LaTeX Tips & Template & Writing Advice

Marcus Hutter

Google DeepMind & ANU

http://www.hutter1.net/lattt.pdf

August 28, 2025

Abstract

This document serves 3 purposes, intended for my students and others who might find it useful. (1) A non-polished annotated LaTeX mega-template. (2) Some LaTeX-related advice but is no substitute to reading [KD04]. (3) Some technical writing guidelines but is no substitute for reading [Hig98, Zob15]. [KD04, Hig98, Zob15] and LaTTT are the bare minimum to know for good scientific writing. LaTTT does not contain advice on how to do research or become a good scientist. Skip over the first unnumbered sections and start with Section 1 "Introduction" on page 4. Suggestions for improvements are welcome. Download the latest version of this template from http://www.hutter1.net/lattt.tex

Contents

Keywords

LaTeX, tips, tricks, template, students, annotated, draft, writing guide.

List of Definitions, Theorems, Examples, ...

A list of definitions is useful for a PhD thesis and for long technical reports with many theorems etc., but not for the typical journal or conference papers. Still it can be useful to enable this list for yourself while working on a paper, even if finally disabled.

Private Notes

*r=research, w=write, v=done, /=obsolete, -=todo, !=must, *=important, ?=maybe*

This section contains private notes. Most people would keep them in a separate file. Advantages of keeping them here is that you can dynamically refer to equations/sections/etc. by `\ref{label}`*. Disable this section with* `\iffalse\fi` *if keeping it compilable is annoying and useless.*

Submission ToDo. *- Proof-read the whole paper 3 times, on a sentence level, a word, and a symbol level. v Set if in bib to false so internal bbl is used. - Use pdflatex and not tex->dvi->pdf (for pgfplots and href to work)*

ToDo. *v Take into account all important items in all todo lists. - Move items in Section 'Miscellaneous' to an appropriate section, if existent/possible. - Wrap text diagonally around Figure 2.*

To Research. *- Find better/newer/more useful packages. - Check out compression tips at http://thomas.deselaers.de/computing/texsqueezing.html*

To Write. *? Paragraph 'Motivation and 'Contents' and in Section 1. v Make list of macros: symbol/macro/explanation, similar to List of Notation. - Avoid using publisher-specific macros as far as possible*

Experiments. *- No experiments in this paper.*

Maybe/Extended. *? Add general advice on how to write a good research paper, or refer to GoodWrite.txt*

More ideas. *- None at the moment*

Paper Notes. *- References to other notes, e.g. those on paper only, in order not to forget that they exist and where to find them.*

To Read. -

Have read. *v Higham (1998) Handbook of Writing for the Mathematical Sciences*

Titles/Keywords/Abstract.

Titles: - LaTeX Tips & Template - Annotated LaTeX Template - LaTeX Template and Advice for LaTeX novices

Keywords (for title): LaTeX, tips, tricks, template,

Info: - ERA FOR codes: Default: 010404(40

ASCII-Abstract (pure ASCII version of abstract for various services such as arXiv):

Contents. *Before you start writing a paper, make a detailed table of contents, at least down to paragraph level. Thereafter use it for structuring your actual paper, and tick off completed items.*

Introduction: - Introduction (what problem attacked) - Motivation (who cares) - Main contribution (What's new) - Contents (Sections)

Setup/Preliminaries/Main Results/Notation/Definitions/Related: - Related work (state of the art)

Preliminary Results: - paragraph

Main Result: - paragraph

Experiments: - Setup - Results - Evaluation

Discussion: - Summary - Outlook - Conclusion (risks and payoffs)

Appendix: - Proofs - Background material

"It may be nice to start a more easy-going paper or your PhD thesis with a quote."

— *Marcus Hutter (2012)*

1 Introduction

This document is an annotated LaTeX template with tips and tricks and pitfalls I've collected over the years. There is some learning curve for LaTeX and the recommended packages, and this template itself takes some time to learn and get used to, but will save time in the long run. Of course, adapt and improve it to your personal needs. If the template is too massive for you, simply strip it down to what you need now, and incorporate more once you are more experienced and/or need it.

This document is neither an introduction to LaTeX nor a comprehensive advice on how to write a good research paper or thesis. For a light first introduction to LaTeX, see e.g. [GH97]. For general advice on good technical writing, see e.g. the clear, structured, to the point, informative, and funny book [Hig98], and some of the 300 references therein.

This document is mainly about issues related to "type-setting" and technical writing advice. It is continuously under construction, so neither well-polished nor well-balanced. It grew and evolved over the last two decades, so is a bit messy, and some LaTeX choices are dated due to more modern packages. Download the latest version of this template from http://www.hutter1.net/lattt.tex. A PdfLaTeX-compiled version with all private comments enabled is at http://www.hutter1.net/lattt.pdf. If some external links are broken, you can usually recover the content via the Wayback Machine https://web.archive.org/.

The template, or at least several of its features, can be useful even if the final version of your paper must conform to a different style. Indeed, it is possible and useful to toggle between styles.

Motivation. *[[ToDo: write motivation]]*

Main contribution. Providing a LaTeX template and scientific writing advice for my PhD students (and others), to help them get started with LaTeX and writing papers, and save their (and my) time.

Highlights what novice writers often do wrong:

LaTeX advice:

- *Use LaTeX* for typesetting your paper, not any other document editor.
- *Tidy:* Keep your LaTeX neat and tidy and well-documented like code, but clean it before uploading to arXiv.
- *Branching:* Essentially never branch your LaTeX file.
  Use compiler switches `\if\then\else` instead.
- *Vector graphics:* All diagrams must be vector graphics (svg or pdf) never gif or png. jpg only for photos.

- *Diagrams:* Create diagrams preferably with LaTeX's pgfplots/tikz, but Python's matplotlib is ok too. Use strong/pure colors.
- *Scaling:* Never use scaling commands (esp. if they change font size). This includes diagrams and tables.
- *Font size:* All fonts must be at least 80% text-size, including in all diagrams (math indices are the only exception).
- *Logarithm:* The default diagram axis should be logarithmic. Only use linear if logarithmic really makes no sense.

Writing advice:

- *Must read* (not skim) [Hig98, KD04, Zob15] and LaTTT before you write your first paper. These highlights are not enough.
- *Planning:* Have a detailed plan before you start writing. Write down a table of contents (= todo list) of your paper down to single fact (paragraph or even sentence) level.
- *Paragraphs:* Every paragraph needs a (potentially invisible) heading. Same for theorem(-like environment)s.
- *Proof-read* your own text at least 3 times. This is your job; nobody else's.
- *Produce quality:* Regard your paper as a piece of art. Don't write ugly papers.
- *List of notation:* Every paper needs a list of notation.
- *Abstract* needs to answer: current sota, what, objectives, how, what's new, who cares, significance, effectiveness, expected impact, consequences, with minimal jargon.
- *Introduction* needs to cover: background, rationale, problem statement, objectives, scope, limitations, assumptions.
- *Know and follow all rules* and guidelines and advice until (a) you truly understand why they exist and (b) your 10th paper and (c) you have better counter-arguments and (d) your co-authors agree.

Contents. *[[ToDo: write contents]]*

2 LaTeX Templates & Macros & Packages & ...

LaTeX. Scientific papers are usually written in LaTeX, since LaTeX is designed for it. Its main advantages over many other typesetting systems are the separation of the logical structure of the document from the typesetting, the professional layout of text, its power to type-set equations, its many packages for specialized purposes (some described below), it is free, and it is widely used - most science undergraduates know it. Its main disadvantages are that it is not Wysiwyg, it takes more time to learn, some simple things are hard to figure out how to do, and it can be stubborn and resist to do what you want.

- *Use LaTeX:* Since there is no acceptable alternative to LaTeX, the decision whether to use it or not is an easy one.

- *TEX:* While TeX provides low-level commands and does the type setting, LaTeX provides additional high-level commands for structured documents.
- *Windows:* MiKTeX is the most popular (La)TeX typesetting package for Microsoft Windows (also available for Linux). WinEdt (Windows-only) is a great LaTeX editor which works smoothly with MikTeX. It's not free, but cheap and worth its money, and many Universities have a licence.
- *Linux:* Under Linux, a popular and good package/editor is TexMaker (and the similar TexStudio). My preferred all-purpose editor (for coding as well as LaTeX) is Visual-Studio-Code (with LaTeX Workshop extension)
- *Overleaf* is the easiest but not the best tool for collaborative LaTeX editing.
- *Tidy:* Keep your LaTeX neat and tidy and structured (as e.g. LaTTT.tex) as you would (hopefully) do with your program code (and everything else in life).
- *Must read:* Read (not skim) Kopka's guide to LaTeX [KD04] when writing your first report and on demand about TeX [AHB03, Eij92].
- *Examples:* If you need to mimic a construct in some of my papers, you can find the LaTeX source at http://www.hutter1.net/official/bib.htm and/or at arXiv.org.

Templates. There are many style files that determine the global layout and often provide extra commands. The most common standard style is `\documentclass{article}`. Many publishers demand that you use their style which builds on this one but with customized layout and extra commands. Many authors (like me) develop over time own styles e.g. for TechReports with own macros.

I found it most effective to maintain a general paper-template (this one) and base each new paper on a ruthlessly adapted copy if it. The alternative, of developing a separate style file, and then use it for own papers, did cost me more time (and nerves). Each paper is different and one is constantly improving or changing preferences. Papers can take years from initiation to publication, which constantly causes problems when trying to compile old papers with your newer style file. Keeping versions around seems even worse than the template approach. Always using the previous paper as a template for the next paper also has obvious disadvantages.

Most conferences provide their own style file and template you have to use. Search for `\ifconf` in this template and include the appropriate pieces of their template there. This is usually easier than it sounds and worth it.

Macros. Defining and using macros has obvious advantages. They allow to abbreviate long and repeatedly occurring expressions, and, if used consistently, make it easier to later change style or notation. They also make you independent from publisher-specific macros.

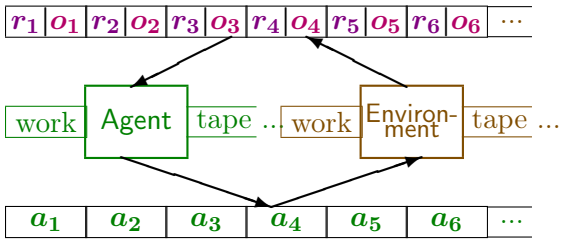
There are several 'standard' macros defined in the preamble of this documentand listed and explained in Appendix A. For instance `\beq\eeq` (`\beqn\eeqn`) `\bqa\eqa` (`\bqan\eqan`) generate (un)numbered equation (arrays). Apart from being shorter than the original commands, they are more systematic and hence easier to remember, and can be customized to change e.g. spacing.

Compiler switches. One of the most useful features of TeX are compiler switches. With `\newif\ifxxx`, `\xxxtrue` or `\xxxfalse`, and `\ifxxx \else \fi` you can define, set, and use compiler switches. Several are defined in the preamble of and used in this document.

For instance, a useful switch is for private comments within the text. Use `\todo{some todo}` or `\note{some todo}` for scattering todos and notes throughout the document, and set `\privatetrue` or `\privatefalse` to enable or disable them.

Compression. Often you write a paper for a conference, which turns out to be too long. Most conferences have page limits. You could branch into a long journal/TecReport version and a short(ened) conference version, but then you have to maintain two documents until acceptance of your conference paper which may take a while. The alternative is to scatter `\ifshort` *long text* `\else` *shortened version* `\fi` or `\ifshort\else` *optional extra paragraph* `\fi` within a single file. This also avoids the embarrassing problem that your supervisor fixed bugs in your conference version, and later is annoyed to find the same bugs in a journal version and/or your PhD thesis. `\conftrue` and `\conffalse` switches between conference style and journal/TecRep style.

When you're down to (nearly) the right number of pages, but then have to add a sentences, it can get quite annoying to trim it back. `\compresstrue` removes a bit of space in lists, figures, and equations. Used mildly for an initial submission is usually ok, but must not be used (and hence is disabled) in the final version. More aggressive space-saving techniques should not be used [DHD20].



Finally, by wrapping text around small figures such as the one on the right with the **wrapfig** package, one can save quite some space.

Packages. Many useful additional packages have been developed for LaTeX and can be found at the Comprehensive TeX Archive Network http://www.ctan.org/. The ones I found most useful are included in the preamble of this document. Some of them are described below. In general, use packages sparingly and only standard and well-documented and stable ones available from CTAN or those from publishers that you have to use, since some packages do not work together, some make LaTeX slow such as pgfplots, and to minimize problems with the publisher.

Non-ASCII Symbols. Math heavily uses non-ASCII symbols, such as different alphabets (e.g. greek γ), different fonts (e.g. $\mathbb{N}$), special symbols (e.g. $\sum$,$\int$), $^{\text{super-}}$ and $_{\text{sub}}$-scripts, and many others. Traditionally they are entered via an escape sequences '\', e.g. ''`\int`' becomes $\int$. `\usepackage[mathletters]{ucs}` and `\usepackage[utf8x]{inputenc}` allows to enter many UniCode esp. math symbols directly in the LaTeX source, which increases readability of the LaTeX source. XeTeX is even more powerful. Of course this is only/most useful if you can conveniently

enter them. In Linux you can activate a 'Compose' key and file for that purpose, and can also define your own key associations in the file ''.XCompose' in you home directory. Here is my .XCompose file.

LaTeX - Miscellaneous.

- *Demarcate* cleanly, at all time, (1) polished/finished paragraphs/parts, from (2) unfinished parts under constructions (put them in `\ifuc\sl ...\rm\fi`), (3) private comments (put them in `\ifprivate\it...\rm\fi` or `\todo` or `\note`), (4) old/obsolete parts (put them in `\ifold...\fi` or `\iffalse...\fi`), (5) good but excluded stuff (put them in `\ifoptional...\fi`), etc. Once a part under construction is finished, remove the `\ifuc...\fi`. This helps your co-authors navigate your mess, and at any time you can compile a clean sharable version.
- *Symbols:* A comprehensive 150+ pages list of LaTeX symbols can be found at: http://www.ctan.org/tex-archive/info/symbols/comprehensive/
  Easier, simply draw a symbol you need and get the LaTeX command: http://detexify.kirelabs.org/classify.html
- *LaTeX experiments:* Use `\ifuc\fi` also for LaTeX "experiments". Iff not possible, add `\todo{Remove}` or so.
- *Displayed equations:* Use `$$` or `\[` or `\begin{equation}` wherever possible (smarter vertical space layout) and `\begin{align}` only when needed.
- *Packages:* Use packages sparingly! Don't use unmaintained packages that haven't been updated for 5y+ years. Don't use packages for a single simple feature which can be emulated w/o a package with a couple of LaTeX-lines.
- *Comments:* Use `%` only for short comments e.g. invisible paragraph headings, or single unused sentences. Use `\if* \fi` for old, under construction, optional, private, etc stuff, and leave a comment why this is "commented" out and the intended future use of it.
- *Macros:* Design and use own macros systematically and judiciously and consistently.
- *Inlined fractions:* Avoid inlined fractions (use `$()/()$`). Never use inlined double fractions, or inlined $\frac{a_i}{b_i}$.
- *Line breaks:* Break LaTeX lines at meaningful places (fullstops, commas, sentence parts, etc) to ease editing. Don't break at fixed width or arbitrarily, nor have a whole paragraph in one line (it messes up github diff). (not a hard rule)
- *Redundancy:* Minimize redundant use of `{}` and use space in math strategically rather than excessively, e.g. `V_\nu^\xi` instead of `V_{\nu}^{\xi}`, and `\sum_{\nu\in\M}` instead of `\sum_{\nu \in \M}`.
- *Brackets:* Use `\left(` and `\right)` cautiously and replace by `\Big(` or `big)` etc. if they stick out too much. Essentially never use them inline.
- *Breaking math:* Break LaTeX math at meaningful places, e.g. outermost equalities.

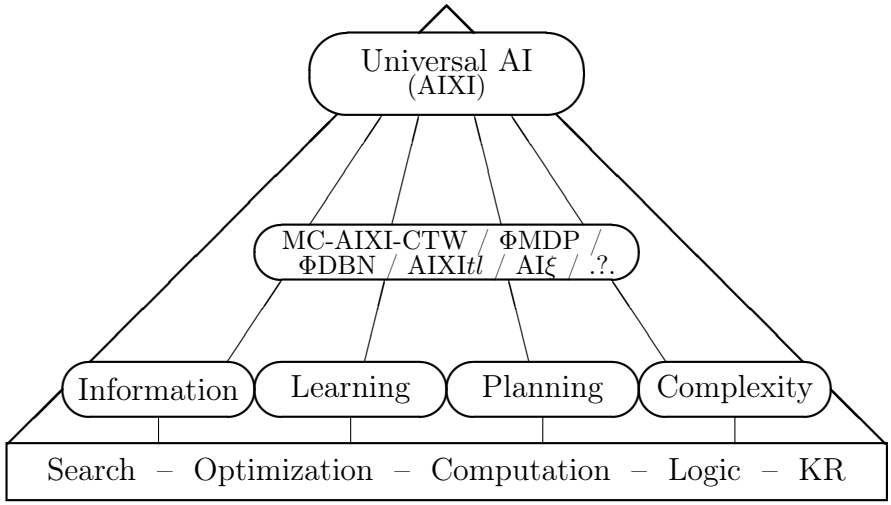
- *Citation style:* Use `\citep` for '(cite)' and `\citet` for 'cite', even if alpha-style does not distinguish, since the publisher/conference may demand a different bibstyle.
- *Consistency:* Use LaTeX consistently and systematically (spacing, `{}`, `_^` vs `^_` etc.). One reason is to make global find-and-replace easy. E.g. always use `V_\nu^\pi` (`_` before `^` and no brackets) instead of mixing it with `V^{\pi}_\nu` and x other variations.
- *Referencing floats:* Some non-satisfactory solutions to automatically reference esp. figures as 'here' or 'below' or 'Figure #' Package `varioref.sty` or https://tex.stackexchange.com/questions/32285/referencing-to-above-or-below satisfactory. or https://tex.stackexchange.com/questions/646462/how-to-add-a-hint-arrow-up-down-at-the-reference-for-direction-in-which-to-loo
- *Paragraphs:* Use `\paragraph{Heading.}` within long (sub)sections. A (visibly) named paragraph should have length $\approx \frac{1}{2}$ page. Theorems/Examples/Figure[H]/Remarks also count, i.e. visible paragraphs are not needed if there are plenty.
- *QED adjustment:* Use `\\[-8ex]` at the end of the last line inside an `align` environment if it is the last displayed equation in a proof or example or remark to place the end-of-env symbol properly.
- *Good but optional stuff:* Mark good stuff to exclude with `\ifoptional % in good condition but excluded because ... \fi`
- *Theorem titles:* Every theorem/proof/example needs a description `\begin{theorem}[name or description]`
- *Spaces:* Be very careful and consistent with spaces and empty lines. Know the difference and what you are doing. E.g. use `e.g.\_` not `e.g._` for correct spacing (`_`=space). It results in 'e.g. bla' (good) rather than 'e.g. bla' (bad). E.g. `<empy-line> \end{proof}` creates an empty line in the pdf and/or unwanted paragraph indentations. `Hello \index{blah} World` creates a double space. Put % in an empty LaTeX line and glue `Hello\index{}` etc. Never use absolute spacing such as `3cm`, but use relative spacings such as `8ex` or `0.6\textwidth`.
- *Tabs:* Don't use tabs but 2 or 4 spaces.
- *Non-breakable space:* Use `Section~\ref` and not `section \ref` but best use `\Cref{}` of `\package{cleveref}`.
- *Orphan words:* `\looseness=-1` at the beginning of a paragraph avoids

3 Diagrams

There are powerful tools to create sophisticated diagrams and plots. The result has then usually to be exported as an svg or eps or pdf *vector* graphics. *Never* produce and include a pixel image from a vector graphics. It's ugly, wastes space, and is unprofessional. Some tools allow to export or create diagrams in LaTeX. Whenever

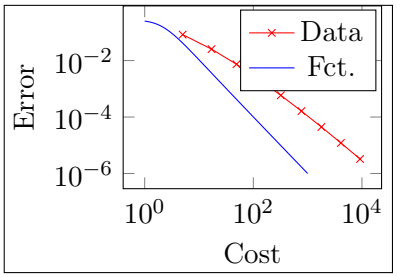
possible, use them: Fewer tools to rely on, better integration with LaTeX (e.g. LaTeX equations within diagrams), easier and more consistent layout and type-setting. Some of them are recommended below.

LaTeX Pictures. The internal picture environment of LaTeX supports (only) simple diagrams (line,vector,circle,box.text) such as the agent-environment diagram above or such as



It is not very comfortable to create them. It usually requires you to draw the diagram first on grid paper and then transfer the coordinates, but for simple diagrams such as the two presented ones this can be effective. Especially if space is tight, one can even create inlined diagrams such as $\overset{y=a}{r=2/3}$ (s) $\overset{y=b}{r=2/3}$ (e) $\overset{y=*}{r=i}$. There had been a (by now clearly outdated 40 year old) MS-DOS tool named TeXCAD, that allowed to draw pictures and to export them as LaTeX picture files.

Package for plots: pgfplots. Definitely use package pgfplots for all your plots (and tables). It's fantastic. You don't need to know much for easy plots such as the one on the right. The many examples in the 360-page manual which come with the package allow to cut and past and adapt even sophisticated plots, without having to read the manual. I got great plots after a couple of hours such as Figure 2, which still requires less than 30 lines of LaTeX code. You could even consider automatically generating the LaTeX code from your program. Very often two figures have to be aligned side-by-side as Figures 5 and 6.



Linear vs logarithmic scale. Most people use linear scale (5,10,15,20,25,30,...) by default and only if the plots look terrible switch to logarithmic (1,2,4,8,16,32,...),

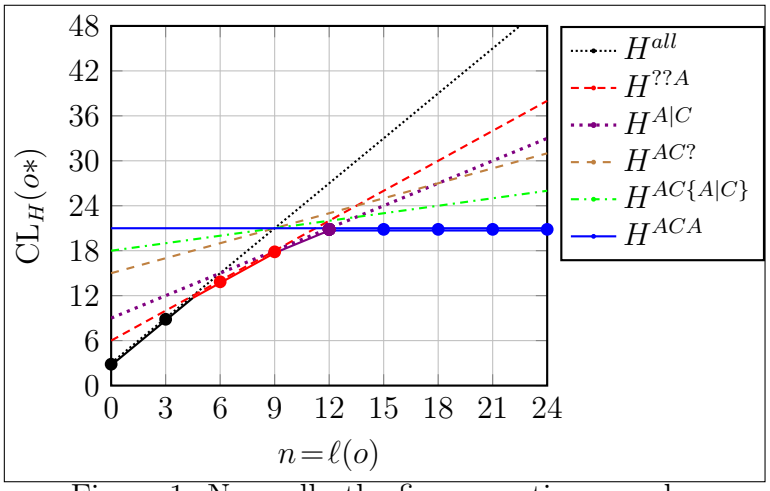


Figure 1: Normally the figure caption goes here.

Figure 2: This is a more sophisticated diagram created with pgfplots. Still it requires less than 30 lines of LaTeX code.

Figure 3: You can also put the caption on the side or have two figures side-by-side with separate or joint captions.

and often not even then. *The default scale for virtually everything should actually be logarithmic*, since linear is rarely better. This is particularly true for "big-data" plots. For instance, on a log-scale x-axis you can much better see what is going on for small x compared to a cramped linear scale. While large x get compressed, this region often lacks detail anyway and is primarily interesting for its asymptotics, which can actually be better read-off from a log plot: Linear plots often appear converged even though they are not. If $x=0$ is important, shift $x \rightsquigarrow x+1$, otherwise the first data point gets cut off. Even for $x \in \mathbb{R}$, a both-sided log plot may be useful $(\mathrm{sign}(x)\cdot\log(|x|+\varepsilon))$ [Hut13, Fig.1]. So *using linear scale needs justification, not logarithmic scale.* For log-x-plots, if the scale is extensive, it is also important that the data points are exponentially spaced. For instance in python, use `np.geomspace(1,1000,25)` or if x needs to be an integer, use `np.unique(np.geomspace(1,1000,25).astype(int))`. This "rule" actually applies way beyond plots: Even more important, experimental parameter-sweeps are nearly always better and more efficient if spaced exponentially, rather than linear. If you're still not convinced: Logarithmic spacing is even used for currency: We do not have 5,10,15,20,25,30,35,.... spaced Dollar notes but 1,2,5,10,20,50,... for good reasons.

~~€~~	½	x
⇝	x	x
table		x

More packages for diagrams.

- *Tikz:* Use package **tikz** to create (low-level vector) graphics such as flow charts. http://www.texample.net/tikz/examples/ is a useful repository of examples.
- *Xy:* Use package **xy** for simple graphs and diagrams such as

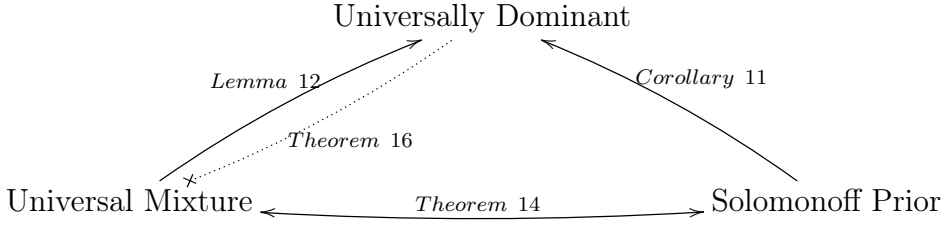


Figure 4: Very often two figures have to be aligned side-by-side as the ones below, It's a bit tricky to do this properly and perfectly. Captions are either joint (as this one) or separate (below), but *never* both (as here; use package subfigure for that).

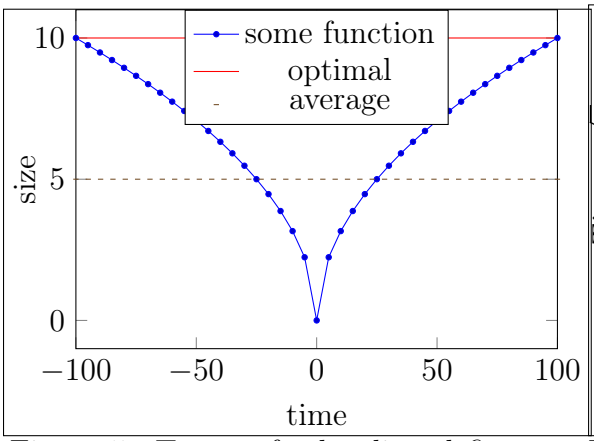


Figure 5: Two perfectly aligned figures. Box is only to illustrate minipage width.

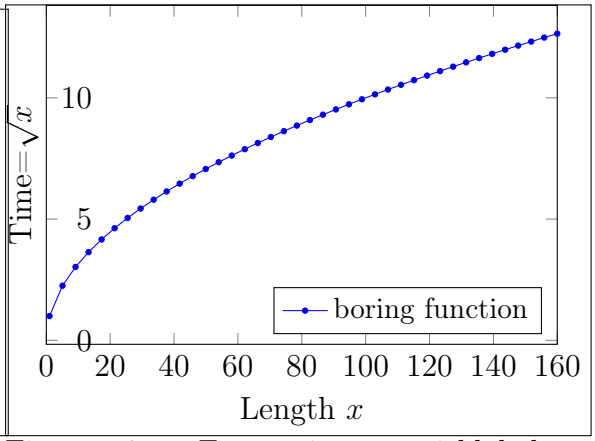


Figure 6: Fancy inner yticklabels. Box is only to illustrate minipage width

- *External figures:* Use and include externally generated eps/pdf/png image (with `\includegraphics`) only if the above tools cannot be used (effectively).
- *Beamer:* Use package **beamer** for technical slides containing equations. It's widely used and very powerful.
- *Posters:* There are also LaTeX packages for making posters. See http://www.hutter1.net/ai/sqlearn.pdf for such a poster example.
- *Payoff:* As with LaTeX itself, it takes some time to learn these tools but the time is paid off within a year or less.

Do NOT scale your diagrams:. TL;DR: Font size in diagrams need to be at least 80% of the main text font size - no exceptions. How to efficiently and professionally achieve this: Make the default size of all your diagrams 5cm×4cm with labels in normal text size. This is likely the maximal size you can afford in conference papers due to the page limits, and is also often a reasonable size for journal papers, putting 2-3 diagrams side-by-side. If/once you know the target size better, use your best estimate of it. Do *not* scale images in LaTeX e.g. with `width=` or some scaling environment like `\adjustbox`. Even if you generated a 20pt font (and double line with) and scaled it to 50%, this is *not* the same as (unscaled) font size 10pt (and standard line width). You *may* use the `scale`-command in `tikzpictures`, since it scales the figure without scaling font sizes or line widths, but still you want to do this from the outset and not as an afterthought. If you scale images down later, you run into all kinds of problems, which you don't want to deal with especially a day before the submission deadline. Produce figures in (at least roughly) the right size from the very beginning with whatever tool you use, e.g. in Python MatPlotLib use `plt.rcParams['font.size']=10`

and `plt.rcParams['lines.linewidth']=1`, crop all white space `plt.savefig('picture.pdf',bbox_inches='tight',pad_inches=0)`, and of course produce vector graphics (svg,pdf) not pixel graphics (gif,png) and never ever jpg. Once everything essentially fits as it should, you could wrap it into some scaling command for *mild* adjustments for a perfect fit.

4 Revising/Proof-Reading/Cleaning Papers

Before submitting a paper, in particular the final version, be sure to properly polish it. In particular:

- Clean the LaTeX of the final version.
- Check for global consistency of style and notation
- Add Acknowledgements
- Proof-read the paper

Details follow:

Cleaning the LaTeX.

- *Preprints:* www.arXiv.org is a preprint server where many scientists upload their Technical Reports (TR). Reports are non-removable, so the same care/polishing should be applied as for published papers.
- *Cleaning:* Additionally, since the LaTeX becomes available, clean the LaTeX from all private comments ('%') you don't want to become public and switches `\if\else\fi` etc.
- *Danger:* Cleaning is a dangerous process. Keep the LaTeX compilable at all times and check that (the length of) the generated pdf is the same. If the length changed even a single line, you did something wrong (uncommented a comment, kept/deleted the wrong if-then-else path, ...).
- *Branching:* Only branch the LaTeX and clean it in the last minute. Continue with the uncleaned original document (not the cleaned) as your master document for further edits
- *Documentclass:* Further, no publisher style file should be used, but plain 'article', for TR for copyright reasons.
- *When?* A good time to upload the TR, is after you have submitted it to some conference or journal, received feedback, and revised it accordingly. This saves you from the potential embarrassment of uploading (irrevocably) something erroneous.
- *Template:* If you're not a LaTeX wizard, use http://www.hutter1.net/lattt.tex (this file) as a template and adapt it to your needs.
- *Switches:* It can save quite some redundant work by keeping the TR and journal/conference version in one file using `\ifconf\else\fi` or creating a `\newif\iftecrep` switch.

- *Journals:* For instance, journals and some conferences also ask for the LaTeX, so some cleaning is appropriate there too.
- *Packages:* Ideally the TR version for arXiv is as little dependent on external files as possible (packages, style-files, bibs, figures) especially exotic ones.
- *Macros:* If you have excessive lists of unused macros, maybe trim them too.
- *Includes:* Unless you paper is very long, integrate all `\include *.tex` files into the main LaTeX file.
- *Bibliography:* Integrate the `*.bbl` file into the LaTeX file (you should not upload your whole bibliography file). At the very least use bibAuxExtract.
- *Further:* See the arXiv pages for paper submissions and additional recommendations.
- *arXiv cleaner:* Google provides an arXiv cleaner in Python https://github.com/google-research/arxiv-latex-cleaner/, but it doesn't handle `\ifownswitch \else \fi`. It only handles `\iffalse \fi` correctly. I have a fragile extension which I could share.

Acknowledgements.

- It is important not to forget to acknowledge any grant agency that possibly supported the work.
- The best way not to forget is to add a flag `\newif\iffinal\finalfalse` and add the acknowledgement now in the current drafts and your template for future papers, and set `\finaltrue` in the final version.
- For instance `\iffinal\footnote{This work was [in parts] supported by ARC grant DP150104590}\fi` on the first page, or `\iffinal\paragraph{Acknowledgements}` We thank Name, Name, and Name for valuable feedback on earlier drafts and the Australian Research Council for support under grant DP150104590. `\fi` before the references.

Proof-reading. Proof-reading your own paper (too often) can be tiring, so should be done sparingly, but when done, then it should be done properly. Use LaTeX-Diff http://www.ctan.org/pkg/latexdiff for proof-reading revised versions, especially when asking a fellow student or your supervisor! If you experience problems, Exclude the hyperref package and replace `\DIFadd{\intertext` with `\intertext{\DIFadd`.

Times when careful (symbol by symbol) proof-reading is crucial are:

- When first submitting a paper
- When submitting the final version of an accepted paper (most important)
- When checking the proofs you get from the copy editor: This applies mostly only for backward journals that allow copy-editors to mess around with the LaTeX and introduce more errors than they remove. I had cases where they removed a 'non' in the abstract negating the key contribution of the article, or reformatting of the equations making them wrong and meaningless. What to check: Everything (see other items).

- Additionally check for correct English and for correct mathematics. Let the paper at least once proof-read by a colleague or student who is able and willing to read the paper on a symbol, word, and sentences level.

5 General Writing Advice

There is plenty of systematic and professional technical writing advice out there. The (1998) 'Handbook of Writing for the Mathematical Sciences' by Nicholas Higham [Hig98] is a fantastic resource with plenty of further references. You should read this before writing your first report or paper. The (2014) 'Writing for Computer Science' book by Zobel [Zob15] is also good.

Below is a somewhat random collection of tips and references for writing good scientific papers and related issues, somewhat tailored towards computer science and machine learning. Some advice you won't find anywhere else, and some you may disagree with, so there's a justification Section 6 for some.

Meta guidelines.

- *Advice:* Read and follow advice in http://www.hutter1.net/lattt.pdf.
- *Template:* Familiarize yourself with and adapt the LaTeX template http://www.hutter1.net/lattt.tex
- *Check:* Once you believe your paper is finished, read "Items to check" below.
- *Guidelines* are not rules and can be broken, but only if (a) you understand the motivation behind a guideline and (b) you have better arguments to go against them and (c) your co-authors agree.

General advice.

- *LaTeX:* Use LaTeX, not a (docx/odt) Wysiwyg editor, in particular if your paper contains formulas (see Section 2).
- *IDE:* Overleaf is a painless (simple) browser-based collaborate LaTeX editor. VSCode has a great 'LaTeX-Workshop' plugin, and a Github plug-in for professional versioning and collaboration. There are also specialized LaTeX editors (see Section 2).
- *LLMs:* If you use Large Language Models (LLMs) for editing your paper, you need to be extremely carefully check every diff, as if you believed an adversary tries to ruin your paper. The same holds if you let the LLM write new paragraphs. By now, LLMs are such good word/math-smiths that noticing snuck-in nonsense requires diligent proof-reading. This (as of August 2025) still happens regularly in mathematical proofs, and occasionally in text.
- *Documentation:* LaTeX files need to be well-documented, like good code (esp. why a particular choice has been made). For instance, why has a certain part been commented out (with `%` or better `\iffalse\fi`) and what it is about. Otherwise you lose track what is garbage and what is potentially still useful and what is just duplicates/copy-paste, and your co-authors will be confused.

- *Branching:* Never branch a paper, except one day before the final deadline of the final to-be-published paper. (see §compiler switches and §compression for details).
- *Theorems & other environments:* Like every paragraph, so also every theorem, definition, remark, example, etc. should have a name/title, for essentially the same reasons.
- *Proof-reading:* Produce quality, proof-read your own text at least 3 times (immediately, the next day, much later again, in LaTeX as well as pdf). It's not the role of anyone else (co-authors, reviewers, supervisors) to fix your sloppy work.
- *Errors:* Fix ALL bugs reported by proof-readers reliably. If you do not understand a comment or disagree, leave a todo in the LaTeX unless you're sure to skip.
- *ToDos:* Unless you are single author, mark todo's from whom to whom, e.g. `\todo{MH2MH:...}` and as done.
- *Deletions:* Do *not* delete text written by co-authors. Put major edits into `\iftrue <new> \else <old> \fi`. Only the authors should delete their own writing (including todos).
- *Spell-checking:* Use an instant spell-checker that marks wrong words already while writing.
- *English:* For non-native speakers (possibly even for natives) the CCC Guide to Grammar (see ref. below) is very enjoyable, and useful, in particular, since rules are often "justified" and not regarded as god-given untouchable "truths". Strunk and White's (2000 4th edn) Elements of Style is a classic.
- *References:* Read the "references" below before you write your first paper, and possibly again after you wrote a couple of papers.
- *Write modular:* Every paragraph should be between 4 lines and 1/2 a page. Every paragraph must have a (potentially invisible) heading `[%]\paragraph{heading.}` (no exceptions!). If you can't find a concise descriptive heading, then something is wrong with the paragraph. It also helps your co-authors to quickly find things.
- *Paper/(sub)section/paragraph headings:* Headings should be descriptive and at most 10-12 words (no generic 'Experiments', 'Methods', ...). 'and' in a heading indicates a potential problem (see 'Justifications' below).
- *Justifications:* Don't waste time writing justifications to mailing lists or blogs or co-authors or reviewers. Write a (short) paragraph for (later) use in your paper (and email a copy).
- *Planning:* First write down a table of contents (= todo list) of your paper down to single fact (paragraph or sentence) level. Only thereafter wrap it into nice English (this is particularly time-saving for non-native speakers).
- *Focus:* Mark items as core vs nice vs optional. Only once the core has been written and the paper could be published as is, start adding nice and optional content if you are inclined to do so. If not, add an outlook section of all that

could have been written or done.
- *Target:* Don't write papers *for* conferences or *to* deadlines. Write papers *about* a scientific idea or investigation, possibly for a particular audience, succinct but with all necessary details. One day before the conference deadline, branch off a version and shorten it for the conference (or use `\if\fi` in TeX for creating versions).
- *No copy-paste:* If it is really necessary to explain something multiple times (e.g. abstract, intro, main, conclusion), do so in different ways or from a different angle or different degree of formality, rather than repeating yourself.

Parts.

- *Keywords:* Make a useful and relevant list of general and specific keywords
- *Title:* Form various tentative titles out of the most important keywords. Describe content and purpose of work preferably in plain (non-technical) terms.
- *Abstract:* Summary, topic, problem, objective, methodology, main findings, conclusions, keywords. Emphasis is on main findings and conclusions. <=150 words.
- *Table of contents:* A 15+ page paper should have a TOC.
- *Introduction:* See below
- *Conclusion:* See below.
- *Cross-reference* theorems, sections, remarks, ... generously.
- *List of notation* is a must for math-heavy papers. It also helps yourself to develop good and consistent notation. See below.
- *Index:* Any scientific book needs an index. A poor index indicates the author does not care about its own book. Clean the index from similar/double entries. Simplest is to convert everything to a canonical form: all lower case (except acronyms and proper names) and singular. Use noun instead of adjective or verb-form in case of conflict or doubt. Nearly always use `\indxs{hello}{world}` and avoid 3-level indices.

Structural / Layout.

- *Aesthetics* matters even in scientific papers! It is a piece of art. If you lack a sense of beauty or elegance, read about it and mimic it to the best of your abilities.
- *Style* should ideally be somewhat uniform and consistent. This includes length of chapters (∼20p), sections (∼5p), subsections (∼2p), named paragraphs (∼1/2p), unnamed paragraphs (4-10 lines).
- *Hierarchy:* Generally flatten nested or deep structures where reasonably possible. A branching factor of ∼5 is good. Unless your paper/chapter is over 40 pages, avoid subsubsections. If you have only two (sub)subsections in a

(sub)section, consider splitting them into two separate (sub)sections. A single (sub)subsection in a (sub)section is a hard No. If you have subsections, all material in the section needs to be in subsections, except for a short 'abstract/summary/intro'.

- *Capitalization:* Title and (sub)section headers in TitleCaps (all words start with capital letters except short words.). Standard capitalization for everything else (paragraph, ...). Names of theorems: all lower case including first word (except Names).
- *Floats:* Use `[htbp]` only for figures less than 1/3 page or not at all. Only figures without caption should be forced [H]ere, but then just remove the figure environment altogether. Never use [h!].
- *Footnotes:* Use very sparingly and short footnotes only. Integrate those relevant and possible into the text, remove uninteresting ones. Anecdotal and off-topic footnotes are fine.
- *Cross-reference* theorems, sections, remarks, ... generously.

Math.

> *"There is nothing that can be said by mathematical symbols and relations which cannot also be said by words. The converse, however, is false. Much that can be and is said by words cannot be put into equations, because it is nonsense."* — *Clifford Ambrose Truesdell (1966)*

- *Must read* Higham (1998) 'Handbook of Mathematical Writing'
- *Math↔words:* Explain every math formula in words. Formalize all used concepts mathematically unless they are tangential to the paper (cf. quote above).
- *Notation:* Always use meaningful and consistent notation, even for simple indices. Never use the same symbol for two different things within the same paper, not even indices.
- *Numbering:* Use one and the same counter to number all environments (theorem/definition/lemma/...) This makes it easier to find them when reading the paper. llncs documentclass offers [envcountsame].
- *Punctuation:* Don't put punctuation at the end of a displayed equation. It is more confusing than helpful.
- *Footnotes:* Never put footnotes on math/equations.
- Use `equation` (`$$ $$` or `\[\]`) for single-line (un)numbered equations, because (only) they use smart vertical spacing. Use `eqnarray` and `align` (only) for multi-line equations.
- *Empty lines:* Do *not* use empty lines before or after equations unless you deliberately want to start a new paragraph.
- *Common knowledge:* Rule of thumb: if there's a wikipedia page, it may be ok to forgo to define an object or provide a reference (e.g. algebraic group, probabilistic Turing machine, NP-Hard).

- *Algorithms:* Algorithms require specification of Input and *Output!*. Use Require/Effect if Input/Output seem inappropriate, like global variables and side-effects.
- *End-of-environment marker:* (Automatically) add `\qed` ■ and `\eoe` ♦ and `\eor` ● at the end of every proof/example/remark, unless they are boxed or in a different font. Otherwise the reader does not know where the environment ends.
- *Book:* No "home-made" proofs in textbooks (with 99% prob. they are worse or wrong): Search for the neatest suitable existing proof and adapt. If you still think your home-made proof is superior, use it but double check with an expert it is indeed correct.
- *Inlined equations:* Avoid inlined math expressions $a=b$, $c=d$ separated by comma. Try to squeeze some word between, e.g. 'and' or 'it holds' or 'we have' or else.
- *Displayed formulas:* Avoid "hanging" displayed formulas such as '$a+b$' in favor of (ideally true) *statements* such as $c:=a+b$ or $x\leq a+b$.
- *Brackets:* Use square brackets E[] and Var[] and P[event] consistently, but $p(x)$ for mass function and density.
- *Sequences:* Denote sequences with (x_t), *not* $\{x_t\}$, the latter are (unsorted) (multi)sets or families.
- *Max:* Use $\max\{a\}$ not $\max(a)$ nor $\max a$.
- *Displayed 'and':* Use `~~~\text{and}~~~` to separate equations in displayed text.
- *Displayed spacing:* Use `~=~` and `~\leq~` etc. for displayed equations.
- *Propositions* are Theorems that have easy proofs.
- *Lemmas* are required for for proving Theorems but are otherwise deemed uninteresting.
- *Corollaries* easily follow from a theorem/lemma/proven result, usually not from a Definition.
- *Sharp inference:* Never write '$a>0$ implies $b>0$' in a proof if $b=a$ (or $b\geq a$) unless you also explicitly point out $b=a$ (or $b\geq a$). Write '$b=a>0$' or '$0<a\leq b$' instead.
- *Unwanted new lines:* Don't use \\ in the last line of an equation/table/... It creates an empty line.
- *Auto(un)numbered equations:* Package `autonum` can automatically suppress numbering of unreferenced equations, but important equations should be numbered even if not referenced (use `\ignoreoutput{\ref{eq:label}}`).
- *String↔sequence:* Preferably use 'sequence' for infinite length and 'string' for finite length.
- *Generic↔specific:* If you define, say 'weak X' and 'strong X', do *not* give 'X' a specific meaning, such as 'X' w/o qualifier is (short for) 'asymptotic X'. Use 'X' *only* to generically refer to any notion of 'X', be it weak or strong or asymptotic or else. Same with symbols: If you define A_+ and $A_{\mathbb{R}}$ and A_{whatever},

do not give A a specific definition. Only use it to generically refer to any of the $A_{...}$.

English.

- *Simple sentences:* Short and easy (nearly staccato) sentences. No German, as this one, slide ins.
- *Which hunt:* Use 'that' for necessary clauses instead of a no-comma 'which' unless it sounds awful: Example: 'The car that broke' not 'The car which broke'.
- *Parenthesis:* In case of doubt use ", blah, " instead of (blah) in text. Avoid nested parenthesis.
- *Hyphenation:* Learn the hyphenation rules: The major one is 'a b c' means 'a (b c)', and 'a-b c' means '(a b) c'. Don't use 'a b-c' except 'b-c' is an established 'word'.
- *Future tense:* Don't use "we will" but simply "we" everywhere for referring to text further down. Use 'we will' only if you literally refer to the future. e.g. "we will write another paper".
- *Such as↔like:* Use 'such as' (if example) instead of 'like' (if comparison) consistent with English rules.
- *As,since,because:* Don't use 'as' if other words are available like 'since or because'.
- *We↔passive:* Use "we define" or "our definition" if it is our version;
  preferably use "is defined as" or "this definition" if it is standard.
- *Colon:* After ':' upper (lower) case if what follows is (not) a full sentence.
- *We↔you:* As a rule of thumb, use 'we' instead of 'you'.
- *Pronouns:* Don't use 'they' for singular. Use 'it' for agent. Consistently use 'he' (e.g. for player 1) and 'she' (e.g. for player 2) as in Osborne & Rubinstein (1994) book to ease understanding, etc.
- *Specificity:* Use most specific word fitting context:
  Bayes $\subset$ statistics $\subset$ math $\subset$ theory $\subset$ science $\subset$ entity
- *Word repetition:* Avoid word repetitions, e.g. "From Definition 7 of a measure ..."
  rather than "From the definition of a measure (Definition 7) ..."
- *Hence&al.:* "Hence/thence/whence" mean "from here/there/where"
  or "because of this/that/which"
  (cf. "hither/thither/whither" meaning "to here/there/where")
- *First word:* Every sentence should start with an English word. Don't start a sentence with a number or math or (alpha)numerical reference [ABC34] etc. If putting [ABC34] later is awkward, use 'Name et al. [ABC34] showed that' ...
- *Contractions:* Minimize the use contractions in formal writing, e.g. use '* not' instead of *n't.

- *Homophones&more:* Check for proves vs proofs, lose vs loose, an vs a, beliefs vs believes, than vs then, it's vs its, like vs such as, which vs that, ...
- *MDPs↔pdf's:* Preferably pluralize upper-case acronyms with *s* (MDPs) but lower-case acronyms with *'s* (pdf's)

Miscellaneous.

- *Highlights:* Ideally a paper can be understood by only reading the Table and Figure captions and Definitions and Theorems, meaning they are sufficiently self-contained and explanatory.
- *Conventions:* If you are not sure about a particular (minor) convention, search (in files or internet) for both versions, and adopt the majority (GoogleSearch→Tools→Count). If it is major, unify and possibly use a macro, which makes it is easier to change later.
- *Precise references:* Give precise references `\cite[Thm.1.2.3.]{Paper}` or [p.123] or [Sec.1.2.3] where possible.
- *History&reference:* Book: Keep an 'Annotated Bibliography' or 'History&Reference' section dense and compact. At least one reference per sentence is a good rule of thumb.

Introduction. (from http://www.clet.ait.ac.th/EL21OPEN.HTM)

- *Coverage:* After reading an introduction, the reader should be able to answer most of the following questions: (background, rationale, problem statement, objectives, scope, limitations, assumptions)
- *Background:* What is the context of this problem? In what situation or environment can this problem be observed?
- *Rationale:* Why is this research important? Who will benefit? Why do we need to know this? Why does this situation, method, model or piece of equipment need to be improved?
- *Problem statement:* What is it we don't know? What is the gap in our knowledge this research will fill? What needs to be improved?
- *Objectives:* What steps will the researcher take to try and fill this gap or improve the situation? How is the problem solved?
- *Scope:* Is there any aspect of the problem the researcher will not discuss? Is the study limited to a specific geographical area or to only certain aspects of the situation?
- *Limitations:* Is there any factor, condition or circumstance that prevents the researcher from achieving all his/her objectives?
- *Assumptions:* In considering his/her method, model, formulation or approach, does the researcher take certain conditions, states, requirements for granted? Are there certain fundamental conditions or states the researcher takes to be true?

- *Research problem, solution, background*
  (adapted from http://www.cs.cmu.edu/ jrs/sins.html)
- *Get to the point:* Either the reader is an expert who does not need grandmothering - or - he is a beginner who will neither understand all the terms nor the rest of the paper anyway.
- *Reference to sections:* Integrate the references to sections in the description in the introduction of your paper, rather than to append a list of: In Section .. we do ..
- *:* You should write every sentence as if the reviewer/reader will toss your paper if the first half of the sentence isn't interesting.

Conclusions. (adapted from http://www.cs.cmu.edu/ jrs/sins.html)

- *Conclusion≠summary:* A lot of people have picked up the misconception that they should conclude their paper with a summary. A conclusion is about the implications of what the reader has learned. Of course, a conclusion is also an excellent place for conjectures, wish lists, and open problems.
- *Larger context:* Embed your specific work in larger context in the conclusions. Just the opposite way as for the introduction, where you start with the general context.
- *Limitations&extensions:* State limitations and possible extensions of the work.

Heilmeier-like summary. Questions that should be answered in the abstract & introduction|conclusion of a project proposals, but most apply to papers as well.

- What is the problem you are tackling?
- What is the current state-of-the-art?
- What's new in your approach?
- Who should/might care about your work?
- What are the risks and payoffs of your approach?
- What have you already accomplished?
- What is your plan for success?

Items to check. (origin: for ICML authors and reviewers) Does your paper get the best scores in all of the below categories? Improve your paper if not!

1. *Scope:* Is the paper relevant to the venue?
2. *Novelty:* Does the material constitute a novel unobvious contribution to the field or if it is tutorial in nature, does it review the field appropriately?
3. *Usefulness:* Are the methods, theories, and/or conclusions particularly useful (usefulness should be well supported by results)?
4. *Sanity:* Is the paper technically sound (good methodology, correct proofs, accurate and sufficient result analysis)?
5. *Quantity:* Does the paper contain enough interesting material?

6. *Reproducibility:* Are the methods introduced or considered sufficiently described to be implemented and/or to reproduce the results?
7. *Demonstration:* Has the efficiency, advantages, and/or drawbacks of the methods introduced or considered been sufficiently and convincingly demonstrated theoretically and/or experimentally?
8. *Comparison:* Has a sufficient method comparison been performed?
9. *Completeness:* Is the paper self contained, rather than referring to other publications extensively?
10. *Take-aways:* Does the paper clearly state its objectives (in the title, abstract, and introduction) and delivers them (in the abstract, body of the text, and conclusion)?
11. *Bibliography:* Is the background properly described in the introduction and/or discussion, with an adequate bibliography?
12. *Outlook:* Are the results critically analyzed and further research directions outlined in a discussion or conclusion section?
13. *Data availability:* Are the data made available to other researchers?
14. *Code availability:* Is the implementation made available to other researchers?
15. *Readability:* Is the paper easily readable for experts interested in this topic?
16. *Notations:* Is the used notation standard and consistent
17. *Figures:* Is the paper well and sufficiently illustrated by figures?
18. *Formalism:* Are the methods clearly formalized by a step by step procedure (e.g. algorithm pseudo-code or flow charts provided)?
19. *Density:* Is the length appropriate, relative to the contents?
20. *Language:* Is the English satisfactory?

References & Sources. (roughly in decreasing order of relevance)

- Comprehensive advice for PhD students by Toby Walsh http://www.cse.unsw.edu.au/ tw/phd.html
- Higham (1998) Handbook of Writing for the Mathematical Sciences
- Senda (2002) Title Generation
- Langley (1999) Crafting Papers on Machine Learning
- Parberry (1994) A Guide for New Referees in Theoretical Computer Science
- Parberry (1993) How to Present a Paper in Theoretical Computer Science: A Speaker's Guide for Students
- Kevin Korb (1998) Research Writing in Computer Science
- Richard Nordquist (2014) Guide to Grammar and Writing: https://web.archive.org/web/20140209052714/http://grammar.about.com/
- Zobel (2004) Writing for computer science, Springer-Book.
- Guidebook for Writing a Thesis: http://www.ait.ac.th/education/LanguageCenter/ait-writing-services/guide-book/
- How to read a research paper: http://www.cs.brandeis.edu/ cs227b/papers/introduction/howToRead.txt

- Three Sins of Authors in Computer Science and Math
  http://www.cs.cmu.edu/ jrs/sins.html
- Writing up Research Introductions http://www.clet.ait.ac.th/EL21OPEN.HTM
- BibTeX bibliography sources:
  Millions of (messy) bibs: http://scholar.google.com/
  7 million (messy) CS bibs (till 2023):
  https://web.archive.org/web/20230721051248/http://liinwww.ira.uka.de/bibliography/
  My 1000 (tidied) AI bibs: http://www.hutter1.net/official/mhother.bib
  Own publications bibs: http://www.hutter1.net/official/bib.htm

More / not so good / marginally related / classics.

- Strunk & White (2000) The Elements of Style (4th Edition)
- Robert Peters (1997) Getting What You Came For: The Smart Student's Guide to Earning an M.A. or a Ph.D.
- JMLG is an irony (but unfortunately partly true) how to survive in the research environment http://www.jmlg.org/guides/guides_for_the_perplexed.htm
- Scrutiny of the introduction: http://sepwww.stanford.edu/sep/prof/Intro.html
- Scrutiny of the Abstract: http://sepwww.stanford.edu/sep/prof/abscrut.html
- How to Start Research: http://ceng.usc.edu/ helmy/research-start.html
- How to write a 2 page initial project proposal:
  http://nile.cise.ufl.edu/ee499/proposal-outline.htm
- How to write a 10 page project proposal:
  http://nile.usc.edu/ee499/report-outline.htm
- Write to Express, Not to Impress:
  https://medium.com/swlh/write-to-express-not-to-impress-465d628f39fe

Presentations / talks / slides.

- *Beamer:* Use package **beamer** for technical slides containing equations.
  It's widely used and very powerful.
- *Format:* Use Letter or A4 or 4:3 format, not 16:9.
- *Tailor* your talk to the audience (collaborator↔expert↔specialist↔scientist↔public).
- *Properly define all* mathematical symbols including spaces (e.g. $x \in \{0,1\}^*$ or $a_t \in \mathcal{A}$ where $\mathcal{A}$ is a finite set. $a_{1:t} = a_1 a_2 .... a_t$, ...).
- *Font size:* Never reduce font size to fit stuff on the slide (reduce amount or split slide).
- *Be concise:* Use one-line short bullet items phrases, not full multi-line sentences.
- *Color:* Color or bold-face key terms (ideally in a consistent color-scheme, e.g. red for definitions, green for results, ...).
- *Examples:* Add as many concrete examples and/or diagrams as possibly illustrating abstract definitions / concepts / results.
- *Motivation:* Spend most of your time on defining and motivating the problem.

- *A Guide for Navigating the ML Conference Scene* by Khimya Khetarpal and Cheng Soon Ong https://virtual.aistats.org/Conferences/2025/BlogPost_1
- *Oral Presentation Advice* by Mark D. Hill http://pages.cs.wisc.edu/ markhill/conference-talk.html
- *Giving an Academic Talk* by Jonathan Shewchuk http://www.cs.berkeley.edu/ jrs/speaking.html
- *The Short Talk* by Charles Van Loan http://www.cs.cornell.edu/cv/shorttalk.htm
- *Confessions of a Public Speaker* (2009) by Scott Berkum

6 Some Justifications

Guidelines and rules require justification. Guidelines can be broken but only if (a) you understand the motivation behind a guideline and (b) you have better arguments to go against them and (c) your co-authors agree. All guidelines/rules in this document have justifications, but writing them all down would fill a book. Most of them should be pretty obvious anyway. Below are only a few:

Table of contents. There seems to be no doubt about the use of a table of contents (TOC) in a book. You can get a feeling of the contents before reading the book and you can easily find appropriate chapters or sections. A TOC in an article serves the same purpose. Sure, if the article is very short or not even has sections, a TOC makes no sense. A TOC should not disturb anyone and the additional space is marginal, so in case of doubt, use a TOC. A possible rule of thumb may be that a TOC should have more then 10 entries or the paper more than 15 pages. The TOC is easy to remove if the publisher does not want it, but you can keep it in the TR version. Some (physics) journals have paper TOCs.

Named paragraphs. Write modular: Every paragraph should be between 4 lines and $\frac{1}{2}$ a page. Give every(!) paragraph a heading, even if made invisible or deleted later. If you can't find a concise descriptive heading, then something is wrong with the paragraph or the structure of your paper. It also helps readers and especially your co-authors during writing to more quickly find things.

This is the best way I know of structuring a document. It is good for writing discipline, reading and searching. Be sure that there are proper transitions between the paragraphs; i.e. they fit together and can be read ignoring the headings.

If you end up with an 'and' in a paragraph (also applies to section&paper) title, you *may* have lumped two topics together, which should be covered in separate paragraphs (sections&papers) whenever possible.

If the publisher complains, telling them to regard the bold-face as emphasizing something, not as a heading, such as 'Theorem', 'Proof', ..., can help. Then it is not in conflict with the journal style.

Punctuation after displayed equations. There are linguistic rules for punctuation in text. These rules are applied by editors, probably without much thought,

also to mathematical equations. Further, there seems to be a tendency to enforce linguistic rules, and regarding a violation as inherently wrong. The problem is that in the best case, an orphan punctuation after a complex equation looks like dirt, in the worst case it confuses because it looks like a multiplication and you expect a continuation of the formula in the next line. I find it silly to rescue linguistic rules in math environments in favor over clarity. An orphan punctuation never helped me to understand (parse) an article better, and that is the primary intention of the punctuation. Also, `$$e^{i\pi+1=0}.$$` makes logically no sense. The punctuation is part of the sentence, not part of the math. So if you are that pedantic about punctuation, you should also pedantically put the punctuation outside the math environment (`$$e^{i\pi+1=0}$$.`) where it belongs, and enjoy the result. It seems that the French agree(d) with my view (see 1993 book by Olivier Faugeras).

Theorem numbering. While it seems more natural to have separate counters for theorems, definitions, corollaries, etc, it is better if they are enumerated jointly with one counter, since this makes it *much* easier to trace them later and avoids confusing e.g. Theorem 3 with Corollary 3. In TRs this can be ensured by defining
`\newtheorem{theorem}{Theorem}`
`\newtheorem{corollary}[theorem]{Corollary}`
`\newtheorem{definition}[theorem]{Definition}` etc.
Some style-files have options, e.g. in Springer's llncs this can be achieved by
`\documentclass[envcountsame]{llncs}`

Proofs in appendix versus main text. The decision of whether to put proofs in the flow of the paper or in an appendix is a difficult one. A good criterion is whether it's the proof (techniques) which make the paper interesting or the discussion of the result. Here some more pros and cons:
Pro Appendix

- Most readers are casual and skip the proofs anyway.
  Only the few very interested ones will care to work through the proof.
- A paper littered with proofs looks tough and drives readers away.
- Proofs disrupt the flow of the discussion.
- If the paper has survey character, proving some new results in the main text seems odd.
- The theorem is beautiful but the proof is ugly

Pro Main Text

- The reader does not have to jump forward and backward in order to read proofs.
- The reader is more encouraged to read the proofs.
- The paper or theorem can only or better be understood when reading the proof.
- In mathematics the proof is typically the core of the research paper.

7 Miscellaneous

Every paragraph that is perfectly fine but just doesn't fit anywhere at the moment put here. When the paper is essentially done, try to integrate them into the main paper. If they really don't fit anywhere but are too important to delete, leave them here.

Theorem 1 (name of theorem) *Like every paragraph, so also every theorem, definition, remark, example, etc. should have a name, for essentially the same reasons.*

- x

8 Extended

Many things get started but never get to a finished or polished enough state that they can be integrated into the paper. Put them here and return to them later or let them rot here forever.

9 Discussion

Summary. *[[ToDo:]]*

Outlook. *[[ToDo:]]*

Conclusion. *[[ToDo:]]*

References

- Definitely use BibTeX for bibliographic references. Maintain a single bib file.
- Use dummy fields (e.g. mysummary = "blabla ...") for annotations.
- More convenient for managing many references is to use dedicated software and export to BibTeX. Zotero is great (Customizable, Browser PlugIn, Notes, Tags, Magic Wand, Better BibTeX extension, ...). Mendeley seems very good too.
- Use `\bibliographystyle{alpha}`. This is better than (arbitrary) plain numbers, since it allows expert readers to instantly guess or know which paper is referenced, but is more compact than the 'Author Name (year)' citation style. Publishers often accept alpha, even if they want/prefer other styles. Just try it.
- Try your best to make citations as complete as possible. Of course if some info does not (yet) exist, you leave it partial, but don't forget to complete it (if possible) in the final/proof version.

- Keep your bib-file tidy and internally consistent, rather than dump bibtex entries grabbed from various sources in various styles in random order in your bib file.
- In particular have a unique system for your bibkey, e.g. last name of first author : 2-digit year + optional differentiator (Alchin:06 or Hutter:24uaibook2), ideally keep them sorted, so as to avoid duplicate entries.
- Use bibAuxExtract if you need to create a bib-file that only contains the bibtex entries used in your paper, e.g. if you do not want to share your whole bib file with others.
- For publication (esp. on arXiv) it is better to integrate the generated bbl file into your final LaTeX source, as done below, or at the very least use bibAuxExtract.
- BibTeX bibliography sources:
  Millions of (messy) bibs: http://scholar.google.com/
  7 million (messy) CS bibs (till 2023):
  https://web.archive.org/web/20230721051248/http://liinwww.ira.uka.de/bibliography/
  My 1000 (tidied) AI bibs: http://www.hutter1.net/official/mhother.bib
  Own publications bibs: http://www.hutter1.net/official/bib.htm
- The default list of references is often very spacious and could be compacted by replacing `\begin{thebibliography}{VNHSU11}` in the generated bbl file with `\begin{thebibliography}{ABCD}\parskip=0ex` or so. I have not figured out a more automatic way of doing the latter. Help is welcome.

[AHB03] P. W. Abrahams, K. A. Hargreaves, and K. Berry. *TEXfor the Impatient*. Addison Wesley, 2003.

[DHD20] W. Duivesteijn, S. Hess, and X. Du. How to cheat the page limit. *WIREs Data Mining and Knowledge Discovery*, 10(3):e1361, 2020.

[Eij92] V. Eijkhout. *TEXby Topic: A TEXnician's Reference*. Addison-Wesley, 1992.

[GH97] D. F. Griffiths and D. J. Higham. *Learning LATEX*. Siam, Philadelphia, 1997.

[Hig98] N. J. Higham. *Handbook of Writing for the Mathematical Sciences*. Society for Industrial and Applied Mathematics, 2nd edition, 1998.

[Hut13] M. Hutter. Sparse adaptive Dirichlet-multinomial-like processes. *Journal of Machine Learning Research, W&CP: COLT*, 30:432–459, 2013.

[KD04] H. Kopka and P. W. Daly. *A Guide to LATEX*. Adison-Wesley, 4th edition, 2004.

[Zob15] J. Zobel. *Writing for Computer Science.*. Springer, 3rd edition, 2015.

Alternative Author (Name) Citation Style (often required by journals).

```
\usepackage{natbib}
\bibliographystyle{plainnat}
\bibpunct{(}{)}{;}{a}{,}{,}
```

A Macros

Several general macros are defined in the preamble of this document. Some are used in this document. Some examples and a list with explanations is below.

Itemized lists.

- `\parskip=0ex\parsep=0ex\itemsep=0ex` can be used to make itemized lists more compact.

* Item 2 of this list has a customized label.

- This is more compact list has been created by the `\blob` command.
- LaTeX as well as display are more compact.
- Can be used for single-lined items,
  but `\tab` can be used for items spanning multiple lines if broken by hand.

Displayed equations. `\beq\eeq` (`\beqn\eeqn`) can be used to generate

$$\left.\begin{matrix}\text{numbered}\\ \text{equation}\end{matrix}\right\}\ \text{AIXI}: \quad a_k \;:=\; \arg\max_{a_k}\sum_{o_k r_k}\ldots\max_{a_m}\sum_{o_m r_m}[r_k+\ldots+r_m]\sum_{q\,:\,U(q,a_1..a_m)=o_1r_1..o_mr_m}2^{-\ell(q)} \tag{1}$$

`\bqa\eqa` (`\bqan\eqan`) can be used to generate

$$\begin{aligned}\text{an (un)numbered equation array:}\qquad & \textit{Euler:}\quad e^{i\pi}+1 \stackrel{!}{=} 0 \qquad (2)\\ \text{with second and more aligned lines:}\qquad & \textit{Stokes:}\quad \int_{\partial\Omega}\omega \underset{\uparrow}{=} \int_{\Omega} d\omega\end{aligned}$$

beautiful equation

Example 2 (name) Example generated with macro `\fexample` ♦

Macro	Explanation
`\ifprivate`	print private comments
`\ifconf`	conference or journal versus techreport style
`\ifshort`	short (e.g. conference) version
`\ifcompress`	compressed style
`\ifuc`	parts under construction
`\iffinal`	final versus first submission
`\beq\eeq`	numbered equation
`\beqn\eeqn`	unnumbered equation
`\bqa\eqa`	numbered equation array
`\bqan\eqan`	unnumbered equation array
`\begin{theorem}`	Theorem environment
`\begin{corollary}`	Corollary environment

`\begin{lemma}`	Lemma environment
`\begin{definition}`	Definition environment
`\begin{example}`	Example environment
`\begin{proposition}`	Proposition environment
`\begin{principle}`	Principle environment
`\begin{keywords}`	Keyword environment
`\begin{proof}`	Proof environment
`\idx#1`	put index word also in text
`\indxs#1#2`	symmetric indexing
`\fexample#1#2#3`	handmade example environment
`\paradot#1`	boldface paragraph with dot.
`\paranodot#1`	boldface paragraph
`\blob`	simple itemization without vertical spacing
`\tab`	indent item without blob
`\qmbox#1`	space text space in math mode
`\hrefurl#1`	hyperlinked,underlined,blue url
`\phantom#1`	make #1 invisible but reserve space
`\ignoreoutput#1`	execute #1 but don't render it, i.e. not space
`\qmbox#1`	space text space in math mode

Symbol	**Macro**	**Explanation**
(1)	`\req`	reference to equation
$\xrightarrow{n\to\infty}$	`\toinfty#1`	limit
$\Leftrightarrow$	`\iffs`	if and only if
ε	`\eps`	small real > 0
ϵ	`\epstr`	empty string
	`\nq`	negative 1em space
[[ToDo:]]	`\todo#1`	todo's when `\privatetrue`
[[Note:]]	`\note#1`	notes when `\privatetrue`
■	`\qed`	end of proof
♢	`\eoe`	end of example
€̶	`\Euro`	Handmade Euro symbol
$\frac{\#1}{\#2}$	`\fr#1#2`	textstyle fraction
$^{\#1}/_{\#2}$	`\frs#1#2`	/ fraction
$\mathbb{R}$	`\SetR`	Set of Real numbers
$\mathbb{N}$	`\SetN`	Set of Natural numbers
$\mathbb{B}$	`\SetB`	Set {0,1}
$\mathbb{Z}$	`\SetZ`	Set of Integers
e	`\e`	natural e≐2.71828...
{0,1}	`\B`	boolean
$\mathbb{E}$	`\E`	Expectation
P	`\P`	Probability

$\log_2$	`\lb`	binary logarithm
$\boldsymbol{v}$	`\v`	boldface vector
$A^\top$	`\trp`	transpose of A
Γ	`\G`	frequently used Greek letters
α	`\a`	
δ	`\d`	
γ	`\g`	
σ	`\s`	
θ	`\t`	

B List of Notation

For every document, create a list of notation, even if removed in the final version, and best before starting to LaTeX your research. This gives yourself an overview of all notation you use, and makes it easier to improve your notation: to spot inconsistencies and to make notation more systematic. Never use the same symbol for two different things within the same paper, not even indices. Listing only paper-specific notation is sufficient. For PhD theses and books, you may even want to include standard notation as some of the notation below. Your co-authors, supervisors, reviewers, and if kept your readers, will be thankful.

Symbol	**Explanation**
$a/bc=a/(bc)$	while $a/b\cdot c=(a/b)\cdot c$ though we actually always bracket the latter
$i,j,k\in\mathbb{N}$	index for natural numbers
$\{i\!:\!j\}$	Set of integers from i to j (empty if $j<i$)
$[\![\text{bool}]\!]$	$=1$ if bool=True, $=0$ if bool=False
$\lvert\mathcal{X}\rvert\equiv\#\mathcal{X}$	size of set $\mathcal{X}$.
$\mathcal{A}$	finite alphabet like $\{a,...,z\}$ or ASCII or $\{0,1\}$.
$\mathcal{A}^n,\mathcal{A}^*,\mathcal{A}^\infty$	set of all (length n, finite, infinite) strings over alphabet $\mathcal{A}$
$x\in\{0,1\}^*$	finite binary string
$l=\ell(x)$	length of string x
$t,n\in\mathbb{N}$	time index, e.g. $x_{1:n}$ or $x_{<t}$
$x_{1:n}\in\mathcal{A}^n$	string of length n
$x_{<t}\in\mathcal{A}^{t-1}$	string of length $t-1$
$\cong,\lesssim,\gtrsim$	$=,\leq,\geq$ within additive logarithmic corrections
$\approx,\lessapprox,\gtrapprox$	approximately $=,\leq,\geq$ with unspecified precision
w.μ.p.1	with μ-probability 1
ε	small number >0

ϵ	empty string
$\mathbb{R}$,$\mathbb{N}$,...	set of real,natural numbers
e	natural e$\doteq$2.71828...
P	probability
$P(x) := P[X=x]$	probability that X is x
$\boldsymbol{v}$	vector
$\mathbb{E}$	expectation
$O()$,$\Theta()$	classical $O()$ notation
$O_P()$	stochastic O-notation
$\doteq$ e.g.	$z\doteq 1.96$, equal to within the number of displayed digits
$f(n)\lesssim g(n)$	means $f(n)\leq g(n)\cdot[1\pm O_P(1/\sqrt{n})]$, and similarly $\gtrsim$ and $\approx$
$:=$, $\equiv$, $\stackrel{!}{=}$	definition, equal by earlier definition, want it to be equal
■	end of proof
●	end of remark
♦	end of example & end of notation & end of paper

Good notation. Notation has to serve many needs: Overall it should make the paper as readable as possible.

My advice below is for mathematical outlets and readers who want to follow the details. For non-mathematical outlets and casual readers, a lighter (slightly sloppy) notation, hiding many details, is acceptable and often better.

Notation should as far as possible be,

- self-explanatory (using well-established conventions, suitable letters e.g. t,T,τ for time, ...)
- explicit (not hiding dependencies)
- succinct (one-letter variables, indices instead of arguments, ...)
- systematic (same font for same type of objects)
- grouping (e.g. a,b,c for letters; i,j,k for integer indices; x,y,z for variables; ...)
- minimal (as few as possible definitions)
- consistent (globally valid).
- strongly typed (no symbol, even indices, used for different "purposes").

Novel types of symbols or notation are fine, if they are superior to established notation (cf. Einstein's sum convention, Dirac's dagger, ...). Hiding dependencies makes formulas look nicer, but make them actually harder to understand. Mathematicians have a tendency of hiding dependencies, in contrast to physicists. This is one reason why math publications often look nicer, while physics publications are easier to understand.

C Index

Papers are rarely indexed, but books are, your PhD thesis should better be, and some contributions to edited books have to be. Use `\usepackage{makeidx}` and the index command `\index` and possibly short hands `\idx` and `\indxs` and

```
\addcontentsline{toc}{section}{Index}
\printindex
```

at the end of the document.