LaTeX for academic documents

Pranav Minasandra | pminasandra.github.io

Contents

1 Background

Welcome to my LaTeX workshop. LaTeX (pronounced *Lay-Tech* or *Laa-Tech*) is a powerful software written primarily with the goal of typesetting academic documents. The system on which it is based, TeX, is quite old and forms the basis of a large majority of typesetting systems. It is also Turing-complete, in the sense that interested people can, and have, used LaTeX as a programming language to do crazy things, including writing a control system for a Mars rover. Since we are not masochists, and since LaTeX is supremely unsuitable for anything other than document typesetting, that is all we will focus on.

Systems like MS-Word are often called WYSIWYG systems, which stands for 'What you see is what you get'. LaTeX is referred to as a 'What you mean is what you get system'. The expectation is that you spend not much of your time fiddling with the formatting of your document: the software handles all that for you. You just focus on getting your text down.

We will be writing out code using Overleaf which is an online implementation of LaTeX. Overleaf is very cool, and have added a bunch of features to make LaTeX work smoother. You also avoid having to install

this quite bulky software. However, if you want to install the software locally on your system, you can do so quite easily. There are many implementations, such as TeXStudio and TeXMaker.

2 Getting started with LaTeX

Once you have logged into Overleaf, you are ready to create your first document. Click "New Project" and then click "Blank Project". After you have filled in everything Overleaf asks for, you will be greeted with multiple panels, predominant among which will be two panels: one for the LaTeX code and another showing your PDF. **Delete all the pre-made code in the Overleaf template**, we are going to start learning this stuff from scratch.

Once you've deleted everything, click "Recompile". Overleaf complains immediately with the statement "This compile didn't produce a PDF." This is because we have not informed LaTeX what it is supposed to do.

Now, type the following in the code panel and hit 'recompile':

```
\documentclass{article}

\begin{document}
Hello, world!
\end{document}
```

Congratulations, you've made your first LaTeX PDF! It's not much to look at right now.

2.1 Some basics

The code you wrote above forms the basic elements of a LaTeX document. The first line, `\documentclass{article}`, tells LaTeX that we are writing an article. There are a lot of document classes available. The `book` class is the best for writing theses and, well, books. The `report` class is good for writing multi-part articles, `beamer` for making great slideshows, `letter` for writing letters, and so on. The true number of LaTeX built-in document classes is enormous, and there are many non-built-in ones, for instance, for submitting drafts to specific journals.

LaTeX is run largely by control-sequences. These look like this: `\acontrolsequence{some text}`. We will encounter a lot of these further below. Another way you can tell LaTeX what you need is using environments. These look like this:

```
\begin{anenvironment}
Some text here.
\end{anenvironment}
```

Environments and control sequences can be nested within each other, but of course, based on what you're nesting where, unpredictable bullshit can always occur.

A LaTeX document receives commands with the '%' character, as R and Python do with the '#' character. Below, I show some regions of the standard LaTeXdocument using comments.

```
\documentclass{article}

% This region is called the preamble.
% It's where we configure and set-up
% the document we are generating.

\begin{document}

% This is the body of the document.
% All text here is used to make the
% final PDF.

Hello, world!
```

```
\end{document}

% Everything after the \end{document} is
% ignored. It's a great place to store
% work in progress text.
```

2.2 Formatting text

Here are some control sequences that help you format text in useful ways:

`\textbf{bold}, \textit{italicised}, \textsl{slanted}, \texttt{typewriter-text}`.

These respectively make the text **bold**, *italicised*, *slanted*, and `monospace`. There are also the following control sequences available for controlling the size of the text:

```
\tiny, \scriptsize, \footnotesize, \small, \normalsize,
\large, \Large, \LARGE, \huge, \Huge
```

These can be called like this: `This text will be \tiny{tiny} while this will be \Huge{huge!}`

This text will be tiny while this will be huge!

As LaTeX is a what you mean is what you get system, generally, you don't have to use any of this. There will be some rare cases of course, but generally, all you need for emphasis is the `\emph` command, usually mapped to the italicised text control sequence. `This text needs to be \emph{emphasised}.` : This text needs to be *emphasised*.

3 Using packages

Any more advanced tasks than this will need us to import packages. Importing packages is done with the control sequence `\usepackage` in the preamble of the document (i.e., in the part between `\documentclass{article}` and `\begin{document}`). We will use a package to fill our document with some filler text. Modify your code to look like this:

```
\documentclass{article}

\usepackage{lipsum}

\begin{document}
\lipsum[1-3]
\end{document}
```

Recompile again to see three paragraphs of text, which exist to show you what text would look like here.

You might have noticed that the paragraphs are clumped up and way more compact than they need to be. Add the package `parskip` just like you added the package `lipsum` above to fix this.

As a final example, we will set the margins of the document to use more of the available space. This uses the package `geometry`, with some arguments. The line you need to the preamble is

`\usepackage[a4paper, margin=1in]{geometry}`.

Here, `a4paper` and `margin=1in` are called arguments.

Through this workshop, and later when you work on your own documents, we will be using a lot more packages. Just remember that they need to be added and configured in the preamble. All packages are added using the `\usepackage` control sequence.

4 Configuring your document

Now we will tell LaTeX explicitly what exactly we are working on. This involves declaring the title and author of the document. In the preamble, add:

```
\title{My first \LaTeX{} document}
\author{Pranav Minasandra}
```

You also need to add one line after `\begin{document}`, like this:

```
\begin{document}
\maketitle
```

Based on your needs, there are fancier ways of adding authors and affiliations, for instance, if you are writing a paper. We will not talk about them here, but I will help with them during the workshop if you need these methods.

5 Adding lists

5.1 Itemized list

LaTeX supports a variety of lists in text, although not many are used often in academic writing. Unnumbered lists may be made as follows:

```
\begin{itemize}
\item Item 1
\item Item 2
\item This is the third item.
\end{itemize}
```

- Item 1
- Item 2
- This is the third item.

5.2 Numbered list

Similarly, you can replace `itemize` with `enumerate` to get a numbered list:

1. Item 1
2. Item 2
3. This is the third item.

Sometimes relevant in academic writing is the inline list. Add the package `enumitem` to your preamble with the argument `inline`, as follows:

```
\usepackage[inline]{enumitem}
```

Now, you can replace `enumerate` above with `enumerate*` to get an in-line list.

6 Sectioning

You can create numbered sectioning within your text using the control sequences `\section`, `\subsection`, and `\subsubsection`. Try them out! The way to use them in the body of your document is as follows:

```
\section{Introduction}
\lipsum[1-4] %Some dummy text

\subsection{Historical ways of measuring something}
\lipsum[5] %More dummy text
```

Sometimes, you might preferred unnumbered sectioning, especially with sub-sub-sections because you don't really want any sectioning with numbers like `2.2.4.3`. In such cases, you can use the control sequences `\section*`, `\subsection*` and `\subsubsection*` in place of their non-starred versions to add unnumbered sectioning. However, note that unnumbered sections will not automatically appear in your table of contents.

As an exercise, make 'Introduction', 'Methods', 'Results' and 'Discussion' sections in your document.

7 Adding figures

Adding figures in LaTeX consists of knowing to do basically two things. First, you need to know how to add images to a document, and second, you need to know how to convert an image into a scientific figure together with its own figure number and caption.

7.1 Adding an image

First and foremost, you need to add the package `graphicx` to your preamble. This package makes available to you the control sequence `\includegraphics`, which lets you include images of most file-types. I *very strongly* recommend storing your images as PDFs. R and Python (matplotlib) allow this by default, and LaTeX is particularly great at handling PDF images, although I have tried JPEG, PNG, and TIFFs already with no problem. You can also include SVG images, but you might face a small battle with LaTeX in the beginning.

We will now include your first image in your document. Create a folder (button in top left panel) in Overleaf called, let's say, `images`. In this folder, upload an image of your choice, let's say it's called `photo.jpg`. Now write the command:

```
\includegraphics[]{images/photo.jpg}
```

You might notice that the image is wrongly sized, and appears either gigantic or tiny. The first solution is to make images of the correct size in your programming language: 640 Œ 480 is ideal, for instance. However, this option is often not feasible. In such cases, we will add an argument to the `\includegraphics` command to set the width (and automatically therefore the height) of the image we are inserting.

```
\includegraphics[width=0.8\textwidth]{images/photo.jpg}
```

The argument `width=0.8\textwidth` tells LaTeX to scale the image so that its width is equal to 80% of the total width available for text in this document. Other arguments are available to scale the image by a given factor, rotate it by some angle, and set its opacity.

7.2 Making an image into a figure

In academic writing, an image becomes a "figure" when it is supplemented with a caption, numbered for reference, and thoughtfully placed in the document. Figures must help organize and clarify your work.

We'll now create a figure in your document. Here's an example of how to add one:

```
\begin{figure}
\centering

\includegraphics[width=0.8\textwidth]{images/graph-x-y-vs-z.pdf}
\caption{Blood pressure and net worth are predictable by a machine learning
system, or some other equally horrendous example caption.}
\end{figure}
```

In this example, the caption will be places directly below the figure, as is typical in academic writing. Captions should be placed directly below the figure, which LaTeX handles for you automatically.

By default, LaTeX will try to place figures where it thinks they fit best in the document. This automatic placement works well in most cases, but sometimes you'll want more control. You can use optional arguments to the `\begin{figure}` command to specify where the figure should appear:

```
[h] - Place the figure here in the text, as close as possible to the current location.
```

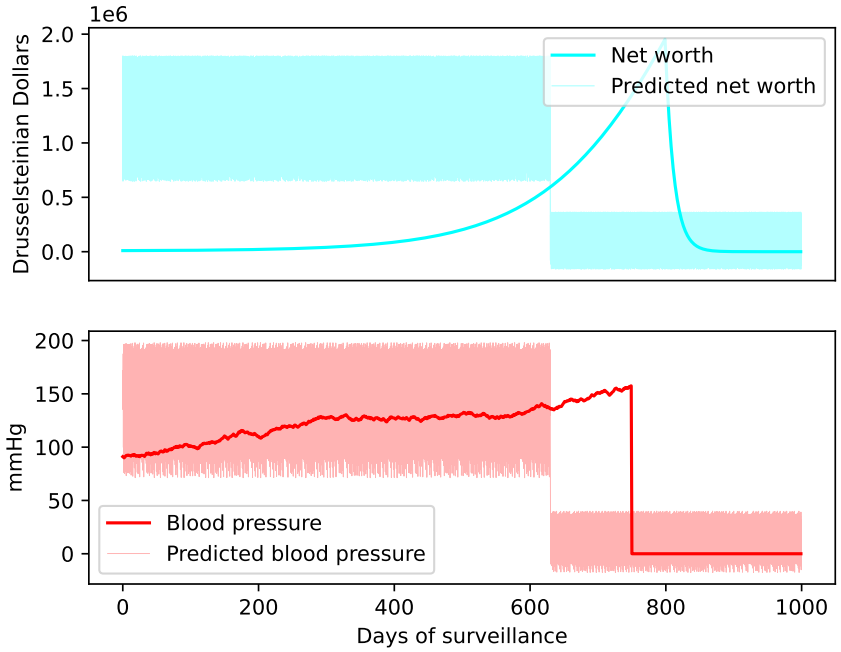

Figure 1: Blood pressure and net worth are predictable by a machine learning system, or some other equally horrendous example caption.

```
[t] - Place the figure at the top of the page.
[b] - Place the figure at the bottom of the page.
[h!] - Force the figure to appear exactly here, even if it disrupts layout.
```

For example, if you want your figure to stay near the text that references it, you can write:

```
\begin{figure}[h]
    \centering
    \includegraphics[width=0.8\textwidth]{images/graph-x-y-vs-z.pdf}
    \caption{Example caption for a well-placed figure.}
\end{figure}
```

Now our figure has been made. Check out Figure 1.

8 Cross-referencing

Cross-referencing is a powerful feature in LaTeX that allows you to connect different parts of your document, such as sections, figures, and equations. By using cross-references, you ensure your document is both easier to navigate and consistent, even as you make changes.

The `\label` command is used to assign a unique identifier to an element in your document, such as a section or figure. The `\ref` command then allows you to refer to that element elsewhere in your text.

For instance, we can label the section *Introduction* and refer to it here to guide readers. Navigate to the place where you added your *Introduction* section, and modify it so that it looks like this:

```
\section{Introduction}
\label{sec:intro}
```

Now you can refer to this section anywhere within this document like this:

```
... as explained in the introduction (\ref{sec:intro}), we have ...
```

Try it out! Similarly, we can label the figure we created earlier and reference it. After your caption in your figure, modify the code to look like this:

```
\caption{... your caption here ...}
\label{fig:figuredescription}
```

And, as before, we can refer to this figure like this:

```
We have shown (Figure \ref{fig:figuredescription} that...
```

The main advantage is that, if you swap the order of sections or figures, the text adapts automatically to still remain valid without you thinking about it too much. Making `\label`ing and `\ref`ing a habit pays off immensely in the long run. However, these are not all the perks of LaTeX cross-referencing.

The package `hyperref` when added modifies your document immensely so that relevant bits automatically become hyperlinks. Add this package as follows in your preamble:

```
\usepackage[hidelinks, colorlinks=true, allcolors=blue]{hyperref}
```

We will now look at two cool new control sequences made available to us by the `hyperref` package. Change the `\ref` to your figure to `\autoref`, and change the `\ref` to your introduction to `\nameref`, and see what happens.

8.1 Good practices in cross-referencing

1. **Choose meaningful labels:** Use descriptive names that reflect the content, such as `sec:intro-litreview` or `fig:k-int-vs-R-minus`.
2. **Avoid hardcoding numbers:** Relying on cross-referencing ensures that section and figure numbers update automatically when you modify your document and figure order. Don't do `\label{fig1}`
3. **Double-check your labels:** Typos in labels can cause errors or incorrect references, so it's worth verifying them as you work. These are the most common culprits in LaTeX errors.

By incorporating cross-referencing, your manuscript will be more professional and adaptable, especially for long or complex projects. When accepted, it will also make the journal copy-editor's job much easier.

9 Citations and bibliographies

Now comes one of the worst, but also one of the most convenient features of LaTeX—its bibliography management system. When used effectively, LaTeX bibliographies are fully deployable and scalable. This means you can move text between completely unrelated documents and still retain meaningful citations, a feature particularly useful for academic and professional writing.

There are many ways to handle citations in LaTeX, but here we will use the `biblatex` package with the `biber` backend. This combination offers flexibility and modern formatting options. To get started, add the following to your preamble:

```
\usepackage[backend=biber,
            citestyle=numeric-comp,
            bibstyle=ieee,
            sorting=none,
            doi=true]{biblatex}
```

9.1 Setting up the bibliography file

In Overleaf, create a new file in your project directory called `bibliography.bib`. This file, often referred to as the *bib-file*, acts as a centralized database of all the sources you intend to cite in your document.

Each entry in the bib-file, called a *bib-entry*, is a structured record containing details about a source, such as the author, title, publication year, and DOI. For example, an article entry will look like this:

```
@article{key,
        author = {Author Name},
        title = {Title of the Work},
        journal = {Journal Name},
        year = {Year},
        volume = {Volume},
        number = {Number},
        pages = {Page Range},
        doi = {DOI (if available)}
}
```

Bib-files are immensely useful because they keep all your references in one place, separate from the main document. This separation allows you to easily update or share your bibliography with other projects or collaborators. Additionally, you can mix and match bib-files, reusing existing references without any manual reformatting.

9.2 Getting bib-entries

The easiest way to add entries to your bib-file is through Google Scholar. Search for the source you wish to cite, click the quotation mark icon under the result, and select the BibTeX option. Copy the entry into your `bibliography.bib` file.

For larger projects, tools like Zotero and Mendeley can save you significant effort. These citation managers let you collect references directly from your web browser, organize them into folders, and export bib-files with just a few clicks. This is particularly useful when dealing with dozens or hundreds of citations, as they automatically ensure formatting consistency and reduce errors.

9.3 Example

Add the following bib-entry to your `bibliography.bib` file:

```
@article{einstein1905,
  author  = {Albert Einstein},
  title   = {Zur Elektrodynamik bewegter K{\"o}rper},
  journal = {Annalen der Physik},
  year    = {1905},
  volume  = {322},
  number  = {10},
  pages   = {891--921},
  doi     = {10.1002/andp.19053221004}
}
```

Then, in your main text, cite the entry as follows:

```
Einstein's paper on relativity revolutionized physics \cite{einstein1905}.
```

Finally, to print the bibliography at the end of your document, add:

```
\addbibresource{bibliography.bib}
\printbibliography
```

Einstein's paper on relativity revolutionized physics [1].

10 Looking forward to the afternoon session

This has been a quick manual to follow along with my talk. It is completely okay if you aren't very comfortable with any of this stuff yet, the point is that LaTeX is like any language—it becomes easier the more you use it. In my own experience, LaTeX becomes much easier much quicker than things like R and Python which you are already likely using.

This afternoon, we will typeset a paper, proposal, or other academic document that you're currently working on using LaTeX. There are many resources online to learn LaTeX quickly and serve as good references, like this one and this one from Overleaf. Let's hope we have a productive afternoon session!

References

[1] A. Einstein, "Zur elektrodynamik bewegter körper," *Annalen der Physik*, vol. 322, no. 10, pp. 891–921, 1905. DOI: 10.1002/andp.19053221004.