Workshop 1: Introduction to LaTeX

Dan Parker and David Schwein

Sunday, September 21, 2014

Welcome to the first of the Science Center's LaTeX workshops! By the end of today's workshop, you will be able to write a basic LaTeX document containing mathematics.

We assume that you have a working installation of LaTeX and a LaTeX editor. If you do not, please consult the installation guide.

1 Introduction

1.1 Questions and Answers

What is LaTeX?

LaTeX is a document preparation system, the de facto standard for mathematics, physics, and computer science.

More technically, LaTeX is a set of macros – extra commands – for the programming language TeX. Donald Knuth created TeX in the seventies because he was unhappy with the typography in one of his *Art of Computer Programming* books. He designed the program specifically for typesetting mathematics, and it was widely adopted in the years following its release. Later, in the eighties, Leslie Lamport wrote LaTeX to extend TeX's mostly low-level functionality. Today, LaTeX has largely superseded TeX.

Why should I use LaTeX?

1. LaTeX documents are beautiful. Their typography is outstanding, especially as compared to something like Microsoft Word.

2. LaTeX can make almost any document you can imagine and many others besides.

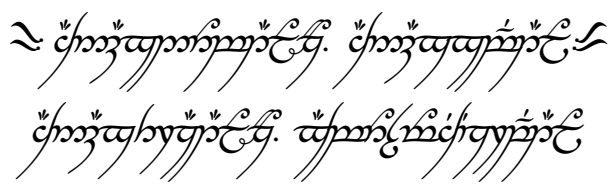
3. LaTeX is free, fast, and has virtually no bugs.

LaTeX is not for everyone, even among academics. While designing these workshops we found, for instance, that the chemistry department at Brown

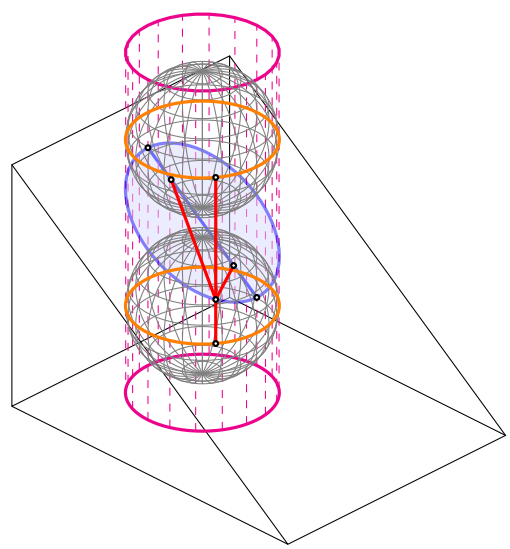
does not care for LaTeX. But mathematicians should certainly use LaTeX, mainly because people will not take you seriously if you don't.[1] For the same reason, papers in other math-heavy scientific disciplines, such as physics and computer science, should also be prepared using LaTeX.

What can LaTeX do?

TeX is a Turing-complete programming language, which basically means that it can do anything a computer can do. Steve Hicks, for example, has written a program in TeX to control a Mars rover.[2] For the rest of us mortals, the Comprehensive TeX Archive Network (CTAN, at `ctan.org`) maintains a database of over 4,700 LaTeX packages, which add extra functionality to the program. (We'll discuss packages in greater detail next time.) One of our favorites is the `tengwarscript` package, which you lets you type the Elvish script Tengwar from J.R.R. Tolkien's *Lord of the Rings*:



A more useful package is `tikz`, used for creating technical diagrams. We'll discuss `tikz` in the final workshop.



Nevertheless, LaTeX is usually used only for typesetting documents.

[1] Scott Aaronson claims that you can catch 60% of wrong mathematical breakthroughs by checking if they were written in TeX. See `http://www.scottaaronson.com/blog/?p=304`

[2] See `http://sdh33b.blogspot.dk/2008/07/icfp-contest-2008.html`

I have no experience with computer programming. Can I learn LaTeX?

Yes! Do not be intimidated: LaTeX and TeX were designed for ease of use and most command have obvious names so that you can guess them. With that said, when you write a document in LaTeX you are programming a computer, and the process for making a document with LaTeX is slightly more complicated than you are probably used to. There are more files for each document, and you can't immediately see the changes to your document.

How do you pronounce LaTeX?

There are two ways to pronounce LaTeX: LAY-tek or LAH-tek. We use the first pronunciation and Brown's computer science department tends to use the second. Everyone agrees, however, that you should not pronounce LaTeX like the word *latex*: this is not that kind of workshop. When you write the word *LaTeX*, remember to capitalize it as we have.

1.2 LaTeX's Philosophy

Document preparation systems come in two kinds: what-you-see-is-what-you-get (WYSIWYG) and what-you-see-is-what-you-mean (WYSIWYM). Microsoft word is a WYSIWYG editor: the file you edit looks the same as the document that comes off the printer. LaTeX is a WYSIWYM editor: the file you edit – a `.tex` file – stores information *about* the document, which is then typeset by the program to create the final document – by default, a `.pdf` file.

WYSIWYM editors nearly always produce superior documents because they reflect the natural demarcation in publication between composing a document and typesetting it, which was historically done by hand using movable type. WYSIWYG editors sacrifice typographic quality for speed: Microsoft Word can typeset your document as you type because it cuts typographic corners, On the other hand, TeX cannot synchronize changes in the `.tex` file and the `.pdf` file because the algorithm it uses to determine line justification and other typographic aspects of the document takes a little too long, around a second or less.[3]

The point is that using LaTeX forces you to focus on the content of your document instead of its form. You should not worry about choosing a font or sizing section headers, for instance: LaTeX makes all of those decisions for you, and it usually does an excellent job of it. Ceding stylistic control to LaTeX will be hard at first if you are accustomed to a WYSIWYG editor; do it anyway, and relearn old habits. With that said, it is possible, though not recommended, to customize any aspect of a LaTeX document. In this workshop we will spend little time discussing how to modify stylistic features of documents, such as typeface or section styles. Remember,

The number-one error LaTeX users make is focusing on form over content.

[3] I lied: there actually are WYSIWYG LaTeX editors. But why would you do that to yourself?

2 Bare-Bones LaTeX

2.1 An Outline of How LaTeX Works

Before you make your first document in LaTeX, it will be helpful to have an idea of how the parts of the system fit together. To make a document with LaTeX, you edit a file ending in the `.tex` file extension. This file contains plain text – it is essentially a `.txt` file. When you tell LaTeX to typeset the `.tex` file it creates a `.pdf` file – your document – and a few auxiliary files we won't worry about in today's workshop. This means that whenever you modify the `.tex` file, you must typeset the document again to see the changes in the PDF file.

2.2 Your First LaTeX Document

Go to your favorite directory and create a new folder, which will contain the LaTeX document you are about to make. (We encourage you to place every LaTeX file in its own folder because the compiler makes auxiliary files that can clutter the system.) Next, open your LaTeX editor and save a new file. Copy and paste the following text into the file.

```
\documentclass{article}

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

Your LaTeX editor has a button that says "typeset", or perhaps you know the shortcut for typesetting (usually a variation of ⌘+T; you should learn it). Press the button and typeset. If you see a document appear, congratulations! Your LaTeX system works and you have made your first document with it.

If you have previous experience with LaTeX you may have expected the `.tex` file to contain more commands. Many new LaTeX users make documents using templates from the internet, which contain custom formatting commands and tend to be quite complicated. In the interest of simplicity we have tried to keep the documents we make in this workshop as simple as possible: you should understand every command.

In any case, every LaTeX document requires only two commands. The first of these is `\documentclass{article}`, which tells LaTeX that you will be creating an article. There are other document classes, such as `letter` or `book`, but article is the default and works well for most documents. Every LaTeX command has the same syntax as `\documentclass{article}`: a command begins with a backslash and any arguments it takes are surrounded in curly braces. The `\documentclass` command takes a single argument, and other commands take no arguments or more than one.

The second requirement is the two lines containing `\begin{document}` and `\end{document}`. The first tells LaTeX to begin making the document; the second tells LaTeX that the document is finished. Consequently, any text appearing

between these two lines will be incorporated into the document and any text appearing after `\end{document}` will be ignored.

The space between `\documentclass{article}` and `\begin{document}` is called the *preamble* or the *topmatter*. It contains commands that modify global document parameters, such as the typeface or the appearance of headers and footers.

2.3 Spacing

To make a space between two words, place a space between them. (No surprises there.) One space is enough because LaTeX condenses interword spaces longer than two characters: typesetting `O  brave  new  world!` gives "O brave new world!" Since LaTeX treats a single line-break like an interword space, you can also make a space between two words by placing the second word on the next line after the first.

To make a block of text into a new paragraph, place either an empty line or the command \\ before it. For example, typesetting either

```
But they think the attacks on the president are unfair,
so they rally round.

The wider public's views are murkier.
```

or

```
But they think the attacks on the president are unfair,
so they rally round. \\
The wider public's views are murkier.
```

gives

But they think the attacks on the president are unfair, so
they rally round.

The wider public's views are murkier.

2.4 Sections

It's usually a good idea to divide a long document into sections, subsections, and even subsubsections, and LaTeX has nice built-in commands to do so. They take a single argument, the name of the section.

```
\section{...}
\subsection{...}
\subsubsection{...}
```

The commands above number the part of the document they create just as the sections of this document are numbered. Automatic numbering is one advantage of using LaTeX over other document preparation systems. Imagine typing a 100-page-long thesis in Microsoft Word and adding one section at the

beginning: you would have to renumber all of the sections to compensate. If you had used LaTeX, the sections would have been renumbered automatically.

Perhaps you are not impressed by automatic numbering because you prefer unnumbered sections. To remove the numbers, use the starred versions of the commands.

```
\section*{...}
\subsection*{...}
\subsubsection*{...}
```

2.5 Titles

There are two steps to give your document a title: tell LaTeX what to put in the title, and tell LaTeX to typeset the title. For the first step, LaTeX needs to know the title, author, and date of the document: in the preamble, use the commands `\title{...}`, `\author{...}`, and `\date{...}`. For example, the preamble to the `.tex` file for this document contains

```
\title{Workshop 1:\\ Introduction to \LaTeX}
\author{Dan Parker and David Schwein}
\date{\today}
```

The command `\today` prints today's date; the command `\LaTeX` prints LaTeX. For the second step, tell LaTeX to make the title by writing `\maketitle` immediately below `\begin{document}`.

2.6 Font Styles

Most typefaces – including Computer Modern, the default for LaTeX – comprise a variety of weights, italicizations, and other styles. You can access them with the following commands.

`\textit{italic}`	*italic*
`\textbf{bold}`	**bold**
`\textsc{small caps}`	SMALL CAPS
`\texttt{teletype}`	`teletype`

You can also mix styles.

`\textbf{\textit{bold-italic}}`	***bold-italic***
`\textsc{\textit{small caps-italic}}`	*SMALL CAPS-ITALIC*
`\textsc{\texttt{small caps-teletype}}`	`SMALL CAPS-TELETYPE`

To emphasize text – like *this* – use the command `\emph{...}`, which compensates for the style of the surrounding text.

`\textit{\emph{She} was unhappy}`	She *was unhappy*
`\texttt{\emph{He} was not}`	*He* `was not`

2.7 Special Characters

LaTeX reserves certain characters for special use, such as a dollar sign $ for displaying math. Printing these characters is usually straightforward.

2.7.1 Hyphens and Dashes

Hyphens connect the parts of a compound word and are printed with a `-` character.

`a non-negotiable ten-step plan`	a non-negotiable ten-step plan

There are two types of dashes: en dashes – which are about as wide as the letter "N" – and em dashes—which are about as wide as the letter "M", or twice as wide as an en dash. An en dash is printed with `--`; an em dash is printed with `---`.

`pp. 126--128`	pp. 126–128
`Wait---come back!`	Wait—come back!

2.7.2 Quotation Marks

The quotation characters on your keyboard will not typeset the way you expect them to.

`"This" looks bad`	"This" looks bad

The proper way to quote is to use `` ``` `` for the left marks and `''` for the right marks.

`` ``A `feisty' dog'' ``	"A 'feisty' dog"

On a QWERTY keyboard, the key for `` ` `` is left of [1] and the key for `'` is right of [;].

2.7.3 Comments

To prevent LaTeX from typesetting text, place a % before it. The entirety of the line following % will be ignored.

`Where? % Here`	Where?

2.7.4 Ellipses

The technical term for three periods in succession is *ellipsis*. To print an ellipsis use the command `\dots`.

`This is ugly...`	This is ugly...
`This is better\dots`	This is better…

2.7.5 Miscellaneous Characters

Finally, LaTeX reserves a few special characters for its own purposes. You can access most of these characters by placing a \ before them.

```
\{ \} \% \$ \& \_ \#
```

{ } % $ & _ #

There are, of course, exceptions. To print \ use the command `\textbackslash`. To print ˜ or ˆ use the commands `\~{}` and `\^{}`. What happens if you remove the `{}`?

2.8 Lists

LaTeX has commands for making two types of lists: bulleted lists and numbered lists. To make either type of list, you must first tell LaTeX to enter the appropriate *environment*. Roughly speaking, an environment is a portion of your document that LaTeX treats differently from the text of the document. For instance, LaTeX includes environments for centering text, including figures, or making tables.

To begin an environment, use the command `\begin{<environment>}`; to end an environment, use the command `\end{<environment>}`. Does this sound familiar? It should, because we use `\begin{document}` to begin every LaTeX document and `\end{document}` to end every document.

To make a bulleted list, use the `itemize` environment. To make a new bullet, use the command `\item`.

```
\begin{enumerate}
 \item Me
 \item Myself
 \item I
\end{enumerate}
```

1. Me
2. Myself
3. I

To make a numbered list, use the `enumerate` environment. Again, the command `\item` adds a new item to the list.

```
\begin{itemize}
 \item We
 \item Ourselves
 \item Us
\end{itemize}
```

- We
- Ourselves
- Us

To make a list with more than one level, nest list environments inside of each other. How many of these environments does LaTeX let you nest?

```
\begin{enumerate}
 \item Ends in \emph{gry}
  \begin{itemize}
   \item \emph{angry}
   \item \emph{hungry}
   \item Anything else?
  \end{itemize}
\end{enumerate}
```

1. Ends in *gry*
   - *angry*
   - *hungry*
   - Anything else?

3 Mathematics

LaTeX was designed specifically for writing math: there is a command to print virtually anything you can write. And because there are many mathematical symbols you can write, there are quite a few commands for printing mathematics. We don't want to bombard you with long lists of commands, so instead we're going to explain how to write mathematics in LaTeX by explaining the three most common mathematics environments. During next week's workshop, we'll cover mathematics in more detail. Today is a gentle introduction.

The three types of mathematics environments differ only in the way they are typeset. Any command available in one math environment is available in any other.

Since Δ interchanges the two groups inside the parentheses it is enough to compute the first of them, i.e.,

$$H^1_{\text{unr in } \Sigma-\mathfrak{p}}\left(\mathbf{Q}_\Sigma/L,\ K/\mathcal{O}(\nu)\right). \tag{4.2}$$

The inflation-restriction sequence applied to this gives an exact sequence

$$\begin{aligned} 0 \quad &\to \quad H^1_{\text{unr in } \Sigma-\mathfrak{p}}\left(L(\nu)/L,\ (K/\mathcal{O})(\nu)\right) \\ &\to \quad H^1_{\text{unr in } \Sigma-\mathfrak{p}}\left(\mathbf{Q}_\Sigma/L,\ (K/\mathcal{O})(\nu)\right) \\ &\to \quad \operatorname{Hom}\left(\operatorname{Gal}(M_\infty/L(\nu))\,,\ (K/\mathcal{O})(\nu)\right)^{\operatorname{Gal}(L(\nu)/L)}. \end{aligned} \tag{4.3}$$

The first term is zero as one easily checks using the divisibility of $(K/\mathcal{O})(\nu)$. Next note that $H^2\left(L(\nu)/L,\ (K/\mathcal{O})(\nu)\right)$ is trivial. If $\nu \not\equiv 1(\lambda)$ this is straight-

Figure 1: The three types of equations in their natural habitat.[4]

1. Inline formulas: formulas inside a sentence. Usually short and not containing any tall symbols, like sums or integrals.

2. Displayed formulas: formulas on their own line with spacing before and after. Used for long, tall, or important equations.

3. Aligned formulas: several formulas, each on its own line, with a certain symbol (often =) in the same place in each line. Used for expressions that are too long for a single line, or computations with multiple steps.

3.1 Inline Mode

Let's begin with inline formulas, the most commonly used of the three types.

[4]This is an excerpt from Wiles' proof of Fermat's Theorem. Don't worry if it's completely unintelligible to you – we can't make sense of it either. The point is to see the three types of formulas in action.

```
Let $x = r\cos\theta$
and $y = r\sin\theta$.
Then $r = \sqrt{x^2 + y^2}$
and $\theta = \arctan(y/x)$.
```

Let $x = r\cos\theta$ and $y = r\sin\theta$. Then $r = \sqrt{x^2 + y^2}$ and $\theta = \arctan(y/x)$.

What's going on here? To make an inline formula, enclose the expression in `$`. As we promised, command names make sense: for instance, the command for the greek letter θ is `\theta`. Anything between the dollar signs is interpreted as math, and not as regular text: in math mode, TEX behaves differently.

- All letters are automatically italicized. This gives the reader a visual cue that they're looking at a variable instead of a normal letter.
- Commands for mathematical symbols, such at fractions and greek letters, can be used. Most of these commands will not work outside of math mode.
- The spacing is tighter. For instance, spaces between letters are removed.
- Blank lines are not allowed.

There are thousands of math commands in LATEX, enough to make any symbol you could want and many more you probably don't. If you know what a symbol is called but don't know the command for it, you can find it in one of the many lists of LATEX math symbols available on the internet.

- Brown's CS department has a list of basic symbols, at the address `cs.brown.edu/about/system/software/latex/doc/symbols.pdf`.
- A comprehensive list of symbols in LATEX – 164 pages – is available at `mirror.hmc.edu/ctan/info/symbols/comprehensive/symbols-a4.pdf`.

If you don't know what a symbol is called then try using the website Detexify, which has you draw the symbol and guesses the corresponding LATEX command.

Here is a table of the most common math commands.

`$a^b$`	`$a_b$`	`$\log(x)$`	`$\alpha$`	`$\sqrt{x}$`	`$\frac{a}{b}$`
a^b	a_b	$\log(x)$	α	$\sqrt{x}$	$\frac{a}{b}$

3.2 Display Mode

Large symbols, such as sums or integrals, are usually too large to fit into a single line – such as $\sum_{n=0}^{\infty} \frac{1}{n^2}$, which is so large that LATEX has to add space around the line to compensate. Moreover, some equations are so important that they should be separated from the body of the text and placed on their own line. Displayed math mode is the mode of choice for expressions like these. Use `\[` to enter displayed math mode and `\]` to leave it.

```
\[
\int_0^1 f(x)\,dx =
\lim_{n\to\infty}
\sum_{i=0}^{n-1}
\frac{1}{n}
f\left(\frac{i}{n}\right)
\]
```

$$\int_0^1 f(x)\,dx = \lim_{n\to\infty} \sum_{i=0}^{n-1} \frac{1}{n} f\left(\frac{i}{n}\right)$$

Although it is not strictly necessary to place `\[` and `\]` on their own lines, this is one of many practices that make it easier to read LaTeX code.

3.3 Aligned Mode

To do *aligned mode*, we need to make a modification to our document's preamble. Add the line `\usepackage{amsmath}` anywhere before the `document` environment begins. To enter align mode, go into the environment `align*`.[5] To use aligned mode, type equations as you would for inline or display mode, but with the following modifications:

- Place `&` before the symbols you want to have vertically aligned.
- Place `\\` at the end of every line.

```
\begin{align*}
(a+b)^2
&= (a+b)(a+b) \\
&= a(a+b) + b(a+b) \\
&= a^2 + ab + ba + b^2 \\
&= a^2 + 2ab + b^2.
\end{align*}
```

$$\begin{aligned}
(a+b)^2 &= (a+b)(a+b) \\
&= a(a+b) + b(a+b) \\
&= a^2 + ab + ba + b^2 \\
&= a^2 + 2ab + b^2.
\end{aligned}$$

[5] Why `align*` instead of `align`, you ask? Try using the `align` environment and see what happens.