A GitBook Example for Teaching

dr. Caspar J. van Lissa

2026-02-19

Contents

Chapter 1

About GitBooks

A *GitBook* is a useful tool for creating (open?) educational resources. It is an online "book" format, that can be hosted directly from a GitHub repository. You are currently reading a GitBook designed to help you get started creating your own educational GitBooks for your courses (how meta!). It does this in two ways: By explaining how to create GitBooks, and by serving as a template that you can copy and edit, instead of having to start from scratch. This template GitBook has all the settings that I consider to be useful for educational GitBooks, but you can always customize it.

I will focus specifically on GitBooks that are made in Rstudio, using the

`rmarkdown` markup language, rendered using the `bookdown` package, and hosted on GitHub. If you want to get started, skip ahead to Chapter 2; if you need more convincing, keep reading below.

1.1   Why use a GitBook for teaching?

To spead the workload

My challenge was that I had to translate all tutorial instructions from proprietary software to R, and there was not enough time to complete this task before the course commenced. By making the tutorial instructions available in this GitBook, I was able to continue translating tutorial instructions *while the semester was ongoing*, and push updates to GitHub in time for each session, which were immediately available to all students. The parallel with the current situation is that some courses are now forced to start teaching in an online format, without having enough time to completely prepare. By using a GitBook, you can spread out the workload of preparing your materials across the semester. This is the finished GitBook

To contribute or use existing Open Educational Resources

Another key advantage of using a GitBook is, that you can easily make your course materials available for others to use under an open access license, or perhaps you can use an existing GitBook from the internet and adapt it for your own uses. GitBooks can be easily duplicated and adapted, just like any other project hosted on GitHub. Contributing Open Educational Materials can help reduce the workload on teachers around the world, and can improve the quality of the materials used thanks to online collaborating and feedback.

To benefit from formatting advantages

GitBooks also have two formatting advantages over classic PDF or Word files. First, they are Rmarkdown files, and can thus include blocks of R (or Python) code that can be evaluated, and whose results are rendered to the file. Second, they are interactive web pages, and as such, can have dynamic features (such as answers to assignments that can be hidden, or boxes where students can fill out an answer to be checked). Additionally, other web pages or interactive apps can be embedded within the page. So whereas a traditional document is static, GitBooks can be interactive.

How do GitBooks work?

GitBooks consist of an Rstudio project, with several Rmarkdown files containing the chapters of the book. Inside Rstudio, users can press a "Build Book" button, which renders all of these chapters to a nicely formatted HTML book (and a PDF file for users to download). Users can push the finished book to a GitHub repository, and indicate on GitHub that the book should be hosted on GitHub pages. Voilà!

Getting started

If you are convinced that this tool might benefit your teaching, your first point of action is to prepare your system for creating GitBooks (Chapter 2). After that, you can get a copy of this GitBook as a template (Chapter 3). Then, you can start tweaking it for your own course!

Chapter 2

Prerequisites

This GitBook is created in Rstudio, using the `bookdown` package. To get your system set up correctly, you have to install several software packages, and register on GitHub. You only have to perform these steps once, and the entire process should take approximately 1 hour if you start from scratch. In case some software is already installed on your system, you can skip related steps. Follow these steps in order:

1. Install R from cloud.r-project.org
2. Install Rstudio Desktop (Free) from rstudio.com
3. Install Git from git-scm.com. Use the default, recommended settings. It is especially important to leave these settings selected:
   - Git from the command line and also from third party software

   - Use the OpenSSL library

   - Checkout Windows-style, commit Unix-style line endings

   - Enable Git Credential Manager

   - If you run into any trouble, a more comprehensive tutorial on installing Git is available at happygitwithr.com.
4. Register on GitHub
   - Go to https://github.com/ and click "Sign up". Choose an "Individual", "Free" plan.
   - Request a free academic upgrade. This allows you to create *private repositories*, which are only visible to you and selected collaborators, and can be made public when your work is published.
5. Connect Rstudio to Git and Github (for more support, see this Rstudio article, and this blog post)
   a. Open Rstudio, open the Tools menu, click *Global Options*, and click

Git/SVN

b. Verify that *Enable version control interface for RStudio projects* is selected

c. Verify that *Git executable:* shows the location of git.exe. If it is missing, manually fix the location of the file.

d. Click *Create RSA Key.* Do not enter a passphrase. Press *Create.* A window with some information will open, which you can close.

e. Click *View public key*, and copy the entire text to the clipboard.

f. Close Rstudio (it might offer to restart by itself; this is fine)

g. Go to https://github.com

h. Click your user icon, click *Settings*, and then select the *SSH and GPG keys* tab.

i. Click *New SSH key.* Give it an arbitrary name (e.g., your computer ID), and paste the public key from your clipboard into the box labeled "*Key*".

j. Open Rstudio again (unless it restarted by itself)

6. Install all required packages by running the following code in the Rstudio console. Be prepared for three contingencies:

   - If you receive any error saying *There is no package called XYZ*, then run the code `install.packages("XYZ")`

   - If you are prompted to update packages, just press `3: None`. Updating packages this way in an interactive session sometimes leads to errors, if the packages are loaded.

   - If you see a pop-up dialog asking *Do you want to install from sources the package which needs compilation?*, click *No.* If this leads to errors, then please follow *Step 3* from this online guide, and run `install.packages("devtools")`. This will take a long time, but will allow you to install packages from source.

Run the following code to install the required packages:

```
install.packages("bookdown")
install.packages("tinytex")
tinytex::install_tinytex()
git2r::config(global = TRUE, user.name = "your.name", user.email = "your.email")
```
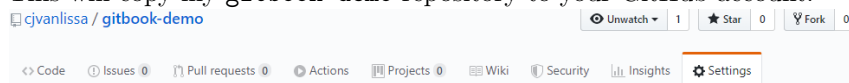
That's it! Everything should be installed and connected now.

Chapter 3

Get your GitBook

To get your GitBook, you should follow these steps:

1. Go to https://github.com/cjvanlissa/gitbook-demo
2. In the top right of the page, click `Fork`.
   This will copy my `gitbook-demo` repository to your GitHub account.

3. My repository is now copied to your account. It is a template repository, which means that you can create a *new repository* based on this one.
4. Create a new repository for your own GitBook. Create one for a course you've been wanting to update. In the top-right corner of the GitHub website, click the + icon, and select "New repository":
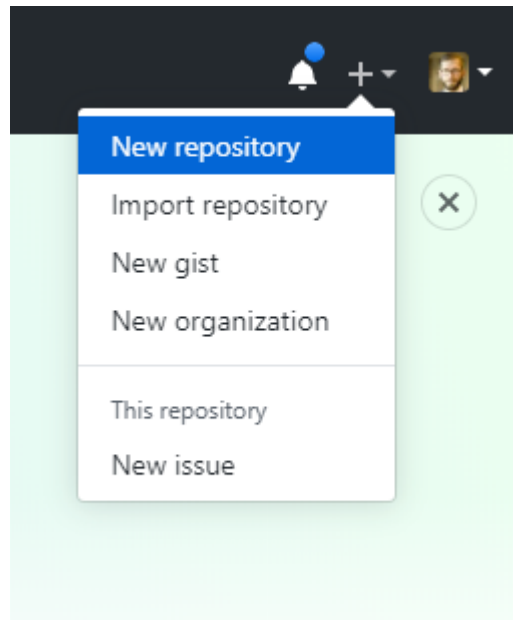
5. In the dialog, select the `gitbook-demo` as "Repository template", and give the repository an appropriate name for your course. Then, press `Create repository`:

6. Now, go back to Rstudio on your computer. In Rstudio, click `File > New Project`. A dialog will open. If you set up Rstudio with Git correctly, the dialog should have an option to create a new project from Version control. Click it:
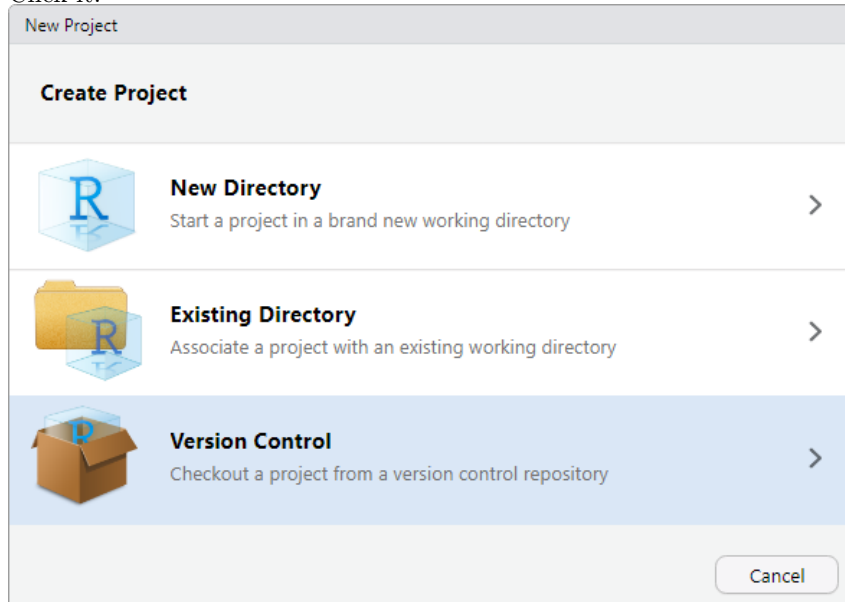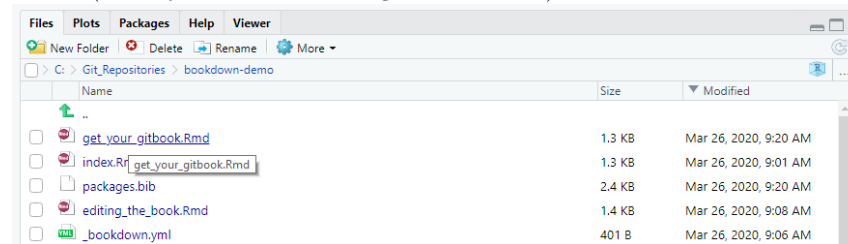

7. In the next dialog window, you should copy the URL of the GitHub repository you created in *Step 5*, like so:
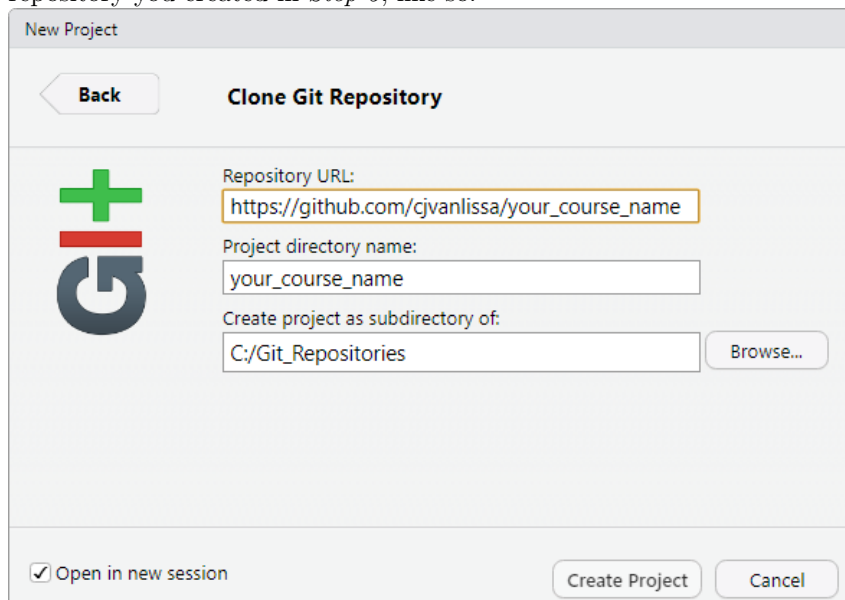

8. Now, in Rstudio, you can open files for editing and create new files (ex-
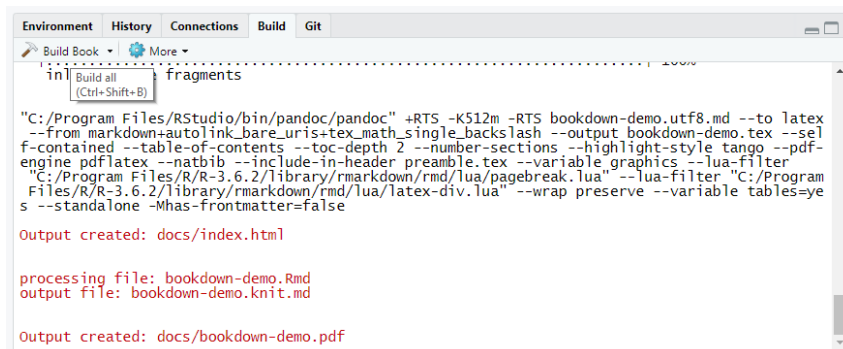
plained in the next Chapter). Open files by clicking them in the Files editor (usually in the bottom right of Rstudio):

9. After you make a change, it will show up in the Git tab (usually in the top right of Rstudio). You must Commit the change locally, and then Push the change to GitHub to update your repo. To Commit, select the file and click the Commit button. Write a short message to describe the changes you made, then click the Commit button again. Now, press Push to send your commits to GitHub.
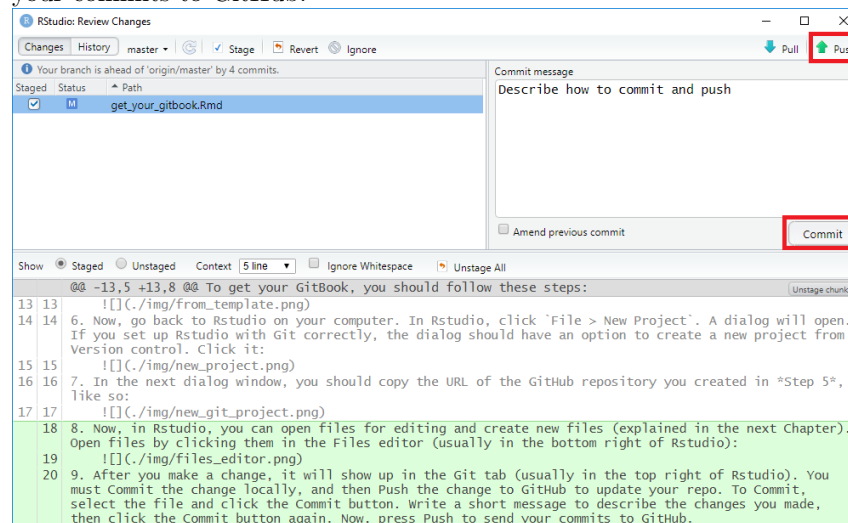
10. To render your book as a GitBook, you must "Build" it. In the top-right panel of Rstudio, you see a "Build" tab. In this tab, simply click the "Build Book" button to build your book. You should see a lot of rendering messages, until a window pops up with your brand new GitBook. If you get errors at this stage, you probably made a mistake in preparing your system (see the previous Chapter).

11. Building the book generated a lot of new files in the `./docs` directory. This directory contains the website files for your GitBook. Open the Git tab again, verify that the `./docs` directory is listed, and Commit and Push all of these new files as described in *Step 9*.

12. There is only one last remaining task: To publish your GitBook on GitHub pages. Once you do this, any change to the `./docs` folder that you push to GitHub will lead to an immediate update of your GitBook website. Go back to the GitHub page for your Repository. Click on the `Settings` tab on the top right of the page:
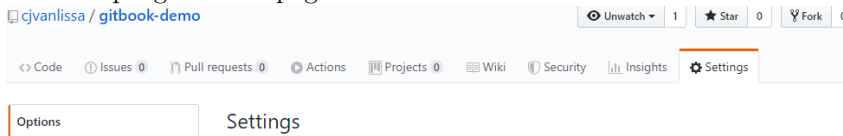
13. On the Settings page, scroll all the way down until you reach a section called `GitHub Pages`. There, under the "Source" heading, click the word `None`, and select `master branch /docs folder`. When you select it, the page will update, and if you scroll back down to the `GitHub Pages` section, you will see the URL where your GitBook is published. The first time, it will take a few minutes for your GitBook to come online. When you publish updates to the GitBook however (simply by following *Step 11* again), the update will be near-instantaneous. The Pages section should now look like this (and that is hopefully the link where you found this book):
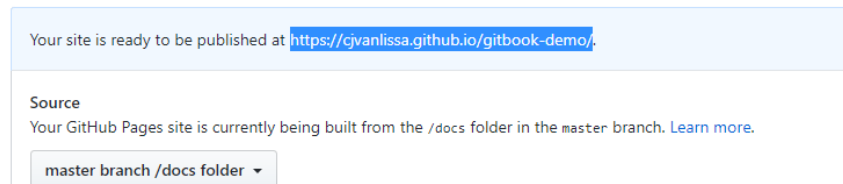
Chapter 4

Editing the book

The contents of the book are written in **RMarkdown**. You can use any formatting code that Pandoc's Markdown supports, e.g., a math equation $a^2 + b^2 = c^2$. Moreover, you can include chunks of R-code, like this:

The results of these chunks can be rendered to the GitBook:

```
## [1] "This is an R-command!"
```

To edit the book, you can change the text in the `.Rmd` files. Each Rmd file should contain **one and only one** chapter. A chapter is defined by the first-level heading `#`, e.g., `# Editing the book`.

Any sub-headings within the chapter are indicated with several `#` signs, e.g., `##` (level 2) and `###` (level 3).

4.1 Creating new chapters

To create a new chapter, you must follow two steps: 1) Create the file, and 2) Include it in the list of chapters.

First, to create the file for a new chapter in Rstudio, click `File > New File > Text file`. At the top of the file, write your chapter heading, as explained above. Then, click `File > Save`. Save the file as `.Rmd`, without spaces in the file name, e.g.: `editing_the_book.Rmd`.

Second, to include it in the list of chapters, open the file `_bookdown.yml` (click it in the Files explorer in the bottom right of Rstudio). This file has a list of `.Rmd` files to be included in the book. In this example, the list looks like this:

```r
tmp <- readLines("_bookdown.yml")
cat(tmp[grep("^rmd_files", tmp):grep("references\\.Rmd", tmp)], sep = "\n")
```

rmd_files: ["index.Rmd", "prerequisites.Rmd", "get_your_gitbook.Rmd", "editing_the_book.Rmd", "figures_tables.Rmd", "examples.Rmd", "open_educational.Rmd", "use_in_course.Rmd", "licenses.Rmd", "references.Rmd"]

Insert the file name of your new chapter in the desired position in this list.

4.2 Linking across chapters

You can label chapter and section titles using `{#label}` after them. The labels can be used as cross-references. For example, we can link to Chapter 5. If you do not manually label chapters, there will be automatic labels anyway, e.g., Chapter 6.

4.3 Advanced editing

The convenient Rmarkdown Cheat Sheet by Rstudio covers most of the knowledge required for advanced Rmarkdown editing. You can print it out and stick it to your wall!

Chapter 5

Figures and tables

Figures and tables with captions will be placed in `figure` and `table` environments, respectively.

```r
par(mar = c(4, 4, .1, .1))
plot(pressure, type = 'b', pch = 19)
```
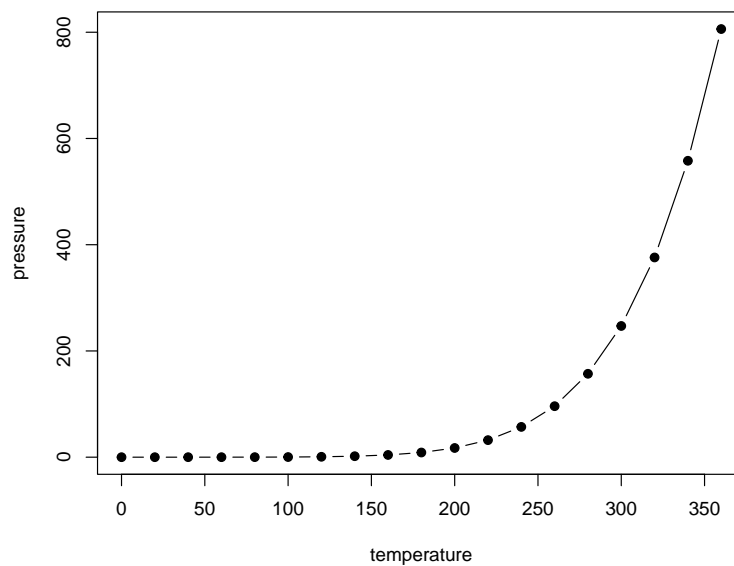

Figure 5.1: Here is a nice figure!

Reference a figure by its code chunk label with the `fig:` prefix, e.g., see Figure 5.1. Similarly, you can reference tables generated from `knitr::kable()`, e.g., see Table 5.1.

Table 5.1: Here is a nice table!

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa

```r
knitr::kable(
  head(iris, 20), caption = 'Here is a nice table!',
  booktabs = TRUE
)
```

You can write citations, too. For example, we are using the **bookdown** package (Xie, 2025) in this sample book, which was built on top of R Markdown and **knitr** (Xie, 2015).

Chapter 6

Examples

Here are some examples of other GitBooks for courses; if you want to have your GitBook added to the list, please send a Pull Request (here's how to send a pull request).

6.1 Statistics with R (H. Quene)

https://hugoquene.github.io/emlar2020

A GitBook for a tutorial on *Statistics with R (Basics)*, held as part of the workshop on Experimental Methods in Language Acquisition Research (EMLAR, https://emlar.wp.hum.uu.nl/), Utrecht, on 17 April 2020. This compact introduction helps you with your first steps into R.

6.2 Theory Construction and Statistical Modeling (C. J. van Lissa)

http://cjvanlissa.github.io/TCSM

A GitBook for the course *"Theory Construction and Statistical Modeling"*, with some interesting code, for example: Blocks of answers to the tutorial questions that can be collapsed and expanded.

6.3 Doing Meta-Analysis in R (C. J. van Lissa)

http://cjvanlissa.github.io/Doing-Meta-Analysis-in-R

A GitBook on doing meta-analysis in R, based on the book 'Doing Meta-Analysis in R', by Mathias Harrer, Pim Cuijpers, & David Ebert, and adapted to focus on the metafor package, and exploring heterogeneity using metaforest. The original

can be found here: https://bookdown.org/MathiasHarrer/Doing_Meta_Analysis_in_R/

6.4   Métodos quantitativos em Psicologia com R (L. Anunciação)

https://anovabr.github.io/mqt/

This book provides a short and to-the-point exposition on the essentials of statistics, and was written for undergraduate students at the Pontifical Catholic University of Rio de Janeiro (PUC-Rio). To a lesser degree, the mathematical modeling of statistical questions will be addressed. This book might be relevant for Portuguese-speaking students who enroll for laboratory-based statistics and anyone who wants to learn R.

Chapter 7

Open Educational Resources

UNESCO defines Open Educational Resources as *teaching, learning and research materials in any medium – digital or otherwise – that reside in the public domain or have been released under an open license that permits no-cost access, use, adaptation and redistribution by others with no or limited restrictions.*

Open Educational resources can help lighten the workload on individual teachers, who can collaborate with the development of high-quality open access resources, instead of having to develop their own proprietary materials from scratch. Moreover, Open Educational resources are inclusive, lowering the barrier to knowledge acquisition for learners around the world, and enabling lifelong learning for those outside academia.

Many universities support Open Educational Resources. Here are just a few (feel free to send a pull request with other relevant resources).

- **OER Commons**: A freely accessible online library of open educational resources.
- **Utrecht University Figshare**: Open learning objects from Utrecht University.
- **Johns Hopkins University OCW**: Open public health courses and materials.
- **University of Pittsburgh OER**: Big List of Open Educational Resources.
- **MERLOT**: Online learning and support materials and content creation tools, led by an international community of educators, learners and researchers.

Chapter 8

Compatibility with existing systems

Many universities offer digital platforms for learning. You might wish to embed your GitBook within these existing systems. Here are two ways in which you might do that. Currently, this section only discusses BlackBoard, but the same principles should apply to other platforms.

8.1 Add a hyperlink

You can add a link to your GitBook in the BlackBoard course menu by following this tutorial.

8.2 Embed the whole book

You can add a Blank Page to your BlackBoard course menu, and fill that page with a full-size "iframe" - a web page within the web page. This tutorial explains how to do it. It is possible that your university is blocking this feature, however.

Chapter 9

License your GitBook

In the spirit of Open Science, it is good to think about making your course materials Open Source. That means that other people can use them. In principle, if you publish materials online without license information, you hold the copyright to those materials. If you want them to be Open Source, you must include a license. It is not always obvious what license to choose.

The Creative Commons licenses are typically suitable for course materials. This GitBook, for example, is licensed under CC-BY 4.0. That means you can use and remix it as you like, but you must credit the original source.

If your project is more focused on software or source code, consider using the GNU GPL v3 license instead.

You can find more information about the Creative Commons Licenses here. Specific licenses that might be useful are:

- CC0 ("No Rights Reserved"), everybody can do what they want with your work.
- CC-BY 4.0 ("Attribution"), everybody can do what they want with your work, but they must credit you. Note that this license may not be suitable for software or source code!

For compatibility between CC and GNU licenses, see this FAQ.

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr.* Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2025). *bookdown: Authoring Books and Technical Documents with R Markdown.* R package version 0.46.