make sure to use unique figure labels.

Chapter 6

Equations

6.1 Equation syntax

The syntax for equations is similar (but not identical) to LaTeX.

LaTeX code:

```
\begin{equation}
\label{eqn:friedman}
    \left(\dfrac{\dot{a}}{a}\right)^2 + \dfrac{kc^2}{a^2} = \dfrac{8\pi G}{3}\rho
\end{equation}
```

Rmd code:

```
\begin{equation}
    \left(\dfrac{\dot{a}}{a}\right)^2 + \dfrac{kc^2}{a^2} = \dfrac{8\pi G}{3}\rho
(\#eq:friedman)
\end{equation}
```

$$\left(\frac{\dot{a}}{a}\right)^2 + \frac{kc^2}{a^2} = \frac{8\pi G}{3}\rho \tag{6.1}$$

You can also use the latex

```
\begin{align}
...
\end{align}
```

format for equations. If you're converting from LaTeX to markdown with pandoc it may convert equations to

```
\begin{aligned}
...
\end{aligned}
```

which also works.

LaTeX subequations and intertext

I haven't been able to get subequations and intertext to work in bookdown. LaTeX equations of the form

```
\begin{subequations}\begin{align}
\vec{E} &= \left( x,t \right)
\intertext{and in 3 dimensional space as}
\vec{E} &= \left( x,y,z,t \right)
\end{align}\end{subequations}
```

should be written as separate equations with the text between written outside the equation environment, e.g.

```
\begin{align}
\vec{E} &= \left( x,t \right)
\end{align}
and in 3 dimensional space as
\begin{align}
\vec{E} &= \left( x,y,z,t \right)
\end{align}
```

6.2 Equation numbers and labels

The syntax for the maths is the same, but the labelling changes. To label and equation add

```
(\#eq:label)
```

just before `end{equation}`. Only equations with labels will be numbered. If you don't want numbers then don't label the equations, but numbers are helpful.

6.3 Cross referencing equations

The syntax for cross-referencing equations is similar to sections and figures, i.e.

```
Eqn. \@ref(eq:friedman) is the Friedman equation
```

will give "Eqn. (6.1) is the Friedman equation".