A SECOND LATEX EXAMPLE

TERRY BRIDGMAN

Abstract. This example file will introduce you to some additional features of LaTeX. Your project writeups don't need an abstract, but if you later write a paper, you'll need it.

1. Additional math mode information

When you bold something and within it have a math mode command, the math symbols do not become bold. This is usually not an issue because boldface is not the most common of typefaces, but can be annoying in things like titles, chapter headings, and subsection headings (note section headings are not in bold). The "proper" way to boldface your symbols is to enclose them in \boldsymbol{}. Example: **The variable x will not be bold here. However, we can make $\boldsymbol{x}$ bold.**

1.1. Display math mode.

Some formulas or other mathematical expressions are long and complicated, or they just have a lot of subscripts and superscripts, and they don't work well as in-line text. For those we'd like to center them on their own line. You could use the "center" environment with math mode, but it is more convenient to use what's called *display math mode*. This has the added advantage that many symbols are designed to look different (and nicer) in display mode than in in-line mode.

For example, let's typeset a sum. In-line, we have $\sum_{i=1}^{n} \frac{i_2}{2}$. With the centering and ordinary math mode, we would get

$\sum_{i=1}^{n} \frac{i_2}{2}$

Display mode is entered by using \[and left with \]:

$$\sum_{i=1}^{n} \frac{i_2}{2}.$$

Date: May 11, 2009.

Acknowledgments to all resources found via Google.

Much nicer. If you don't want it on its own line but you do want the nice typesetting, you can use "displaystyle": $\displaystyle\sum_{i=1}^{n}\frac{i_2}{2}$. Note the extra pair of braces enclosing the entire mathematical expression, and also how it messes up the spacing of the paragraph. I generally use "displaystyle" only in the context of an exam, where I want to typeset some isolated bit of math with the nice display format but have it be on the left side of the page.

Suppose you're in display math mode (or regular math mode) and you want to include some text. If you just type it in, it will get italicized and the spacing will look a little funny:

$$f(x) = 5 unless otherwise noted.$$

Here we can use \text{} to enclose the text:

$$f(x) = 5\text{unless otherwise noted.}$$

You'll note math mode enforces its own spacing rules, which are good for mathematical expressions but not so much for regular typing. We can override by adding the spaces we want inside the text command.[1]

$$f(x) = 5\text{ unless otherwise noted.}$$

1.2. **Limit and other math operators.** It is nice to have lim, max, cos, and other word-like math operators be spaced correctly, play nicely with things like subscript, and be typeset in non-italicized script. We could escape from math mode and just type them, but that is a kludge that generally still screws up the spacing. Instead, for most widely-used commands like that there are math operator commands: just precede the word with a backslash:

$$\lim_{x\to\infty}\frac{1}{x^2} = 0.$$

Note in display math mode it even puts the subscript underneath the limit sign, not next to it. To define your own math operators (there's no domain operator predefined, for instance), use DeclareMathOperator commands in the header. The example in this file is so: $W_e = \operatorname{Dom}\varphi_e$. Now, if we give it subscripts, even in display math mode it will put them off to the right. If we want them underneath we have to use

[1] Note that the text command requires the package amsmath. If you're using text and words together in a display format, you should really put everything inside the text command: \[\text{ . . . }\], with the mathematical portions enclosed in dollar signs. This seems like a weird twisted hack, but it makes sure the spacing works exactly as it should. If you're not so picky, though, what I've put here will work fine.

DeclareMathOperator* in the header (the * goes between the word “Operator” and the first opening brace).

You have to be a little careful defining new commands, because if the name is already taken the compiler will choke. In the operator case choosing a different name is the only solution I know of.

2. Theorem environment

To write a theorem and have it be nicely typeset, use the theorem environment.

Theorem 2.1. *Puppies are cute.*

The proofs can go in the proof environment, which automatically adds the box at the end.

Proof. If you have ever seen a puppy, you know the theorem is true. If you have not, you can look them up. □

The “label” command allows us to refer to the theorem automatically. This way if we add or subtract sections or theorems previous to the theorem in question, the number will be automatically updated (it takes two rounds of typesetting to update everything: one updates the list of labels and the next updates the references to them). Labels may also be applied to sections, as I have done with §3 below. To refer to a theorem you use “ref”: by Theorem 2.1 we know Fido, being a puppy, is cute.

There are all sorts of other non-theorem theorem environments we can define in the header, including claim, lemma, proposition, etc. These are accessed by replacing “theorem” in the begin…end commands with the appropriate word. The theoremstyle command in the header separates theorem environments that go in italics from those that go in plain text (largely a personal preference decision), such as definitions.

Definition 2.2. A *fish* is a cold-blooded water-dwelling vertebrate.

Theorem 2.3 (Fish ’n’ Chips Theorem)**.** *Fish and chips served wrapped in newsprint tastes better than fish and chips served on a paper plate.*

The square brackets are used in general for titles of theorems or for attributions (one may put citations into the brackets; see the bibliography section below). The square bracket option for the proof environment entirely replaces the word “Proof”:

Proof of Theorem 2.3. The theorem is obvious to anyone who has had opportunity to compare. □

3. Spacing and sizing

It is nice to let LaTeX adjust spaces and sizes, in general, because it does a better job than we poor humans. However, sometimes you want to increase or decrease the size manually or insert extra spaces (or take them out). The commands *bigskip, medskip*, and *smallskip* give additional vertical space between paragraphs; you can guess how much (to use, just precede them with a backslash). If you want more space than that, use \vspace{amount}, where the amount is a number followed by a unit of measure, which can be inches (in), centimeters (cm), the width of the letter 'n' in the current font (en), or the height of the letter 'x' in the current font (ex). Probably there are others, but those four should do. The advantage of the latter two is they scale up or down automatically should you decide to change your font size or style. Horizontal space may be made by replacing 'vspace' with 'hspace'. For real confusion, you can make the amount of vertical or horizontal space negative, to squish things together.

Small amounts of horizontal space may be inserted using a backslash followed by a comma, greater-than sign, or semicolon, which give increasing amount of space in that order. A small negative horizontal space is obtained with \! – I find this useful in situations such as using $\downarrow$ and $\uparrow$ for halting and diverging functions, as they end up too far away from the function without one or two negative spaces.[2]

You can force something to the end of the line or the end of the page by preceding it with \hfill and \vfill. These may not do exactly what you expect; you may have to hard-code a line break (\\ or just leaving a blank line to start a new paragraph) or a page break (\newpage) after the text you want at the end of the line or bottom of the page.

One final note, relevant mostly to making exams. If you want to space two pieces of text at the top and halfway point of a page, the easiest way to do so is [text 1], blank line, \vfill, blank line, [text 2], blank line, \vfill, blank line, \mbox{} at the bottom. The mbox creates something LaTeX sees as having space, so it gives vfill something to push against. Without the mbox at the bottom the second segment of text will not end up in the middle, but at (or perhaps merely near) the bottom.

[2] There are better ways to do this, which adjust better to different font size, etc., but I won't burden you with them right now, especially as I barely know them myself. They have to do with making LaTeX treat a symbol as though it is, say, a close-parenthesis, and space it accordingly.

4. Enumerate and itemize

Perhaps you want to enumerate with lowercase or capital Roman numerals instead of Arabic numerals, or perhaps you'd like to use lowercase or uppercase letters. The way to do that in the enumerate environment is to follow your \begin{enumerate} with [\upshape (i)], or replace (i) by i., or likewise use I, a, or A. This requires the *enumerate* package.

I. This is item 1.
II. This is item 2.

The upshape command is not strictly necessary but allows you to maintain upright counters even in italicized environments like theorem, which looks better.

In the itemize environment you can override the bullet points with other words or symbols by putting them in square brackets immediately after the \item tag:

1. This is item 1.
⇝ This is item 2.

5. Bibliographies

There are all sorts of methods for managing bibliography databases. Our projects will have small bibliographies, so we'll just do it by hand (unless you're feeling ambitious). In the bibliography environment, placed at the end of the document (though before \end{document}), you have a bunch of bibitems. In the braces, you put the reference key, which allows you to then cite by number: The *(Not So) Short Introduction to LaTeX* [1] is a nice online reference for LaTeX. Note that however you format the reference in the bibitem is how it will appear in the typeset document. Also, just like if labels on theorems change, it takes two rounds of typesetting to change the references if you add or subtract bibliography entries.

Finally, the "noindent" command does exactly what it says it should.

You can now typeset most of the course notes. Anything you still want to know how to do for your presentation write-up we will address individually.

References

[1] Oetiker, Tobias. *The (Not So) Short Introduction to LaTeX.* Available from `http://www.ctan.org/tex-archive/info/lshort/english/`