"TeX" Support Proposal

Lorenzo Hess

2025-03-27

Contents

1 Introduction

This document outlines ideas for a comprehensive approach to LaTeX advocacy, covering everything from pro-LaTeX talking points, arguments against word processors (WPs), solutions for group work in LaTeX, and LaTeX code snippets, including full preambles and examples. While any individual could make use of this advocacy material, a group or club would be ideal in terms of providing additional resources and labor. Such a group could be called "TeX Support", so I've used that in the title of this document [1].

2 What's Our Agenda?

We like LaTeX, and we think it's better for preparing documents, in most cases, than Word Processors (WPs). We're also pragmatic, and recognize that many people haven't used LaTeX or may have felt frustrated when they have. Many LaTeX tutorials indeed fail to enumerate and provide what we see as essential foundations for learning, appreciating, and enjoying LaTeX document preparation. For example, many tutorials characterize LaTeX's utility in terms of what it is good at, such as typesetting math or providing robust label and reference management. They fail, however, to provide a comprehensive feature comparison between LaTeX and WPs that would make clear LaTeX's superiority. Tutorials also generally fail to fully compare the LaTeX and WP mindsets, and thus fail to help users realize how WPs hinder writing. Finally, tutorials fail to provide essential tools that make LaTeX editing less tedious, from code templates like preambles to software like Detexify.

We have two general goals: 1) present a comparative, serious, and evidence-based argument for when, and why, individuals and groups should use LaTeX instead of WPs; and 2) to provide code templates, code snippets, and guides for individuals and groups to not only get started with LaTeX, but use it to its full potential.

We're not here to evangelize. WPs do have certain advantages over LaTeX(see the feature comparison in Section X), and it's a free country (and even freer Internet): we don't care what software you prefer or use. We do believe that LaTeX, combined with a proper editor and editing workflow, usually makes writing and formatting faster and better.

3 What is LaTeX?

What is LaTeX? What is a markup language? Do I need to know programming? Is LaTeX programming?

You write instructions (i.e. code) to tell a latex compiler how to prepare your PDF, i.e. how to format, structure, and generally manipulate text, images, tables, etc to make your PDF.

[1] Thanks to Keegan Kuhn for this pun.

Analogies: latex is the movable type of PDFs.

4 Feature Comparison

4.1 The General WP

First cover features, operation (GUI: menus, buttons, and keybinds), and general workflow. Cover the WP mindset.

Advanced Features

Most WPs lack advanced features such as generating a table of contents [2], generating lists of figures, more, and more as well as configuration of all of the above.

A Hidden, Mutable State

WPs are "What You See Is What You Get" [3] editors. What appears in the WP window will appear in your PDF: what you see in the WP is what you're always gonna get. There's no separation between input and output: the writing you put into the WP will be present in the output, whether that's the WP file itself, a generated PDF, or a screenshot of the WP window, perhaps.

Because what you see is what you'll get, any characteristic of your document must be telescoped, projected, or mapped into visual form. Consider boldface styling. Boldface style is the thick version of a font; applying bold to a word makes its font thicker. Crucially, boldface must be applied /to a word/. A word processor, however, has no conception of a word. If you wish to have a word be in boldface, you can select the word by highlighting it and make it bold, often with a keybind or button. Your conception of the bold word, however, has not been mapped one-to-one to the WP: information has been lost. Specifically, your conception of the bold word relies on your conception of the word itself, and this conception in turn relies on the ability to define start and end points of a word. The WP, however, has no conception of a word, and thus cannot define its start and end points. Sure, you may have highlighted the word by starting at one letter and ending at another, but the WP doesn't know this. Your intention, of bolding that word itself (as defined by start and end points), has been lost in the projection of your conception of bolding onto the WPs visual space.

This information loss leads to weird, and often annoying, behaviors. If I bold a word, for example, and then place my cursor in the middle of the word and continue typing, the new letters will remain in bold. Even if I first type a space, which should, in my conception, signify the start of a new word, the letters are bold. In fact, WPs allow you to apply bold style to things that aren't even words, even to spaces. Hopefully my explanation has been clear enough, but consider how ludicrous it is to make a space boldface.

[2] As shown in the next section, many WPs can actually generate a table of contents. Their ToCs, however, are basic and impede workflow, and thus are not considered advanced.

[3] WYSIWYG, pronounced "Wi-see-wig"

The many-to-one mapping from your conception to the visual WP environment doesn't simply cause the lost information to disappear. In fact, WPs require you to keep track of metadata about the document, which can be termed the document's "state".

All this mapping into visual space requires you to keep track of state. You have to remember if you had put a page break somewhere, and if so, where on the page the break was inserted. Or if the boldface ends at the end of a boldface word or if it applies to the space after it as well.

"bold" emphasis style. When you emphasize a word with boldface, you desire that word to be bolded. The WP, however, has little, if any, conception or definition of a "word". In this visual environment

Because of the WP's limitations

4.2 LaTeX

Quoted from https://blog.orvium.io/latex-over-word/:

1. LaTeX is specifically designed to produce high-quality typesetting, which makes your documents look professional and polished
2. since you're essentially writing code, you can meticulously fine-tune your document to look exactly the way you want it or in accordance with the highly specific requirements some journals have.
3. LaTeX is optimized for minimal resource utilization. This allows researchers to work more efficiently on large documents with many equations, figures, images, and cross-references (think dissertations, books, or studies).
4. LaTeX also generates a table of contents, a list of figures, and a complete list of references which you can manually edit in code
5. the
   input and
   include commands allow you to split up sources in a controlled way, effectively making large documents into smaller files that can be managed separately.
6. Since LaTeX operates with plain text files, the level of control you have as a user is beyond what traditional word processors can offer. This can prove very handy when collaborating with multiple authors on a big project, as it allows you to use tools like Git or SVN to implement version control and track changes.
7. Anything that prominently features equations, tables, figures, or other designs is best completed via LaTeX.
8. Academic dissertations and doctoral theses - from the reference system to the automatic and efficient table of contents, LaTeX makes working on gigantic projects such as these very easy. By comparison, researchers using Word frequently save chapters in separate documents to keep the software from lagging up or crashing and thereby losing their work.

Regarding 1, go into what typsetting actually is, i.e. good-looking justification, text wrapping, math, equations.

Compare Overleaf opening "10-page" worth of latex code with google docs, onedrive 10-page document.

Quoted from https://www.baeldung.com/cs/latex-vs-word-main-differences:

1. it serves as a broad category encompassing all word-processing software that immediately displays the final output. Having this characteristic, Word is usually called a What You See Is What You Get (WYSIWYG) editor rather than requiring initial file processing.
2. On the other hand, LaTeX's user interface is distinct from conventional word processors, as it relies on a markup language for document creation rather than a graphical interface
3. Word supports real-time co-authoring, where multiple users can work on the same document simultaneously without conflicts.

Quoted from https://tex.stackexchange.com/questions/110133/visual-comparison-between-latex-and-word-output-hyphenation-typesetting-ligat:

1. how many unnecessary double paces or blank lines have any big Word document of an average user? This mistakes are hardly noticed and corrected and spoiled the format, but simply does not exist in LaTeX.

Regarding 1, maybe Word can highlight two spaces to show an error, and you can just delete the extra space. What latex offers, however, is the ability to forget about these typos. You can focus on your writing, rather than these minute procedural details.

Quoted from https://nitens.org/w/latex/:

1. Popular word processors either lack support for kerning tables or disable kerning by default
2. Most word processors create fake small capitals by adjusting the size of capitals
3. A good typesetting programme should always use contextual intelligence and substitution tables to determine whether ligatures are needed
4. Readability results not only from a good selection of typefaces, but also from a correct distribution of characters and whitespace per line. To attain this goal, most WYSIWYG word processors use relatively dumb justification/hyphenation procedures

Regarding all these, verify for newer WP versions (including cloud):

Quotes from https://www.andy-roberts.net/latex/benefits/:

1. LaTeX is essentially a markup language. Content is written in plain text and can be annotated with various 'commands' that describe how certain elements should be displayed. The LaTeX interpreter reads in a LaTeX marked-up file, renders the content into a document and dumps it a new file. Therefore, it's not an interactive system

that is the de-facto method for document creation nowadays.

2. You can get LaTeX to do just about anything you can think of! Over the years, an overwhelming selection of packages to extend its potential and macros that can simplify complex tasks have come into being, most of which are freely available on CTAN
3. LaTeX's main users are within academia and research institutions
4. There are other crazy packages that you can install which allow you to typeset music scores, chessboards and cross-words! CTAN is the main repository of these resources. Most are well documented and as you can imagine, with LaTeX being around for so long, the number of extensions is vast. The chances are, if you're struggling to do a task, someone will have undoubtedly written a package to solve it easily!
5. Even with simple documents, you can quickly become frustrated by Word's rather unintelligent interference. The hours that are wasted trying to position that image which you know will fit at the bottom of the page, but Word refuses to put it there! How many can relate to this experience? You have your 30 page document with text, tables and images. You just spent the evening getting it formatted nicely - all your figures in the right place and then you notice that one of your paragraphs isn't clear enough. You add one sentence, which then pushes an image on to the next page, leaving a massive gap at the bottom of that page where your image once was. This then daisy-chains down, knocking other tables and images out of place all the way to the end of your document! It's a real laugh. Fortunately, LaTeX is much more clever in this respect and positions your images and tables with a lot of common sense. So, if you want your image to appear at the bottom of a given page, it'll stay there!

   Whilst LaTeX makes decent typesetting decisions for you, if you want to, you can have total control over the presentation of your document.
6. It's difficult to disagree that the output from LaTeX is far superior to what Word can produce. This is emphasised greatest when it comes to documents with high mathematical content, which is a major strength for LaTeX. It also has much better kerning, hyphenation and justification algorithms that simply make the output far more professional than what any word processor. Its algorithms for laying out text are more sophisticated and extremely fine-grained. For example, the accuracy is so high because it uses a measurement known as a scaled point which translates as 100th of the wavelength of natural light!

   LaTeX works with the concept of niceness (well, I suppose technically it's badness - which it works to minimise). LaTeX has a large set of metrics that it evaluates against when generating your document. It experiments with various permutations of parameters and determines the one which gives the "nicest" output. It can take the time to do this because it isn't interactive. Word processors don't have the computational resources available (yet) to carry out the equivalent calculations and still remain interactive. Also, many people forget that typesetting is actually a professional skill - people train for years to learn how to layout publications. Yet, as soon as you open

a word processor, you go about committing typesetting sins all the way. Typesetters know for example that its easier to read sentences that are approximately 66 characters wide. Have a look in your books and count the letters! Also, why do newspapers and magazines have narrow columns? But, the default layout of a word processor gives an average of 100 words per line. I suppose many people don't mind, but you would notice if you read a lot of large documents.

7. One of the reasons why perhaps so many people struggle with Word when creating large documents, is because it is prone to crashes. 'Document recovery' is now a high ranking feature of Word. I'm sure people would prefer if MS would just make their software more stable! ... Because LaTeX is so mature - and developed by extremely clever programmers - bugs are negligible. And even if it were buggy, then there is no risk of you ever losing your original source text. Where as with Word, almost any tool within its integrated environment is capable of corrupting your file if it causes a crash.
8. Word may have the advantage of a GUI which is good for beginners. It reduces the cognitive load as it's a case of recognition verses recall.
9. Word can be extended using its in-built scripting language. It also has document management features to help with large documents. As already mentioned, it has styles that can ensure manageable and consistent presentation. Yet very few people seem to take advantage of them. This is especially worsened by UI improvements that mean Word will hide features that you do not use, which makes it more difficult to remember what Word can actually do.
10. Quite simply, anyone who is writing non-trivial documents and is tired of being let down by the performance of the current crop of word processors. If you are in academia, you really ought to be using it! Anybody writing anything maths related will not find a richer and better quality system. For example, even WikiPedia use LaTeX for rendering any formulas that appear on their site.
11. LaTeX isn't for people who are too lazy or dislike change! I personally found the investment paid off because LaTeX allows me to produce my documents at a greater pace. I know that the enterprise will not be interested as Word is so ingrained, even though their business reports would look so much nicer. Their loss! For everyone else, it's time to give it a fair try, just so that you compare and contrast, then decide which does the job best for your needs.

Regarding 9, verify this last sentence.

Quoted from https://tug.org/texshowcase/:

1. You can compare a Word Processor (e.g. MS Word) setup to a TeX setup as a Camper (or RV) versus having a house and a car. The Camper is for everything: you can live in it, you can drive with it and you can look at it. The Word Processor is like a Camper: it does editing, formatting/typesetting, and displaying. It is not excellent at any of these functions, but the combination is pretty neat. In a TeX setup, these functions are separated, like with having a house and a car.

2. Some things are, however, difficult to do in TeX. Mostly these are the kind of things where you want very fine-grained control over exact positioning of images, wrapping around these images, etc. You can do this in TeX, but it is often (very) cumbersome to get it right and changes may be a lot of work.

Quoted from https://tex.stackexchange.com/questions/94889/how-can-i-explain-the-meaning-of-latex-to-my-grandma:

1. LaTeX is to a book what a set of blueprints is to a building.
2. The LaTeX user is the architect that, designs the blueprint, for the computer and printer to build... Any trade-y can throw together a shed without any kinda of plan, but a beautiful building requires blueprints
3. It does this [movable type] but it uses a computer and so requires less manual labour.
4. https://tex.stackexchange.com/a/94910 (has photos)

Quoted from

First cover features, operation (TUI: writing code and compiling), and general workflow. Cover the latex mindset.

4.2.1 Don't Switch to LaTeX If...

Don't switch to LaTeX if

1. You're not willing to learn the minimum necessary for your needs (might just be math equations, or might extend to bibliographies).

2. All this being said, LaTeX is just text (another benefit!). If you've started a project but can't finish it in LaTeX, you can always copy/paste it somewhere else, or even export it directly to docx with tools like Pandoc (won't export everything perfectly).

4.3 Table of Features

We should quantify as many features as possible and present them in a table for a comprehensive feature comparison.

4.3.1 LaTeX vs. General WP

4.3.2 Overleaf vs. Google Drive vs. Microsoft Onedrive

5 How Tos

5.1 How to Read Error Messages

5.2 How to Debug LaTeX Code

5.3 How to Use LaTeX Package Documentation

5.4 How to Find Help

tex stackexchanger, overleaf.com/learn, google searches

5.5 How to Use Detexify

Really simple.

6 Building Blocks

6.1 Preambles

6.1.1 Useful Packages

6.1.2 Homework

6.1.3 Reports

IQP

MQP

6.2 Tweaks

We can add the code snippet then link some documentation for people to dive deeper.

1. Header and footer for nice homework.
2. Two figures side by side.
3. Appendices
4. Bibliography
5. acronyms

6. landscape pages
7. block quotes
8. list spacing
9. actually put floats where you want them
10. different fonts and math fonts
11. footnotes
12. glossaries
13. page margins
14. date formatting
15. multi-row table cells
16. code styling
17. include PDF pages
18. charts and other graphs
19. justification
20. line spacing
21. reduce vertical space between word and underline
22. vertical table cells
23. URL colors
24. single space table of contents

6.3 Snippet Plugins

Vim, emacs, vscode.

Naming convetions

Good to have: fraction, etc.

6.4 Naming Convetions

6.4.1 Sections

Good practices for naming sections

6.4.2 Labels

Good practices for naming labels, e.g. `sec.subsec.subsubsec` (dot notation) or `sec:subsec::subsubsec` (colon notation).

6.5 How To Manually Modifying Spacing

How to use positive and negative vspace and hspace. How to use vfill and hfill.

Macros for inserting a preset amount of horizontal space, e.g. to separate answers within a box

6.6 LaTeX Editing Workflow

Your own preamble. File templates. Homework templates. Bibliographies

7 LaTeX Editors

TUI (emacs and vim), VSCode, overleaf. Find other popular ones to cover.

8 Overleaf for Collaborative Editing

Don't give Overleaf tutorial (they have docs for that). We should outline how to manage group projects and collaborative writing in latex.

9 Inkscape and LaTeX For Fancy Diagrams

10 Tips

Use % between lines to break up paragraphs in the source but not create a line break in the output.

Having source in one window and previous PDF can be super helpful. Let's say you have variable a = f(b,c,d) and you need to substitute it into equation $y = a^2 + e^{-a/\sqrt{a}}$. Preserving the original PDF allows you to view the y(a) equation in its old form, with only 'a', as you insert f(b,c,d) into the latex source. You can in fact simply copy the f(b,c,d) and paste it into the y(a) equation in the place of every appearance of 'a'. Because the PDF is preserved, you don't have to remember what y(a) is as you insert in f(b,c,d). This means that while your source code will reflect some intermediate equation y(a,b,c,d), your PDF will still reflect y(a), and you won't e.g. lose track of which 'a's you've replaced or not. Also, before recompiling the PDF to reflect y(b,c,d) you can use the old y(a) in the PDF to double check that you've actually inserted f(b,c,d) into y(a) in the correct places in the source.

We find this paradigm of old PDF + prototyped source to be useful in other fields, such as writing papers. While retaining an old version of a sentence in your PDF you can experiment with a new, reworded version, in your source.

11 Working on big projects (mqp, iqp, thesis)

Things to leave for last:

1. diagrams. A final report level diagram will take 30 minutes, if you pay attention to details like exact positioning of the rendered latex
2. structural formatting, including whitespace and figure placement. if you do this last, there's no chance that you'll add some paragraph somewhere which ends up shifting your figure and having you redo all the working of shifting it back where you prefer
3. citations
4. enabling double spacing. you can try this mid-way to gauge how long your paper is, but it'll be easier to write if you can see more of the text (this isn't a latex-specific suggestion, more about writing in general)
5. revising paragraphs so that the last sentences doesn't bleed 1 or 2 words over onto the next line. note that you can potentially regain a lot of vertical space this way.
6. acronyms. you don't need to acro wrap each use, only the first few (in each chapter?)

Latex makes you more attuned to formatting details because it makes things look high quality, so small things that are low quality stand out. i.e. it sets a higher standard than WPs, so you're more likely to correct things in latex than WPs cause they'll stand out more in latex. And the opposite is true: the higher real-time cognitive load of word processors makes you less attuned to detail. Because word allows impossible details like a "bold space character", you'll be in a mindset to not consider the details as much. Also, because word makes it difficult to do things, like move an image to the exact place you want it, your mind will be closed off to fine-tuning your document. Again, because WPs limit functionality that contributes to detail, e.g. easy toc, list of figures, captions, references, your mind will be closed off to thinking about fine-tuning your document.

programmatic find and replace with acronyms for things you don't know what to call yet.

latex is nice because it compresses certain things, like a full page image, into several lines of text. so if you scroll up in WP to the previous line, you might lose context. but in latex you just scroll up a tiny bit in the source, or even don't have to scroll up at all, and you can see the previous line.

Big Projects

1. follow the template and folder hierarchy we provide
2. label chapter in CAPS
3. use something like Lorenzo's Elisp `my-autofill-label` to help with really long labels
4. Use this snippet to help lsp find refs in other files:

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: "../report.tex"
%%% End:
```

5. tell people to add a find-replace pattern, like "(cite)" when the want to cite someone. certainly they should at least put a basic in-text citation or at least identifier, i.e. (Knuth, pg 18). But find-replace patterns can help you insert citations quicker

Copying from google drive in big project

1. copy paragraph by paragraph
2. replace acronyms, quotes, other artifacts etc at the end of final draft
3. you'll have a gray area where people are still updating the cloud doc and you're copying over changes manually
4. you can ask people to comment or highlight any changes to make it clear if you don't need to recopy a section
5. I'd recommend revising your own work directly in the tex