A Beginner's Guide to LaTeX

written by David Xiao
(*edited by Sachin Ravi and Kevin Wayne*)

February 27, 2013

1 Introduction

LaTeX is a document markup language (created by Leslie Lamport) for the TeX typesetting system (created by Donald Knuth). It is very widely used in academia, especially by scientists and mathematicians. More recently, it has emerged as a web starndard for displaying mathematics (e.g., MathJax, Piazza, WordPress, and Coursera). This document is intended for people who have never used LaTeX and want a quick crash course to get started. For a more comprehensive introduction, check out

`http://ctan.tug.org/tex-archive/info/lshort/english/lshort.pdf`.

2 How to find and use LaTeX?

Most Princeton university clusters have LaTeX installed. If you prefer, you can install it on your home system; it is easy to find via your favorite search engine. I recommend using front-end programs such as TeXworks or TeXShop. Alternatively, you can use a text editor such as `emacs` (and its LaTeX-mode) and the command line. For simplicity in exposition, I will assume that you are compiling from the command line for the rest of the guide.

3 Basic rules

Basic LaTeX is just text with typesetting commands. Typesetting commands are usually preceded by "\", and any arguments are usually placed inside curly braces "`{}`".

LaTeX wraps text in adjacent lines as if they were part of the same paragraph. To start a new paragraph, insert an extra "return":

Source:

```
This is
one paragraph.

This is another.
```

Output:

This is one paragraph.

This is another.

To get a newline without starting a new paragraph, use `\\`.

To get a comment, use the percent sign `%` at the beginning of a line. The rest of that particular line will be commented out.

4 Creating a new document

The most basic document has three parts:

```
\documentclass{article}

\begin{document}
This is
one paragraph.

This is another.
\end{document}
```

In practice, you typically create a new document by copying an existing document, deleting the stuff between the begin and end document commands, and changing some information in the header. In COS 423, you will use the following template:

`http://www.cs.princeton.edu/courses/archive/spring13/cos423/problem0-1.tex`.

5 Compiling

Suppose our file is named `problem0-1.tex`. To compile it, invoke `pdflatex problem0-1.tex` in your Unix shell. This will compile the file, assuming there are no errors.[1] If there are errors, you can quit the compiler by hitting "`x`" and then enter. Unfortunately, LaTeX compiler errors are often not very unhelpful in determining the nature of the problem but they usually identify on which line the error occurred.

Once it successfully compiles, you will get a file named `mylatexfile.pdf` that you can open and view.

6 Organization

It is important to organize your document.

6.1 Sectioning

There are two sectioning commands that will be useful for you: `\section{Name of section}` and `\subsection{Name of subsection}`.

6.2 Tables

You can put stuff into tables by using the `tabular` environment. For example:

Source:

```
\begin{tabular}{r|cl}
1st column & 2nd column & 3rd column\\
\hline
a & b & c
\end{tabular}
```

Output:

1st column	2nd column	3rd column
a	b	c

Note that the command is called `tabular` and *not* `table`. Important points:

[1] One quirk is that you may have to invoke `latex` twice if you are using labels or references. See Section 6.4.

- The "`{r|cl}`" after the tabular `\begin{tabular}` indicate the alignment of the three columns: right, center, and left. This is mandatory as it specifies the layout of the table. For more columns, type more alignment commands, e.g. for a table with 5 columns all aligned to the right, you would use `rrrrr`.
- The vertical bar `|` between the `r` and `c` indicates that a vertical line should be drawn between those columns.'
- The `&` separates the columns in the body of the table.
- A `\\` signifies the end of each line of the table.
- The command `\hline` means that a horizontal line should be inserted.

6.3 Lists

You can put stuff into ordered and unordered lists by using the `enumerate` and `itemize` commands, respectively. For example:

Source:

```
Unordered list:

\begin{itemize}
\item This is one item.
\item This is another.
\end{itemize}

Ordered list:

\begin{enumerate}
\item This is the first item.
\item This is the second.
\end{enumerate}
```

Output:

Unordered list:

- This is one item.
- This is another.

Ordered list:

1. This is the first item.
2. This is the second.

6.4 Labels and references

It is useful to refer to the section number at times. This may be done by using the `\label{labelname}` command. Place this right after you start a section. Then, you may refer to the section number by using `\ref{labelname}`. This will also be useful to refer to math equations.

LaTeXcreates and uses a bunch of auxiliary files. Be sure to invoke `latex` twice to compile a file that has labels and references (or if thoses labels and references have changed since the last compilation).

7 Math

The primary reason to use LaTeX is its ability to typeset mathematical expressions.

7.1 Math mode

Math expressions are separate from text in LaTeX. To enter a math environment in the middle of text, use the dollar sign `$`, for example `$F = ma$` produces $F = ma$. Everything between the two `$` signs will be treated as a math formula.

To type a math expression that is on its own line and centered, delimit it with `\[` and `\]`:

Source:

```
The following is an important equation:
\[ E = mc^2 \]
```

Output:

The following is an important equation:

$$E = mc^2$$

To give an equation a number and have it referable, use the `equation` environment and use a `\label` command:

Source:

```
The following is an important equation:
\begin{equation}
\label{emc}
E = mc^2
\end{equation}
Please memorize Equation \ref{emc}.
```

Output:

The following is an important equation:

$$E = mc^2 \tag{1}$$

Please memorize Equation 1.

To typeset several equations together and have them properly aligned, use the `align` environment:

Source:

```
Some important equations:
\begin{align}
\label{einstein}
E & = mc^2 \\
\label{newton}
F & = ma \\
\label{euler}
e^{i \pi} & = -1
\end{align}
```

Output:

Some important equations:

$$E = mc^2 \tag{2}$$
$$F = ma \tag{3}$$
$$e^{i\pi} = -1 \tag{4}$$

The equations are aligned along the `&` and each line is terminated by `\\`. To suppress the equation numbering (i.e. if the equations won't be referred to) use `align*` instead of `align`.

7.2 Writing math expressions

I will only go over a few common mistakes and hard-to-find expressions regarding how to write math expressions. It is quite intuitive otherwise, you can figure most things out quickly with trial and error. All expressions in math mode may be nested within each other arbitrarily.

- Superscript and subscript are done using `^` and `_` characters. Note that if you want multiple characters in the super/subscript then you need to surround them with curly braces: `$e^i\pi = -1$` gives $e^i\pi = -1$ whereas `$e^{i\pi} = -1$` gives $e^{i\pi} = -1$.
- Fractions are done using `$\frac{1}{2}$` which gives $\frac{1}{2}$.
- To do a binomial coefficient, use `$\binom{n}{k}$` which gives $\binom{n}{k}$.
- Modular arithmetic can be written using the `\pmod{n}` and `\bmod{n}` commands. The first puts parentheses and a lot of space around the `mod` and the second does not.
- $\forall$ and $\exists$ are written as `\forall` and `\exists`.
- $\neq$, $\geq$, and $\leq$ are written as `\neq`, `\geq`, and `\leq`.
- $\cdot$ (e.g. for multiplication) is `\cdot`.
- $\ldots$ is produced by `\ldots` and $\cdots$ by `\cdots` (notice the relative height).

- $\circ$ is `\circ`.
- $\cup$, $\cap$, and $\setminus$ are `\cup`, `\cap`, and `\setminus`.
- Large $\cup$ and $\cap$ signs that behave like summations (see below for summations) are written as `\bigcup` and `\bigcap`.
- $\mathbb{Z}$, $\mathbb{R}$, etc. are produced using `\Z`, `\R`, etc.
- $\mathbb{E}$ is produced with `\Exp`.
- $\mathcal{O}(n)$, $\Omega(n)$, $\Theta(n)$ are produced with `\BigOh{n}`, `\BigOmega{n}`, and `\BigTheta{n}`.
- $\mathbf{P}$ and $\mathbf{NP}$ are produced using `\P`, `\NP`, etc.
- ℓ (as opposed to l) is produced with `\ell`.
- $\approx$ is produced with `\approx`.
- $\hat{x}$ and $\bar{x}$ are done with `\hat{x}` and `\bar{x}`. A longer bar may be written using `\overline{\SAT}`, which produces $\overline{SAT}$.
- x' and x'' are done with `x'` and `x''`, respectively.
- ε (as opposed to ϵ) may be written with `\eps`.
- $\in$ and $\notin$ are written as `\in` and `\notin`.
- Negations may be done with `\not`, for example `\not\geq` gives $\ngeq$.
- $\{0,1\}$ is abbreviated as `\zo`.
- The probability sign Pr is defined as `\Pr`.
- $\{\}$ are done with `\{` and `\}`.
- To draw parentheses, braces, or brackets that resize to match the contents, use `\left` to precede the left one and `\right` to precede the right one:
  Source:

  ```
  \[ \Pr\left[\sum_{i=1}^k X_i > c \right] \leq 2^{-\Omega(c^2 k)} \]
  ```

  Output:

  $$\Pr\left[\sum_{i=1}^{k} X_i > c\right] \leq 2^{-\Omega(c^2 k)}$$

- Arrays are similar to tables, except they must be used in place of tables when in math mode: instead of using `\begin{tabular}` and `\end{tabular}` use `\begin{array}` and `\end{array}`. Again, you must give a column specification for how the columns are to be laid out.
- Spacing is very different in math mode so text in the middle of a formula is set strangely. If you want to have text in the middle of the formula, use the `\text{some text}` command. For example,
  `$\P \neq \NP \text{ implies that } \SAT \notin \P$` produces $\mathbf{P} \neq \mathbf{NP}$ implies that $SAT \notin \mathbf{P}$.

- Summations and products are done using `\sum` and `\prod` respectively. Parameters can be given for the summation/product as well:

Source:

```
\[ \sum_{i=1}^\infty \frac{1}{2^i} = 1 \]
```

Output:

$$\sum_{i=1}^{\infty} \frac{1}{2^i} = 1$$

- Piecewise functions may be defined using the `piecewise` environment:

Source:

```
\[ f(x) = \begin{piecewise}
1 & \text{ if } x = 0 \\
0 & \text{ if } x \neq 0
\end{piecewise} \]
```

Output:

$$f(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{if } x \neq 0 \end{cases}$$

- You may define new commands using `\newcommand{\commandname}{definition}`. This is essentially a macro, so that whenever `\commandname` appears, the text of `definition` is inserted in its place. If `\commandname` is already taken, either use a different name or use `\renewcommand{...}{...}` to overwrite the old definition.

- There are many many other symbols available. You can search for "latex symbols" online and come up with the references.

8 Writing pseudocode

It is often necessary to state pseudocode when dealing with specific algorithms. Two packages that are useful for writing pseudocode are the *algorithm* and *algorithmcx* packages. We use the *algorithm* environment so that the code is not across different pages. We use the *algorithmcx* package to write the specific pseudocode. An example is presented below.

Source:

```
\begin{algorithm}[H]
  \caption{Euclid's algorithm}
  \begin{algorithmic}
    \INPUT $a \ge 0$, $b \ge 0$
    \OUTPUT the gcd of $a$ and $b$
    \Function{gcd}{$x, y$}
      \If {$b = 0$}
        \Return $a$
      \Else $\;$
        \Return \Call{gcd}{$b, a \bmod b$}
      \EndIf
    \EndFunction
  \end{algorithmic}
\end{algorithm}
```

Output:

Algorithm 1 Euclid's algorithm

INPUT: nonnegative integers a and b
OUTPUT: the greatest common divisor of a and b

```
function GCD(x, y)
    if b = 0 then return a
    else return GCD(b, a mod b)
    end if
end function
```

For a more comprehensive review (if-statements, for-loops, while-loops, etc), check out `http://en.wikibooks.org/wiki/LaTeX/Algorithms#Typesetting_using_the_algorithmicx_package` and/or `http://get-software.net/macros/latex/contrib/algorithmicx/algorithmicx.pdf`.

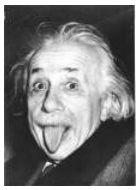
9 Figures

The easiest way to include figures in LaTeX is to draw the figure using some other software package (such as Powerpoint, Keynote, Illustrator, xfig, or Google drawings) and save it as a .pdf file (perhaps after cropping it to the desired size). We use the *graphicx* package to import it into our document. Suppose for example, we have an image pdf file saved in our working directory as "einstein.pdf". We can use the following code to include the image in our document.

Source:

```
\begin{figure}[p]
  \centering
  \includegraphics[width=0.3\textwidth]{einstein.pdf}
  \caption{Image of Einstein.}
  \label{fig:einstein}
\end{figure}
```

Output:



For more details about figure formatting, check out
`http://en.wikibooks.org/wiki/LaTeX/Importing_Graphics`.