An Introduction to LaTeX

David Liu

January 7, 2019

1 Introduction

This document is intended as a gentle introduction to some of the basic features in LaTeX. Questions, comments, and error-finding are most welcome.

Getting started with LaTeX is very easy: either download some LaTeX software (e.g., from here: `http://latex-project.org`), or use an online editor like `https://www.overleaf.com/`.

What you've downloaded from the course webpage, sample_latex.tex, is a LaTeX source file (think Python .py file). Open it up in your chosen LaTeX editor, and compile it using pdflatex. This is the equivalent of running a Python source file. The result is a new pdf file that contains a typeset version of your source file, based on the text and commands in the source file. Every time you change the source file, you must recompile using pdflatex to see the changes in the pdf.

2 Basic Commands

2.1 Text

Just start typing, and you'll produce a paragraph. Here is *italics*, **bold**, and `monospace (code)`. For readability, you can type sentences on multiple lines in the source file, and they will show up in the same paragraph.

Use a blank line to separate actual paragraphs. The first paragraph of a section is not indented, while the rest are. Here's a LaTeX trick: use the forward quote key at the top-left of your keyboard to do beginning quotation marks, and the regular apostrophe key to do ending ones. Observe: Professor Liu is "cool." [1]

2.2 Math

Let's get to the good stuff! To make x=3 look nice, we use *inline math mode* with $ signs: $x = 3$. There, wasn't that nice? Here's some more math: $x^2+y^{10} = -\pi \cdot \rho \circ \alpha / \frac{1124}{z_0}$ Notice how that fraction at the end looks a little squished? Here's how to make math equations stand out a bit more, and given them some extra room:

$$\frac{1}{2}A^{Z^{ddd}} + \frac{B}{C} + D_{Y_{123}} = \sum_{i=1}^{10} \frac{1}{1+\frac{1}{i}}$$

And you can just keep on typing after that. Notice the use of (nested) superscripts and subscripts.

Here's an example of using text while in math mode.

$$A = \{n \in \mathbb{N} \mid n \text{ has exactly 4 factors}\} \text{ is a set that I care deeply about.}$$

There will be times that you might want to produce multiple equations down a page. Doing this with the \[...\] isn't great. A much better solution is the use the **align*** environment:

$$\begin{aligned} x^2 + 4x + 3 &= 0 \\ (x+1)(x+3) &= 0 \\ x &= -1, -3 \end{aligned} \tag{Eq. 1}$$

[1] Seriously.

3 Three Useful Environments

3.1 Lists

Say you're doing an assignment and want to organize your solutions by question (this should be a goal...). One way to do this is the following:

1. Here's my answer to question 1.

2. So far so good, but then when my solution is really long and starts to run over the page width my numbering starts to blend together. And of course this is an assignment so I'm writing a lot of words, and it's getting to the point where a single paragraph is multiple lines long.

This is especially troublesome when I want to have multiple paragraphs in my solution. Again, a good idea to prevent huge walls of text assaulting the markers.

Here's another paragraph before my answer to question 3. Just because.

3. Hey, here's my answer to number 3. Try not to miss it among all the paragraphs. Because there are a lot of paragraphs.

While I could try to play around with indentation and margins, there is a much better way. Use the **enumerate** and **itemize** environments!

1. Wow, an ordered list!
2. Wouldn't it be so meta to list out the reasons this environment is good?
3. It can even do nested lists!
   (a) For those pesky questions with multiple parts. We did change the numbering myself, though.
   (b) There is a default numbering scheme. Try deleting the [] parts to see what happens.

- Okay, an unordered list.
- Maybe less useful for assignments, but you never know.
- This also supports:
  - nesting.
- And both list environments support all of the usual math stuff:

$$(x+y)^2 = x^2 + 2xy + y^2$$
$$(x+y)(x-y) = x^2 - y^2$$

3.2 Tables

This environment is a bit rarer, and there are many options and more advanced environments you can use.

Here we're killing *three* birds with one stone:

- showing you how a table works
- giving you a list of symbols you might find useful as a reference
- introducing the csc package, which provides easier to remember names for some of the commands

Symbol	LaTeX command	csc command
$\wedge$	\wedge	\AND
$\vee$	\vee	\OR
$\neg$	\neg	\NOT
$\Rightarrow$	\Rightarrow	\IMP
$\Leftrightarrow$	\Leftrightarrow	\IFF
$\forall$	\forall	N/A
$\exists$	\exists	N/A
$\in$	\in	\IN
$\notin$	\notin	\NOTIN
$\sum_{i=0}^{n}$	\sum_{i=0}^{n}	N/A
$\prod_{i=0}^{n}$	\prod_{i=0}^{n}	N/A
$\to$	\to	N/A
$\mathbb{N}, \mathbb{Z}, \mathbb{R}$	\mathbb{N,Z,R}	\N, \Z, \R
$\mathcal{O}$	\mathcal O	\cO
Ω	\Omega	N/A
Θ	\Theta	N/A

If we think of more symbols to add, we'll update this table. If you have any suggestions, please let us know!

4 Miscellaneous

This section will be updated as the term progresses, as ideas come to us, either from course staff or from you!

4.1 Images

This subsection shows how to import a basic image file into your document. It's also possible to use LaTeX commands to draw an image directly, but we won't cover that here. Please note that the easiest way to import an image file is to put it in the same directory as your LaTeX source file.

Figure 1: It's a tiny polar bear in a snowstorm.

4.2 Algorithms

All of the code in this course will be written in Python style, though we will not strictly worry about the exact syntax. When you refer to functions, variables, or other code-related nouns in your text, you can use the command "texttt" to get a traditional monospace font. For example, in the code of `my_function` below, the fourth line is `print(i)`.

The easiest way to write code in LaTeX is using the **verbatim** environment, provided you don't need to typeset math in your code.

```
def my_function(n):
    i = 1
    while i < n:
        print(i)
        i = i + 1
    print("Done!")
```

4.3 (NEW!) Structuring proofs: headings and framed sections

When you're writing a longer proof, you may want ways of dividing up sections of your proof so that they're easier for both you and your reader to understand. One way to do this is to use section headers or text styling; we illustrate the latter below.

> **Case 1:** Assume .... We will prove ...
>
> *[Proof body for Case 1...]*
>
> **Case 2:** Assume .... We will prove ...
>
> *[Proof body for Case 2...]*

Another technique is to use the `framed` environment to put a box around a part of a proof. For example:

> Here is some text that has been framed. You could put a part of a proof in here to make it as separate from the other sections of your proof.

5 Resources

- An online LaTeX editor (easiest to get started): `https://www.overleaf.com/`. Also lots of great tutorials on here!
- Download LaTeX on the official LaTeX webpage: `http://latex-project.org/`.
- A relatively comprehensive introduction to LaTeX. **Highly recommended.** `http://ctan.mirror.rafal.ca/info/lshort/english/lshort.pdf`.
- A LaTeX wiki. Most Google searches lead here. `http://en.wikibooks.org/wiki/LaTeX`.
- An amazing application of machine learning. Use it to find commands based on the symbol. `http://detexify.kirelabs.org`.
- A graphical LaTeX editor (must download software) `http://www.lyx.org/`.