A new LSP Book

Firstname Lastname

April 20, 2023

Contents

Chapter 1

In the beginning

1.1 Markdown philosophy

[1.1] This directory is a skeleton to write a monograph for *Language Science Press* (LSP) in Markdown. MARKDOWN was introduced by John Gruber as a easy-to-use method to write content for webpages. The principles have spread widely and are formally codified as COMMONMARK. The basic idea is to make it easy to write content, while the details of the formatting are added by an automatic conversion.

[1.2] This possibilities of this automatic conversion has been greatly extended by PANDOC as introduced and maintained by John MacFarlane. Pandoc can convert between dozens of different output formats, allowing for a great freedom for the visual display of your text. Pandoc also offers many extensions to the rather basic possibilities of the original markdown/commonmark proposals. Pandoc also has a robust system to add functionality through additional modules (called 'filters') when needed.

[1.3] Although Pandoc offers a wide range of bidirectional conversions between all kinds of formats, the current system for LSP suggests that you write in Pandoc-enhanced Markdown and convert from that basis to any desired output format. The problem is that every format (e.g. LaTeX, docx, odf) has its own design-quircks, and Pandoc will never be able to convert every detail between all formats. The Pandoc-Markdown structure offers a convenient set of markup options that are garantueed to be converted well and are sufficient for scientific texts.

[1.4] Any writer's desire that is currently missing in Pandoc-Markdown can be easily added by FILTERS. In essence, Pandoc filters are very similar to packages in LaTeX. Because Pandoc is still relatively new, there are not as many filters available as there are LaTeX-packages, and there is also not yet a central repository for filters like CTAN for LaTeX. However, filters (especially Lua-based filters) are really easy to write and adapt, especially when compared to the rather arcane format of LaTeX-packages. The current LSPmarkdown skeleton includes various special filters to prepare a scientific book that is published on the open web. Some more details about the rationale for these filters is explained here.

[1.5] There are some major benifits when using Pandoc-markdown:

- the raw text that you will write is very simple structured and easy to read in its raw form, especially when compared to LaTeX.
- markdown strictly follows the design philosophy of separating presentation from content. You write the content, while leaving the presentation to the output format by

an automatic conversion.
- the raw markdown can be converted into many different output formats using Pandoc, e.g. Latex, PDF, HTML, docx, etc. Because the output is generated automatically, the styling and formatting is always consistent.

There are various limitations to the current markdown setup: [1.6]

- If you depend on specific functionality provided by LaTeX-packages (e.g. producing syntactic trees), then you can still use them within markdown. However, such LaTeX-specific insertions can only be conversted to LaTeX and will not be converted to other formats, like HTML or docx.
- You are currently restricted to the possibilities as described in Section 2. Anything outside of those bounds have to be added by tailor-made Pandoc filters. Suggestions and request for new filters are welcome.

Note that there are various other approaches that are similary to the current LSPmarkdown skeleton. Most prominently there is Rmarkdown and the related Bookdown, which are both based on `knitr` by Yihui Xie. Those approaches embed Pandoc inside R-packages and thereby offer many nice options for the visualisation of quantitative data. However, these approaches are strongly geared towards usage within RStudio. [1.7]

1.2 Installation

For writing a book with the LSPmarkdown skeleton you can use any text editor of you choice. However, prefereably stay clear of full-fledged word-processors like Microsoft Word or OpenOffice because they tend to automatically add all kind of markup in the background, often onbenowst to the user. [1.8]

Currently, the free editor VISUAL STUDIO CODE is in very active development and probably the best choice if you are not yet entangled to any other text editor. However, any text editor is fine, e.g. TextMate, BBedit, Sublime Text, Atom, or even Emacs/vim if you are so inclined. [1.9]

In all modern editors (like Visual Studio Code) it is possible to open a complete directory/folder, so it becomes easy to switch between editing the different files in the LSPmarkdown directory. The current LSPmarkdown directory should be the starting point of your book. Simply rename the directory to your liking and change the current content (which is only included as an example). [1.10]

If you want to convert your text to any of the LSP-based outputs you will need to install some additional software. This can be done through a package manager like `Homebrew` for macOS. However, it might be easiest for new users to simple install the following three pieces of software separately: [1.11]

- **Pandoc**, install instructions at https://pandoc.org/installing.html
- **pandoc-crossref**, install instructions at https://github.com/lierdakil/pandoc-crossref. This is a Pandoc-filter for cross-referencing that you will have to install separately, because depending on your computer you will have to install a different version.
- **LaTeX**, install instructions at https://www.latex-project.org/get/. LaTeX is needed to produce PDF output as used at the *Language Science Press.* Included in this skeleton is also an option for a quick PDF creation that can be used for the preparation of drafts in PDF form.
- **Libertinus font** from https://github.com/alerque/libertinus/releases. This font is used to produce output in the format of the *Language Science Press.* The HTML file will attempt to download this font by itself when it is not installed on your computer.

1.3 Setup

[1.12] To prepare your book there are three basic files with settings that you will have to adapt to your needs:

- the file `1_TITLEPAGE.yaml` contains the basic metadata that will end up on your title page
- the file `2_CONTENTS.yaml` contains the list of the Markdown files that make up your book. Simply list the filenames in the order as they should occur in the book here.
- the file `3_SETTINGS.yaml` contains a few further user-settings. Specifically, you will have to specify the location of your bibliography here (see also Section 2.7).

[1.13] The current LSPmarkdown directory is prepared for direct upload as a git-repository, e.g. at Github or Gitlab. That is probably the easiest way to share your work with others, also in any pre-publication status. There are a few files included here for a more transparent sharing of your work online:

- a `LICENSE` file with a CC-BY license in accordance to the LSP guidelines.
- a `README.md` file. Please write a few lines in this `README` file to introduce your book.
- a `NEWS.md` file. This file can initally be ignored because it is geared to documenting updates for possible subsequent editions of the book.

Chapter 2

Writing in Pandoc-markdown

2.1 Basic editing

[2.1] The basic principles of writing in Markdown are explained at various places online. However, the most authorative summary is provided by the COMMONMARK project. The most basic rules are:

- use hashes (#) at the start of a line for headings. The number of hashes indicates the heading level.
- use stars (*) around text that should be set in *italics*.
- square brackets `[]` are used for all kind of links and cross-references (see below).

2.2 Pandoc advanced editing

[2.2] The possibilities of markdown are enhanced by Pandoc with various crucial formatting options as specified in the PANDOC USER'S GUIDE. These formatting options will all be retained in the various conversions from Markdown into other formats (e.g. HTML or PDF). For example, footnotes are included by using the following format inside your text at the position where the footnote mark should occur.[1]

```
^[Footnote text between square brackets and a circumflex symbol before the brackets.]
```

[2.3] In the current LSPmarkdown framework there is an additional filter added that will change strikethrough formatting (by enclosing double tildes) into SMALL CAPS formatting. This is a convenience option because linguistic texts often use small caps, but only rarely strikethrough. To remove this automatic conversion, simply remove the filter `strikeout-to-smallcaps` from the conversion files (e.g. from the file `tohtml.yaml`).

2.3 Cross-referencing

[2.4] Pandoc itself contains some basic cross-referencing options. However, it is strongly recommended to install the extra software PANDOC-CROSSREF. This allows for various flexible

[1]This is an example footnote. Footnote text between square brackets and a circumflex symbol before the brackets.

options to automatically insert internal cross-references in your text. Basically, you add a label to a heading by adding it behind the heading like this:

```
## Some Heading {#mylabel}
```

In your text you then refer to this heading by typing `[@mylabel]` which will result in a cross reference like this: Section 2. PANDOC-CROSSREF has many more possibilities and options as explained in detail in the user's guide. [2.5]

2.4 Linguistic examples

To add linguistic examples, the LSPmarkdown skeleton includes a Pandoc-filter PANDOC-LING. A full user's guide describing all options and limitations is available. Basically, you can add linguistic examples using the following format: [2.6]

```
::: ex
a. This is an example sentence.
b. And another one.
:::
```

This will result in a numbered example as shown below. You can refer to this example using abbreviations like `[@next]` before the examples or `[@last]` after the example, e.g. see example (2.1). [2.7]

(2.1) a. This is an example sentence.
b. And another one.

2.5 Figures

To insert figures into your text, prepare figures separately and store them in the directory `figures`. In your markdown then add a line like the following below the table. This will be automatically numbered, and you can use the code `[@fig:crossreferencelabel]` in your text to refer to the figure, for example Figure 2.1. [2.8]

```
![Caption text here](figures/myfigure){#fig:crossreferencelabel}
```

To prepare HTML output it is prefereable to convert any figure into SVG format. There are many online converters that can do this. You can include different file formats with the same filename into the directory `figures`, e.g. `myfigure.pdf` and `myfigure.svg`. [2.9]

2.6 Tables

The situation to include tables is not yet very user friendly in Pandoc-Markdown. The editing and conversion of tables is currently a field of active development within Pandoc, so this will probably be improved substantially in the near future. For now, check out the possibilities for formatting tables in the Pandoc user's guide. If you use Visual Studio Code, then there is a useful extension that might be helpful for formatting tables called TABLE FORMATTER. [2.10]

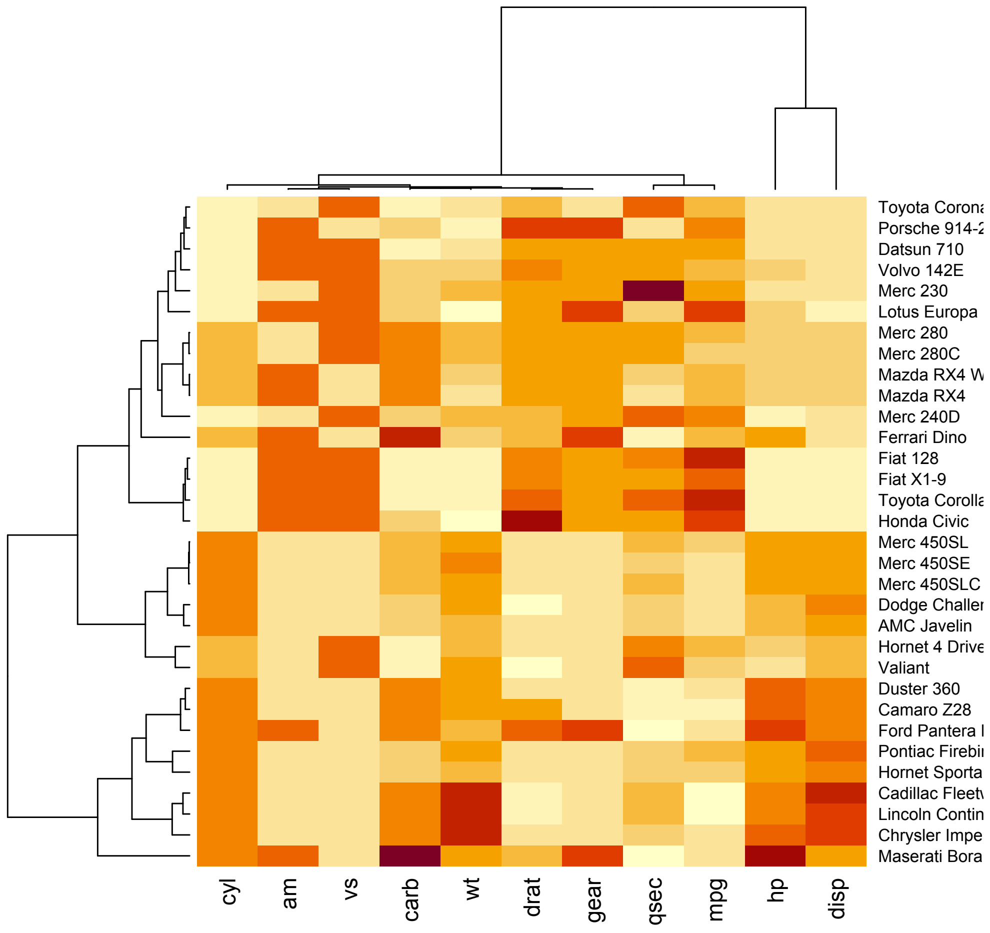

Figure 2.1: This is an example figure.

Table 2.1: This is an example table

Right	Left	Default	Center
12	12	12	12
123	123	123	123
1	1	1	1

[2.11] To add a caption to the table, add a line like the following below the table. This will be automatically numbered, and you can use the code `[@tbl:crossreferencelabel]` in your text to refer to the table, for example see Table 2.1.

```
Table: Caption text here {#tbl:crossreferencelabel}
```

2.7 References

Pandoc includes a CITEPROC extension to treat references and bibliography using the CITATION STYLE LANGUAGE (CSL) framework. The current framework uses the 'unified style sheet for linguistics' by default. There are various ways to include references, but two options seem to be most useful: [2.12]

- if you manage your references in BIBTEX style, then you only have to add a path to your BibTeX-file in `3_SETTINGS.yaml`
- if you manage your references with Zotero then you should additionally install BetterBibTeX.

For the Zotero/BetterBibTeX workflow, you will have to additionally do the following: [2.13]

- create an synchronised BibTeX file in Zotero by using `File >> Export Library…` checking `keep updated`. Save this file somewhere where you can easily find it on your computer. The simplest solution would be save this file inside the LSPmarkdown directory.
- add a path to this file in `3_SETTINGS.yaml`.
- set `Preferences >> Better BibTeX >> Automatic export >> Automatic export: On change`.

In both options your references will have a 'citation key' which typically uses a format like `chomsky1957` (but this format can be changed to your liking). To include a reference in your text use the link as shown below. This will result in a reference in your text (Chomsky 1957: 23). You can add anything you like after the citations key, typically page numbers. To suppress the name inside the brackets, add a dash-minus symbol before the '@' symbol. This will result in a reference to Bloomfield (1925). [2.14]

```
[@Chomsky1957: 23]
[-@Bloomfield1925]
```

Chapter 3

Making the book

[3.1] A few different settings-files are included in this LSPmarkdown skeleton to prepare the final book from your markdown source. These are `yaml`-files with various settings. Such files are called `defaults-files` in pandoc-parlance (see the pandoc documentation).

- the file `tohtml.yaml` is used to convert to LSP-styled HTML
- the file `topdf.yaml` is used to convert to a quick-and-easy draft PDF
- the file `totex.yaml` is used to convert to LSP-styled LaTeX and subsequent LSP-styled PDF

[3.2] To use these default-files you will have to open a terminal/shell in the current directory. If you use Visual Studio Code and you have opened the whole LSPmarkdown directory with `File >> Open Folder…` then this is really easy, also when you have never used a terminal/shell. Simply open a terminal/shell through the menu `Terminal >> New Terminal` and then type the command as specified below.

3.1 LSP-style HTML

[3.3] To convert your markdown to HTML type the following command in your terminal and hit return:

```
pandoc -d tohtml.yaml
```

[3.4] As a result there will be a new file called `index.html` in the directory `docs` with the final HTML output. This file is completely self-contained and does not need any other files to work properly. You can immediately open this file locally from you computer by double-clicking it. It will open locally in you default web browser.

[3.5] Because this file is completely self-contained you can easily share it with other people (just send them the file). When you sync your whole directory with GitHub you can immediately publish this file using `GitHub Pages`. Note that the fonts used in the LSP style are Libertinus Serif (for the main text) and Arimo (for the title page). When these fonts are not installed on your computer the browser will attempt to fetch them from the internet.

[3.6] For final LSP-publication there are few additional minor tweaks to be made:

- The copyright info has to be added at the start of the HTML file. This information has to be manually changed in the file `settings/LSP-copyright.md`. To include

this file in the final output open the defaults-file `tohtml.yaml` and add the line as explained there.
- You might want to remove the coloured hypelinks by deleting the respective lines at the bottom of the settings-file `3_SETTINGS.yaml`.
- To add colour to the title page, you will have the change to colour in the file `settings/LSP-css.yaml` in the line `background-color`.

3.2 Draft PDF

To convert your markdown to PDF you can use the following command in the terminal: [3.7]

```
pandoc -d topdf.yaml
```

This will produce a PDF-file called `book.pdf` in the directory `docs`. This PDF does not use the LSP styling. Instead, the PDF will use the default LaTeX-style from the conversion-software Pandoc. The advantage of this option is that this conversion to PDF is much quicker and easier than the complete LSP pathway as described below. This is particularly useful if you want to produce a quick PDF version for printing or reviewing of your text. Note that you will need LaTeX to be installed on your computer, see Section 1.2 [3.8]

3.3 LSP-style PDF

Finally, there is a conversion option to prepare your book for LaTex-based LSP publication. The preparation of a final book for LSP is somewhat more involved because there are various checks and additional fine-tuning needed for a polished real-life publication. Basically the procedure is as follows: first, convert your markdown into raw LaTex, and second, use this LaTex to proceed through the regular LSP pipeline. To convert you markdown to raw LaTeX you can use the following command in your terminal: [3.9]

```
pandoc -d totex.yaml
```

This will produce a tex-file called `all.tex` in the directory `latex/chapters`. The directory `latex` is a slightly adapted version of the default LSP skeleton to produce a book. You will have to make a few more changes in this directory to produce a complete LSP book: [3.10]

- add author and title information in the file `latex/localmetadata.tex`.
- add any figures (preferably in PDF format) into the directory `latex/figures`

Then you can produce a draft version of the final LSP styled PDF by typing the following command in your terminal: [3.11]

```
make -C latex
```

This will take some time, and might very well spit out many TeX-errors. For now simply ignore these messages. If this process gets stuck, type “x” to break and ask somebody for help. If it finishes, then there will a PDF-file called `main.pdf` in the directory `latex` with your LSP-styled book. The final tweaks to this process will be performed together with the people from the *Language Science Press*. [3.12]

Bibliography

Bloomfield, Leonard. 1925. On the sound-system of central Algonquian. *Language* 1(4). 130–156.

Chomsky, Noam. 1957. *Syntactic structures.* The Hague: Mouton.