

LaTeX Crash Course

Meeting 2: Figures, Tables & Co.

Jan Sprenger

July 2, 2021

In the first meeting of the crash course, we saw the basics: how a LaTeX document is structured, how you write and compile it, how you highlight text, how to format paragraphs, and how you add lists, footnotes, quotations, etc. These are the essential things you need for writing a LaTeX document, especially as a philosopher.

The second meeting deepens your understanding of these basic features of LaTeX and adds other important things:

- some (subjective) recommendations for packages to call
- typing formulas (mathematics, logic)
- embedding figures in your LaTeX documents
- creating tables and diagrams

Typesetting mathematical content is explained in a separate handout.

1 General Stuff

Essential Internet Resources

Two useful resources:

- A wiki book: <https://en.wikibooks.org/wiki/LaTeX/>
- Overleaf Tutorials: <https://www.overleaf.com/learn/latex/Tutorials>

I use the opportunity to say a couple of things about Overleaf. The advantage is that it is quite simple to use and you don't have to install anything on your computer. However, there are substantial drawbacks:

- the usual issues with commercial service providers, changes of terms of service, data security, etc.;
- the editor does not have the convenient shortcuts that editors such as LaTeXila, TeXstudio, etc., provide;
- you need to upload the (separate) bibliography file every time you add a relevant reference to your local `.bib` file (see Meeting 3);
- no control over the compiling process.

For these reasons, I recommend to work with a local LaTeX distribution on your computer.

Useful Packages

Here is a list of packages I usually include in the preamble of my LaTeX document, roughly sorted from essential to strictly optional.

inputenc By default, LaTeX reads only the 256 ASCII characters. The **inputenc** package makes it possible to read other characters that you write in your `.tex` file (e.g., letters with diacritics). I recommend to call it with the unicode option `utf8`, i.e., `\usepackage[utf8]{inputenc}`.

fontenc Output-oriented counterpart of **inputenc**. Chooses a font encoding that supports accented characters. This is important because otherwise TeX would not correctly hyphenate words containing accented characters. I usually call it with the option `T1`.

babel For language support—use the document language (e.g., `english`, `italian`, `piedmontese`) as an option. You can call several options and separate them by commas.

biblatex/natbib For the bibliography—will be explained in the next meeting.

enumitem For customizing lists. One of the best packages, a real miracle. Check out the documentation.

amsmath,amssymb,amsthm For typing formulas, equations, defining theorem environments, and so on.

graphicx For including figures—see below.

url,hyperref For typing URL's (`url`) and inserting hyperlinks (`hyperref`).

geometry For customizing the page format, especially the margins.

fancyhdr For customizing the header and footer (e.g., if you want the chapter number and title in the header).

placeins This is helpful for controlling the placement of floats (figures, tables, diagrams)—see below.

xcolors For calling colors by their name whenever you need them. So you don't have to insert the numeric red/blue/green code by hand.

That said, for each specific thing you want to do in LaTeX there is a specific package. Logicians and philosophers of language will appreciate the **semantic** package, people with special typesetting needs the **setspace** or **nowidow** package, people who want to write in two or more columns (e.g., journalists) the **multicol** package, and so on. For choosing a specific font, you need to call the corresponding package—more on this below.

There is a huge lot of documentation on each package online. Just search on the internet. Same if you have a specific need and you don't know which package resolves it: just search (the keywords of) your question and you will find that somebody has asked the same question on tex.stackexchange.com or some other website. And you will find several answers to the question, sorted according to relevance, usually starting with the sentence “you need to call the package xyz”.

2 Specific Stuff

Choice of Fonts

LaTeX offers you a huge lot of fonts that you can choose. Usually you just set your preferred font in the preamble, by calling the appropriate package, e.g.:

- `\usepackage{courier}` (Courier)
- `\usepackage{helvet}` (Helvetica)
- `\usepackage{lmodern}` (Latin Modern)
- `\usepackage{mathptmx}` (Times)

The default LaTeX font is Computer Modern, by the way. You find the full catalogue of LaTeX fonts [here](#). The Overleaf tutorial is also useful. Geeks will find very detailed information about font encoding and other attributes [here](#).

Numbered Lists

The **enumitem** package has, as I mentioned above, some very nice features for customizing lists. I like in particular that you can interrupt and resume

numbered lists. Suppose you have an `enumerate` environment followed by normal text. Then you can continue the count with the `resume` option, i.e., `\begin{enumerate}[resume]`.

Figures

Figures are introduced with the `\begin{figure}` command and concluded with `\end{figure}`. This opens a `figure` environment. Then you add the figure with `\includegraphics{name.png}`. Below this you add the caption (e.g., `\caption{This is a nice figure.}`) and the label of the figure (e.g., `\label{fig:123}`). You can then refer to it later with `\ref{fig:123}` or `\pageref{fig:123}`.

For controlling the width of a figure, you can add the `[width=\textwidth]` option to the `\includegraphics` command (before calling the file). You can also scale the width of the figure, e.g., `[width=0.5\textwidth]` creates a figure whose width is half of the text width.

For centering the figure, include everything inside the figure environment into a center environment (i.e., `\begin{center} ... \end{center}`).

Tables

Tables are introduced with the `\begin{table}` command and concluded with `\end{table}`. This opens a `table` environment. The rest is analogous to the `figure` environment: you have the proper table (introduced by `\begin{tabular}`) and below this usually two lines containing the caption (e.g., `\caption{Title of Table.}`) and the label of the table (e.g., `\label{tab:123}`).

For the actual creation of tables, it is convenient to use the website <https://www.tablesgenerator.com/>, as Giuliano recommended in his hand-out. This is faster and more convenient than typing code by hand.

Diagrams

Diagrams like causal graphs can be created “by hand” using the `tikz` package and the `tikzpicture` environment (analogous to figures and tables). Whoever is interested in doing this can look up this up him- or herself.

However, there is also a convenient GUI (=graphical user interface) on <https://tikzcd.yichuanshen.de/>. You need to call the `tikz-cd` package in the preamble. Then, the website generates LaTeX code which you can just copy and paste into your `.tex` document. You have to embed the output into a

`figure` environment if you want that LaTeX treats it like a usual float (i.e., a table or a figure).

Float Placement

When you insert tables and figures into a LaTeX document, *it does not automatically appear at the place where you write the code*. This is an important difference to Word and other WYSIWYG programs. Instead, LaTeX chooses the most convenient place for the figure based on its principles of optimal typesetting. Tables, figures and diagrams are all so-called **floats**—because they are floating through the document according to how LaTeX wants to typeset the output.

You can feed LaTeX with your personal preferences for float placement. First, if you do not want to appear floats outside the section to which they belong, you can create a `\FloatBarrier` with this command. For this, you need to call the `placeins` package with the `section` option.

Second, you can indicate options for figure placement in the square brackets after `\begin{figure}[]`. If you write `[t]`, then you encourage LaTeX to place the figure at the top of the page. In normal documents LaTeX will not see typesetting problems and do so accordingly. If you write `[b]`, then the figure should appear at the bottom. If you write `[h]`, then the figure should appear exactly where you inserted it. If you add more than one option, e.g., `[thb]`, you impose a hierarchy of placement preferences. If you want to *force* LaTeX to place a figure at the top, bottom, etc., use an exclamation mark, too, e.g., `[!t]`.

Spelling Check with `ispell` (Unix)

Many LaTeX editors have an internal spelling checker that underlines misspelled words in red, or where you can run spelling checks from top to bottom like in Word. There is also a terminal-based alternative: `ispell` is an excellent Unix-based spelling checker with support for most Western languages. This program dates back to 1971, later it has been translated to C, but it is still operative! One reason more to use it. You install it with

```
sudo apt install tmispell-voikko
sudo apt install ispell
```

Then you simply run the spelling check with

```
ispell -t myfile.tex
```

The `-t` option specifies that the file is in TeX or LaTeX format so that the spelling checker will ignore macros.

LaTeX/Word conversion with pandoc

Perhaps you are afraid of using LaTeX because you might have to convert your LaTeX document to Word for some arcane reasons. Then copy and paste will be a mess. Don't even think about converting "by hand" or through spooky online converters! There is the **pandoc** program, a real miracle, written by philosopher-logician John MacFarlane. It does (almost) everything you need. You install it with `sudo apt install pandoc` and `sudo apt install pandoc-citeproc` on Linux. For Windows, visit this [site](#). You run **pandoc** with

```
pandoc -s myfile.tex -o myfile.odt
```

It is recommended to convert into OpenOffice format `.odt` first (because it is not proprietary) and to convert to `.doc` or `.docx` from there. Two common problems and solutions:

Bibliography If you execute the above command, the bibliography will be empty in the converted document. So you should add the extension `--bibliography=myrefs.bib` to the **pandoc**. Installing **pandoc-citeproc** as explained above is necessary for this extension to work. Elementary file conversion will work with simple **pandoc** as well.

Maths Sometimes math environments do not render as they should. Use the extension `--mathjax` to improve their conversion.

There are some minor imperfections: see this helpful discussion page.¹ However, all in all **pandoc** saves you a huge amount of time. Imagine you would have to delete all the LaTeX macros by hand and to typeset all expressions with sub- and superscripts in Word...

An alternative for document conversion is the program **latex2rtf** that generates a Rich Text File. You call it simply with `latex2rtf myfile.tex`. I like **pandoc** better because **latex2rtf** does not support many common packages. In particular, **amsmath**, **biblatex** and **enumitem** are not yet integrated. But try it out yourself!

pandoc also works in the opposite direction (i.e., from Word to LaTeX)! Admittedly, this will yield a higher workload, but if you want to recycle your old papers for your MA thesis, it is the way to go.

¹Some people recommend to convert first to a **pandoc** markdown file using **pandoc** `myfile.tex -o myfile.md`, to clean the markdown file by hand and then to convert (again with **pandoc**) to a format that Word can read. I did not find the results convincing, but wanted to mention the possibility.