Opinions and Experiences Writing Scientific Articles in LaTeX

Maximilian Soelch

July 23, 2021

This document provides what I consider best practices for writing in LaTeX. Many of these are subjective—ignore them at your discretion! This guide is biased towards scientific articles, yet many recommendations hold generally. To keep this document at a reasonable length, you will find a brief recommendation accompanied with a useful pointer to a more detailed resource. If you find any of the recommendations valuable or objectionable, let me know!

Contents

1 Golden Rules

There is no such thing as too many macros. Macros can feel like overkill at first. After all, you are only writing a note, are you not? Then the note turns into a paper. There will always be a point where you change your mind with respect to notation. And if you do it right, your source code becomes much more readable, particularly math equations.

Space is everything. A lot of what LaTeX does is spacing your content gracefully. Gradually grasping the inner workings is key to writing nice documents but also understanding when it fails.

A simple example is the difference between the dot after an acronym as in etc. and the full stop at the end of a sentence. The space between sentences is larger than that between words. You need to tell LaTeX the dot after etc. is not a full stop. Otherwise it looks awkwared, e. g., if you write like this instead of, e.g., like this. See the difference?

Avoid hard-coding. Avoid hard-coding a certain look of the document, be it via setting specific lengths, positions, font modifications. Instead, use macros, packages, commands, variables. Leave as much flexibility to the compiler as possible and your document will have a higher chance of looking good.

There will be advanced examples below, here is one for illustration: you want to *emphasize* a word? Instead of italics, `\textit`, use `\emph`. By default, it will use italics—but you can change your mind! And, while we're at it, `\emph` *changes* behavior *in italics!*

Stackoverflow is your friend. LaTeX has a steep learning curve. The key is to learn how to search the right queries. There are very few new problems in LaTeX. You'll get the hang of it!

Be consistent. Many of the suggestions in this document are only that—suggestions. Whether you choose to adhere to them is often a matter of personal taste. If you choose to follow some different standard, the key is to be conscious and consistent.

2 System Setup

Distribution Unless you are on a tight disk space budget, install a TeX distribution with lots of included packages like `texlive-full` for Linux.

Compiler The best default for compiling your document is the `latexmk -pdf` command, which should be included in your TeX distribution. It can detect which parts of the infamous multiple build runs, including the bibliography, need to be run based on changes in the files.

Editor Using a versatile plugin or dedicated TeX editor is recommended. I personally use TeXStudio. It is good-not-great, so you may find a solution that suits your preferences better. I have found its following features useful:

- Side-by-side display of source and pdf, which are tightly coupled so one can jump from the pdf to the respective source and vice versa.
- Easy setup of compiler and bibliography tool, including support of the aforementioned `latexmk`.
- Auto-completion of commands from used standard libraries as well your references in the document.
- Good integration of the log, jumping to the line that throws the error or warning.
- Seamless handling of multiple source files for a document.

3 Version Control

It is generally recommended to use version control for your source code. Following some rules makes using version control with TeX documents that much smoother:

Put each sentence on its own line in the source code, it makes comparing changes much easier.

Split large documents into multiple files. One file per section, integrated into the main document with `\input` or `\include`, is a good rule of thumb. That way, changing the order of sections has you change two lines, rather than several hundred insertions and deletions.

Do *not* commit the auxiliary files produced while compiling the document. Use, e. g., `.gitignore` templates[1] to exclude them.

Avoid committing the final document. This will create very large repositories that hold *every* committed revision. To make the latest pdf easily available, use continuous integration. With gitlab, it is as easy as adding a file

```
build_pdf:
    image: aergus/latex
script:
    - latexmk -pdf -cd relative/path/to/paper.tex
artifacts:
    paths:
        - relative/path/to/paper.pdf
```

as `.gitlab-ci.yml`. The final pdf can then be found at <link-to-repo>/-/jobs/artifacts/master/raw/relative/path/to/paper.pdf?job=build_pdf. With github, this is currently more convoluted, the main barrier being that the pdf is zipped, which is inconvenient.

4 Bibliography

Bibliography is one of the features where LaTeX really shines, but it can be a bit confusing to start. To work with bibliographies you need two tools.

[1] https://www.toptal.com/developers/gitignore

The first is a package that defines commands for referencing and the looks of your references and bibliography. There are two major contenders: `natbib` and `biblatex`.

The second is a bibliography compiler. This compiler is the link between your `tex` and `bib` files. Again, there are two contenders: `bibtex` and `biber`.

If there are no constraints, the choice is easy. The combination `biblatex` and `biber` is the more modern and thus strictly superior combination if you fancy support of, among many other things, UTF encoding and media from this millennium in your bibliography.[2] The only—unfortunately rather common—exception is when your template forces you to use something else, usually `natbib` and `bibtex`.

Use `biblatex` via `\usepackage[...]{biblatex}` and tweak the looks of both your references and your bibliography to your liking via package options. For a suggested default, see the provided `nonpaperheader.tex`. A best practice is to *never* use the `\cite` command unless you are writing a macro of your own. Instead, use `\textcite` when your reference is part of a sentence or `\parencite` when your reference is in parentheses at the end of a sentence for the most common use cases, and check the docs if you need something else.[3]

Then there is the actual bibliography. I would recommend using a dedicated bibliography tool rather than managing your `.bib` file manually. There are many good and some bad options out there but no universally good one. I have personally made good experiences with the free, open-source tool Zotero in combination with the add-on Better BibTEX as well as the browser plugin for Firefox. It can conveniently export a TEX-compatible bibliography file and also keep it updated if new papers are added or the data of a reference change. Conversely, Zotero can import a bibliography entry from a BibTEX entry.

The best source for bibliography *entries* is the dblp computer science bibliography data base (https://dblp.uni-trier.de/). It usually has the cleanest and most consistent data. It just struggles with arXiv preprints, for which I have had the best results by using the Zotero browser plugin directly on the arXiv page of the respective paper.

5 Math

5.1 The Correct Math Environment

There are a number of ways to display equations. Without going into the details, use the `equation` environment. Using `$$ ... $$` is discouraged, as is `align`. If you need the latter, use an `aligned` environment inside the `equation` environment.[4]

[2] For a more detailed and nuanced take, see https://tex.stackexchange.com/questions/25701/bibtex-vs-biber-and-biblatex-vs-natbib

[3] If you have to use `natbib`, the equivalent commands are `\citet` and `\citep`. Via the `natbib=true` option of `biblatex`, you can enable backward compatibility of these commands.

[4] As for the reasons, check out https://tex.stackexchange.com/questions/321/align-vs-equation and https://tex.stackexchange.com/questions/503/why-is-preferable-to.

5.2 Equation Numbering

Number every equation unless you have a good reason not to. Not wanting to break the line is often not a good reason, unless you are on a really tight budget with space. You may not need to reference it. But your co-authors, colleagues, peers, reviewers, or students might. That third equation in the right column on page six does not roll of the tongue quite as well as eq. (2).

5.3 Flushing Equations

Flush equations left if you have the choice. You can most easily achieve this with the `fleqn` option to the `amsmath` package.

An equation like

$$E = mc^2 \tag{1}$$

reads better than

$$E = mc^2 \tag{2}$$

in my opinion, but your tastes may differ.

5.4 Operations

A common trap in setting math is misunderstanding space. An easy way to avoid that is to not hardcode certain symbols. An example is `|`. If you type `$A | B$`, you get $A|B$. LaTeX does not understand that the pipe relates the two adjacent symbols. What you actually want is the operator: `$A \mid B$` gives you $A \mid B$.

Note how, e. g., `$A + B$`, gives you $A + B$ with correct spacing even without a macro. LaTeX invokes the correct spacing for the most common operators like $+$, $-$, $>$ etc. Beyond those, you're on your own.

5.5 Text in Math Blocks

If you write min (`$min$`), you need to define m, i, and n.[5] The quickest fix is to write `$\min$` for min. In cases where the predefined macros do not exist, the quick-and-dirty fix is `$\text{min}$`. The correct fix is to understand the semantics and tell LaTeX explicitly, for instance by declaring something an operator with `$\DeclareMathOperator*{\mymin}{min}$` for min. This looks the same as the `\text` solution, but subsequently gets other spacing right, for instance with indexing:

$$\texttt{\textbackslash text:}\quad \text{min}_x \qquad \text{vs.} \qquad \texttt{\textbackslash mymin:}\quad \underset{x}{\text{min}} \tag{3}$$

The correct way of handling it depends on your downstream needs.

[5] Caveat lector: my document defaults make this less pronounced, more by coincidence than by design. I assure you that in the default default, it just looks ugly.

5.6 Parentheses

To automatically scale your parentheses to an appropriate height, use the `\left` and `\right` commands as in `\left(\frac12\right)` for

$$\left(\frac{1}{2}\right). \tag{4}$$

I have found that there are not too uncommon special cases where this solution adds too much spacing on both sides. My default fix is to use the `mleftright` package which provides drop-in replacements `\mleft` and `\mright`, and if you add `\mleftright` in your preamble you can even keep using `\left` and `\right`.

5.7 Punctuation

Equation blocks require punctuation just like text. See the full stop in eq. (4). If you have multiple equations, separate them with commas or full stops. The rule of thumb is easy: read the equation block like prose, and put commas where appropriate.

6 Miscallenous

6.1 References & Citations

You probably know the `\ref` command to reference a section, equation, figure etc. Use the `cleveref` package and the `\cref` command as a drop-in replacement. It gives you a lot of options to configure, looks good by default, and you can change your mind at any time. No more inconsistencies between Figure 5, figure 5, and fig. 5.

When you label things that you want to reference, add a tag to the label, e. g., use `\label{sec:introduction}` or `\label{eq:Emc2}`. That makes it much easier to see at a glance what it is you are referencing. Also, your editor might have auto-completions, and these tags are an easy way to filter.

6.1.1 Links and URLs

The defaults used in most documents for links both to the internet as well as to references within the text are stuck in the previous century. Write URLs in the font of your text.[6] Mono-space font for URLs is a relic from times when people were surprised to see them.

Switch off those neon boxes![7] Their use is very limited, they look horrific, and worst of all they make reading your text unnecessarily hard. I know where I can click, and if I do not, I likely did not want to in the first place.

[6] And here is some correction for nicer URL formatting: https://www.joachim-breitner.de/blog/519-Nicer_URL_formatting_in_LaTeX

[7] And here is how to do that including some alternatives: https://tex.stackexchange.com/questions/823/remove-ugly-borders-around-clickable-cross-references-and-hyperlinks

6.2 Figures

Make your figures as large as necessary, but not larger. Try to keep the overall file size low. This reduces the danger of your reviewer being annoyed by having to work to get your stuff printed because their printer runs out of memory when trying to print your paper. I have been this reviewer.

Use vector graphics wherever possible, i. e., use pdf outputs of your matplotlib figures. An exception to this rule are plots with lots of elements. This is typically the case when you have scatter plots with > 100 points. Convert those into png or some fixed format. If you want to keep vector graphic axes and labels, scatter plots also have a keyword argument `rasterized=True` that is a good compromise.

As for the right package and commands, I have had the most consistently good results with the `subfig` package.

6.3 Acronyms

Use a package and macros for all acronyms. This gives your acronyms consistency, and again it is easy to change your mind later. I have made good experiences with the `acronym` package. As with many things LaTeX, it seems superfluous at first. Once you have a sizable collection of acronyms to carry over from document to document, it is the much more efficient option.

6.4 Tables

Tables are an art in and of itself. Arguably, it is one of the more annoying subjects with TeX. Here are some very incomplete, general recommendations to give you a start:

- Vertical lines as dividers are a sign your layout is incorrect. Use them sparingly, ideally never. White space is almost always the better option as a separator.
- Align your columns correctly. The alignment is determined by the content, and the column header has the same alignment. Text is left-aligned by default. Numbers are right-aligned by default.
- Small Caps are often, not always, a good default for column headers, the command is `\textsc{Column Title}`.
- Make use of `\toprule`, `\midrule`, and `\bottomrule` from the `booktabs` package; but use them with care also. Consider white space, which you can add with `...\\[1em]` at the end of a table line, where `1em` is the amount of white space you want to add.
- Make sure your table is not overburdened by content. Choose appropriate font sizes to get enough white space in. Cut insignificant digits. Imply duplicate values by clever grouping and white space instead of printing it in every row. This is not an SQL table, your target audience is human for the time being.

A recurring pattern here is white space. Mastering white space is the key to readable, informative tables.[8]

[8] If you want to see these principles at work, look at this example: https://github.com/Wookai/paper-tips-and-tricks#tables

6.5 Numbers

Setting numbers in text correctly is not trivial. Did you run 50 or 50 experiments? Are you sowing the results in section 5 or 5? Did you run them in 1994 or 1994?

Once again, consistency is key. The distinction that I have settled on is between *text* and *math* numerals.[9] At times, those notions are a bit ephemeral, but a good rule of thumb is whether you would be likely to assign the number to a variable. You ran $N = 50$ experiments, hence math mode even though you do not perform math on the number. You ran them in 1994 and recorded them in section 5, these numerals are part of the prose.

Do not fall for the but-the-result-is-the-same trap. If you want to know why, read the Knuth comment above.

It is often recommended to use the package `siunitx`, but I have little to no experience with it.

6.6 Footnotes

Footnotes are great to add a little information or pointer on the side. If you put the footnote at the end of a sentence, put no space behind the full stop. If you want to put the footnote on a new line in your source code—for instance, because you adhere to the version control best practices in section 3—you need to use a little trick to avoid the space. Escape the newline with a comment! Put a `%` at the end of your line, then start the footnote on the next line.[10]

6.7 Quotes

Punctuating quotes correctly is a bit annoying. Firstly, different languages have wildly different conventions. Secondly, LaTeX has many different ways to typeset them. The consistently easiest way to deal with this in my book is the `csquotes` package with the `\enquote` command. Then it becomes a matter of setting up your preamble, specifically the document language, correctly and everything remains nice and consistent.

6.8 Table of Content

In your table of content, do not align the page numbers to the right. It may look fancy, especially with those filling dots you get by default. Form follows function. Help your reader find pages, not sum them.

6.9 Log

When you compile your document, take the logs seriously even when they only produce warnings. A document is only ready for publication when it has no warnings that you have not at least googled and understood why you will not fix it.

[9] It is always a good sign when you can quote Donald Knuth to back your opinion, cf. the last paragraph on page 31: https://tug.org/TUGboat/tb10-1/tb23knut.pdf

[10] This trick of escaping the end of a line can be useful elsewhere to avoid spurious white spaces in your document.

If you are brave, add \usepackage[orthodox]{nag} to your preamble.

6.9.1 Overfull and Underfull Boxes

The most common warning that should be purged completely are overfull or underfull boxes. The compiler throws these warnings when it cannot arrange your content with acceptable spacing. Often, this means excessive spacing when the box, a line or a page, is underfull or content bleeding into margins when the box is overfull. There are many small ways to fix these, manually breaking the line, telling the compiler how a certain word is hyphenated, re-ordering or rephrasing a sentence, and many more.

If you have trouble finding the culprit, there are tricks to find them. For overfull boxes, add the draft option to the documentclass options. This will add a black marker on the margin next to the overfull box. To see where the box ends, it can be useful to display frames with \usepackage{showframe}.

Underfull boxes are a little trickier, but with showframe and a keen eye for superfluous white space they are usually easier to spot in the document.

6.10 Publishing Source Files

With the majority of papers uploaded to arXiv these days, it is quite common to share not only your article but also the source code. Keeping in mind that the source code on arXiv is made public, it is good practice to purge your source files specifically for upload.

Run the following commands:

```
latexpand --empty-comments paper.tex > paper-full.tex
sed -i '/^\s*%/d' paper-full.tex
cat -s paper-full.tex | sponge paper-full.tex
```

The first line combines your source code distributed across several files back into one big file. The other two lines remove all comments and superfluous white space in your source file. This way, you do not accidentally publish comments that were not meant for the public eye.

There is a nice python script[11] that analyzes your source code and removes comments, temporary files etc.

It is particularly good for deleting unused files. It also can help you with resizing your images if you have a large paper. It can be a bit too liberal in its compression, so be sure to check the results or deactivate. Either way, it is better to not trust a script with handling your figures.

In any case, be sure to check that this cleaned-up version of your source produces same result.

6.11 Title Casing

Be sure to have consistent title casing. I personally prefer mixed upper- and lower-case title casing, but there are many alternatives. The key, once again, is consistency. There are resources to help you: https://capitalizemytitle.com/

[11] https://github.com/google-research/arxiv-latex-cleaner