An Introduction to LaTeX

1 Introduction

This document is meant to serve as a basic, informal introduction to the mathematical typesetting system known as LaTeX (usually pronounced "Lay-tek"). The idea of LaTeX is to allow professional researchers and students alike to type up various types of documents, which may or may not include mathematical symbols and equations, in an efficient manner. LaTeX is used by virtually all working mathematicians today, as well as by students and researchers in many other fields. It is easy to learn the basics, and one can very quickly learn to compose beautiful documents. There are virtually unlimited features available forLaTeX which allow users to easily customize documents without spending countless hours on minutia.

LaTeX works in two parts. First, the user types into a raw file (with file extension .tex). There are special LaTeX editors which handle this part. This basically looks like a plain text file. For those of you who have used any programming language, this resembles raw code. Then, using a LaTeX compiler, the .tex file is converted into a PDF file which has all of the features included. There are commands available which can alter virtually every aspect of the document. The user types regular text, along with commands, in the .tex file (most commands start with a backslash "\") which the compiler translates into nicely typeset mathematics. For example, if the user types

```
$\frac{3}{5}$
```

into the raw file, it will be compiled as $\frac{3}{5}$. The dollar signs are used for math mode, which will be described below. Virtually any mathematical symbol or notation you can think of has a built-in Latex command, and any exceptions can almost certainly be found in a downloadable package.

Once you are comfortable with the basics, the best way to learn LaTeX is to use it for many different projects and assignments! It is very easy to just search online to learn about more features if you want to typeset something that you don't already know how to. Perhaps the best resource is: `http://en.wikibooks.org/wiki/Latex`. If you want a certain symbol or format, chances are, someone has already done it, and there will either be built-in features or a package which can do it for you. Here, we have an introduction to the necessary basics as well as some features which will be especially useful. Over time, you will learn many more features which will allow you to further customize your documents.

2 Obtaining LaTeX

LaTeX is a very powerful system, and correspondingly, it is a little complicated to set up. However, we have instructions for the three major operating systems which should help you start using LaTeX!

2.1 Windows

To use LaTeX, you need two pieces of software: the TeX compiler, and an editor for LaTeX files.

1. Compiler: Download MiKTeX here: `http://miktex.org/download`. Run the installer. If you plan on using an editor other than TeXworks (the editor packaged with MiKTeX, see below), keep track of the installation directory (the default is `C:\Program Files (x86)\MiKTeX 2.9` or `C:\Program`

`Files\MiKTeX 2.9`). After selecting the installation directory, you will be asked to set your preferences:

Preferred paper: Change it to Letter (for United States)

Install packages on-the-fly: MiKTeX will automatically download missing packages from the package repository as you need them. (This is really really convenient so you don't have to manually find and install packages later.) It is a good idea to select "Yes". You can instead select "ask me first", though this can sometimes cause some bugs. You can always change this setting later.

2. Editor: MiKTeX comes with an editor, TeXworks, so you don't need to install an editor separately. However, if you don't like TeXworks, you can install a different editor. There are a variety of options, eg. see this list. Some common ones are TeXStudio and TeXnicCenter. TeXworks is the easiest to install and configure since it comes with MiKTeX, but other editors may have more features. A particularly useful feature to have is tab-completion of commands.

After installing MiKTeX and your choice of editor, you need to configure your editor to work with MiKTeX to make PDFs. This should be very easy for TeXworks; after installing MiKTeX, you should be able to find TeXworks in your list of programs in the Start menu. If you cannot find it, try looking in your MiKTeX installation directory;
`path-to-directory\miktex\bin\miktex-texworks.exe`

In TeXworks, click the big green circle with a triangle in it to compile your document. Alternatively, use the keyboard shortcut Ctrl-T or go to the menu Typeset and click Typeset. Make sure that pdfLaTeX or pdfLaTeX+MakeIndex+BibTeX (the default) is selected. (This can be done via the dropdown menu next to the compile button, or under the Typeset menu.) After compiling, TeXworks should display the formatted PDF file!

For an example, open TeXworks and type the following text:

```
\documentclass{article}
\begin{document}
This is a test. $y=x^2+1$
\end{document}
```

Save the file and click the green compile button. A PDF should pop up, with text that looks like this:

This is a test. $y = x^2 + 1$

For other editors, you may need to tell it where to find MiKTeX. This is usually done through some sort of "Configure" option or menu in the editor, and you will need to know the MiKTeX installation directory. The instructions will vary for different editors, but should be easy to find with a quick search. Make sure to configure the editor to output to PDF, and not ps or dvi.

2.2 Mac

The easiest way to install LaTeX on Mac is from `http://www.tug.org/mactex/`. There, you can download the MacTeX.pkg which includes everything you will need, including the TeX distribution (the language LaTeX uses to translate the raw file to a PDF), many extra packages, and the program TeXShop, which is both an editor and a compiler for LaTeX. Once you install the MacTeX package, you should be able to open TeXShop and start composing documents! In the TeXShop window, you will see a simple text editor. Make sure the drop-down menu at the top of the window next to "Typeset" reads "LaTeX" and the "Typeset" menu has "Pdftex" checked. Then you can try typing an example document:

```
\documentclass{article}
\begin{document}
This is a test. $y=x^2+1$
\end{document}
```

You'll want to save the .tex file in its own folder, since when TeXShop compiles, it saves many more files in the same directory as the .tex file while making the PDF. Then, click Typeset (or Command+t) to generate a PDF! It will open up so you can review it in TeXShop, and it will also show up in the same folder as your .tex file. The PDF for our sample code should have text that looks like this:

This is a test. $y = x^2 + 1$

2.3 Linux

To use LaTeX, you need two pieces of software: the TeX compiler, and an editor for LaTeX files.

1. Compiler: Many distributions of Linux come with a TeX distribution already. The usual TeX distribution for Linux is TeX Live. You can check if you have LaTeX already installed by typing `which latex` on the terminal, and it should output something like `/usr/bin/latex` if it is installed. (You should also have `pdflatex`, which usually comes with `latex`.) If you do not have a TeX distribution already, you can try to install it from your Linux distribution's package repository or from the TeX Live website.

2. Editor: You can use your favorite text editor (vim, emacs, gedit, etc.), or download a dedicated LaTeX editor. A list of LaTeX editors can be found here. Some common ones are TeXStudio and TeXworks.

To compile a `.tex` file from the terminal, `cd` into the directory containing the file, and run `pdflatex filename.tex`. The will produce the files `filename.pdf` (the file you want) and the auxiliary files `filename.out`, `filename.aux`, and `filename.log` (occasionally a few other files are also produced). Run this command every time you want to update your pdf file. If you are using a dedicated LaTeX editor, there will most likely be a menu button or hotkey to compile the document. You may need to follow the editor's instructions to configure it with your LaTeX distribution (if it doesn't do this automatically), and to make sure you output to pdf and not ps or dvi (generally, this is done by using pdflatex instead of just latex).

To change the default paper size to letter (if for some reason it is not already), use `sudo texconfig paper letter`.

For an example of running from the terminal, create a file with the following text and save it as `test.tex`.

```
\documentclass{article}
\begin{document}
This is a test. $y=x^2+1$
\end{document}
```

Open a terminal and `cd` to the directory with the file `test.tex`, and then run `pdflatex test.tex`. It should indicate that output has been written on `test.pdf`, and you should find the file `test.pdf` in the same directory. The PDF should have text that looks like this:

This is a test. $y = x^2 + 1$

2.3.1 Managing packages (read as needed)

Sometimes, you will need a package that is not included in the default TeX Live installation. Often, the package will be bundled with some other packages and available in your Linux distribution's package repository. For example, `texlive-latex-extra` is a common bundle to install. (On a Debian-like system, it can be installed via `sudo apt-get install texlive-latex-extra`.) If you don't mind downloading lots of packages, there is `texlive-full`, which can be installed the same way. (It is around 1-2 GB).

You can alternatively download a missing package individually from CTAN . Installation instructions are generally found in the `readme` file included with the package, but installing usually takes multiple steps.

3 Preambles and Packages

Before you can start writing more complicated documents, we need to learn about preambles. At the beginning of each document, we start with:

```
\documentclass{article}
```

(or some other document class, which will not be covered here). However, we also need to tell LaTeX what packages to use and other parameters for the document before we begin writing it. This collection of lines before the "begin document" line is called a preamble. It consists of packages (all standard ones are included in the TeX distribution download), custom commands, and parameters for the document.

The idea of a LaTeX package is to give access to more commands. One package which most mathematicians include in everything they do is the "amssymb" package, which defines many mathematical symbols which aren't in LaTeX by default. For example, "not congruent" symbol $\ncong$ is not automatically in LaTeX. I am able to use it here because I included the following line in my preamble:

```
\usepackage{amssymb}
```

Over time you will learn which packages you need for the types of documents you write. Not all packages are just symbols, though! Many packages include tools for formatting, such as making custom headers or designing graphics.

Other parameters for the document which go in the preamble can include formatting rules, such as if you want to change margins or font size (the "article" document class has nice default margins and font size, but if you want to change it, the preamble is where that happens). These changes are very easy to find out how to do online. They are often in the form of packages which don't have any commands. For example, the following is one way to make margins smaller (this document utilizes it):

```
\usepackage{fullpage}
```

You can also make custom page numbering, custom header rules, and many other choices. Most document classes automatically have a nice, standard format, but you might want to change it, and you can easily do so in the preamble. Also in the preamble, you will often include the author, title, and date, all with commands such as:

```
\title{An Introduction to \LaTeX}
```

and then in your document right after the preamble, you would use the following command:

```
\maketitle
```

and the title, author, and date (and other parameters which you may specify) will show up formatted automatically. This document used the `\maketitle` command with blank date and author.

The last major type of command in the preamble is a custom command. One common example is to shorten the script for letters such as $\mathbb{Z}$ or $\mathbb{R}$. The normal command is `\mathbb{Z}`. But with the following command, it will just be `\Z`:

```
\newcommand{\Z}{\mathbb{Z}}
```

It is pretty clear how to use this for other purposes, and each individual will over time decide what commands they would like to customize. If you try to give a name to a command that LaTeX already has assigned, you get a compiler error.

Now that we can start documents, we will learn some basic commands.

4 Basic Commands

Once you are below the "begin document" line, you are composing the document instead of formatting it. If you just start typing words, when compiled they will show up automatically in paragraph form. New paragraphs are formed by leaving one or more blank lines in the raw file. Adding a % sign at any point in a line will comment out that line, meaning the compiler will ignore the rest of the line. Mathematical symbols are almost all exclusive to what is known as math mode. Many symbols cannot compile outside of math mode, and when in math mode, LaTeX formats the text to be a proper equation. Most of your time will be spent outside of math mode, since in most articles, there is much more regular text than math symbols. However, math mode is where the efficiency of LaTeX over alternatives (such as Equation Editor in Microsoft Office) is immediately apparent when first starting out.

4.1 Math Mode Basics

Unless otherwise stated, all LaTeX code in this subsection will be in math mode. There are three main ways to enter math mode. The most common is to use single dollar signs for inline equations: `$x=y$`. Notice the difference between math mode and normal mode: $x = y$ is in math mode, while x=y is not. The next most common way is to use double dollar signs for separated equations:

$$x = y.$$

This is generated by:

```
$$
x=y.
$$
```

You should use inline text for small and simple expressions and equations, but anything more than a few symbols should be on its own. There aren't set rules for when to use inline equations and when not to, and everyone has their own conventions. Finally, one can use the following for a specific equation environment (I will give an example of why this might be more useful than just `$$` later):

```
\begin{equation}
x=y
\end{equation}
```

When in math mode, you are able to efficiently type virtually any symbol you want. Now, we will give some specific commands which are very useful to know.

Subscripts are written with underscore (`x_0` becomes x_0) and superscripts are written with the caret symbol (`x^n` becomes x^n), and both of these can be nested nicely using curly brackets: `x_{i_j}^{k^l}` becomes $x_{i_j}^{k^l}$. Curly brackets are also necessary if you want to have more than one character in a superscript or subscript. Consider the difference between $x^a b$ and x^{ab}. Those are generated by `x^ab` and `x^{ab}`, respectively. In general, when a command takes arguments, it will be in curly brackets.

Fractions are written as: `\frac{a}{b}`. This becomes $\frac{a}{b}$. You can find other versions for fractions formatted differently (e.g. with a slash instead of horizontal line).

Integrals and sums are written similarly (one is "sum" and one is "int"), and the superscript and subscript notation are used here as well: `\sum_{x=0}^n x^2` becomes:

$$\sum_{x=0}^{n} x^2.$$

Greek letters are generally just their names (capitalization matters): `\gamma` becomes γ and `\Gamma` becomes Γ.

Many of the standard elementary functions come with built-in functions so they render nicely. Notice the difference between $sin(x)$ and $\sin x$. These are made with commands `sin(x)` and `\sin{x}` respectively.

There are of course numerous additional symbols and commands that you will use, but the point here is to emphasize how easy it is to write very clean-looking equations using LaTeX. From here on out, you can simply Google terms if you want to know something else. If you just search for the name of the symbol and "latex" there will probably be dozens of pages which will tell you exactly how to implement it and if you need any packages to display it. The same is true for more complicated tasks, of course.

4.2 Sections of Papers

It is usually important to divide your document into sections. LaTeX will automatically number everything for you, even if you move sections around later on. You can of course override the automatic numbering if you need to, but that will not be covered here. The basic way to add sections is with the command:

```
\section{Basic Commands}
```

and LaTeX will number the section and display it as you see here. You can also add subsections:

```
\subsection{Sections of Papers}
```

and subsubsections are done similarly.

5 More Commands

Here we will explore a few specific instances of slightly more complicated types of commands.

5.1 Labeling Objects and Equations

Often in a book or paper we want to refer to equations, figures, or sections. LaTeX has an extremely easy way to do this. Whenever you have a `\begin{...}` environment (usually for equations or images) you may add a label line to that section. Here is what it looks like:

```
\begin{equation}
x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
\label{quadratic}
\end{equation}
```

This will show up as the following:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{1}$$

Then at any point in the paper (before or after the labeled object), you may add the command `(\ref{quadratic})`. That will appear as, e.g.: "we refer the reader to (1) for the solution." If you move the equation elsewhere, LaTeX will automatically renumber all equations in the document to reflect that. The equation environment is used instead of `$$` most often to get labeled equations. This mechanic is the same for bibliographies (more on that later), figures, equations, section names, and other objects.

5.2 Matrices and Piecewise Functions

There are multiple ways to make matrices. Here we will briefly see one simple way. For more customizability, it is easy to find resources online. The matrix

$$\begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix}$$

is given by the following command:

```
\begin{pmatrix}
a & b & c \\
d & e & f \\
\end{pmatrix}
```

The reader should be able to see how to extend this to other matrices. The "p" in pmatrix means parentheses. Without the "p" there would be no surrounding delimiter, which is also a reasonable way to write matrices. There is also an "array" environment which offers much more customizability.

Another useful tool which behaves somewhat similarly to matrices is making piecewise functions. There are again many ways to make them. One way is to use what is known as a "cases" environment:

$$u(x) = \begin{cases} \exp x & \text{if } x \geq 0 \\ 1 & \text{if } x < 0 \end{cases}$$

This is made with the following LaTeX code:

```
\[
u(x) =
\begin{cases}
\exp{x} &
\text{if } x
\geq 0 \\
1 &
\text{if } x
< 0
\end{cases}
\]
```

This is a little complicated, but there is no need to memorize this, as it is perfectly fine to look it up every time you want to use such a setup. It is good to read through this and think about where each line is used. As mentioned, there are numerous options to customize. One good source to check out for more options is: `http://en.wikibooks.org/wiki/LaTeX/Mathematics#Matrices_and_arrays`.

5.3 Theorem and Proof Environments

Because LaTeX is most commonly used for mathematics, it has automatic environments for theorems and proofs. We generally use the amsthm package, and put the following in the preamble to define our environments:

```
\newtheorem{theorem}{Theorem}[section]
\newtheorem{lemma}[theorem]{Lemma}
\newtheorem{proposition}[theorem]{Proposition}
\newtheorem{corollary}[theorem]{Corollary}
```

The first one tells LaTeX that opening the environment "theorem" (see the example) should be labeled with "Theorem" and the automatic numbering should follow the section numbering (hence the "section" in the brackets). The next three define similar environments for lemmas, proposition, and corollary. The word "theorem" in brackets tells LaTeX to use the same numbering as if it were a theorem. Next we give an example of a theorem and proof. Note that the environments are separate because often in math papers the author wishes to state the theorem early and save the proof for later.

Theorem 5.1. *There exist irrational numbers a and b such that a^b is rational.*

Proof. Let $x = \sqrt{2}$, an irrational number. If x^x is rational, we are done. So, suppose x^x is irrational. Then notice the following:

$$(x^x)^x = x^{(x^2)} = x^2 = 2.$$

By assumption, $(x^x)^x$ is of the form a^b for a, b irrational. But we have shown that $(x^x)^x$ is rational. So in either case, such a pair of numbers a and b must exist. □

Notice that LaTeX automatically numbered the theorem as 5.1 since it is the first theorem in section 5. It also automatically includes the "qed" square. All of these features are customizable. Here is the LaTeX code for it:

```
\begin{theorem} \label{irrationalexp}
There exist irrational numbers $a$ and $b$ such that $a^b$ is rational.
\end{theorem}
\begin{proof}
Let $x=\sqrt{2}$, an irrational number. If $x^x$ is rational, we are done.
So, suppose $x^x$ is irrational. Then notice the following:
$$
(x^x)^x = x^{(x^2)} = x^2 = 2.
$$
By assumption, $(x^x)^x$ is of the form $a^b$ for $a,b$ irrational.
But we have shown that $(x^x)^x$ is rational.
So in either case, such a pair of numbers $a$ and $b$ must exist.
\end{proof}
```

It is fairly straightforward to see what is going on in each line. Note that the regular text does not need to be broken into lines; here it is broken up to make the raw code more readable. Identical environments are used for corollaries, lemmas, propositions, and so on.

5.4 Lists

Another important feature you will often use is making lists within documents. The two most common types of lists are "enumerate" and "itemize." The difference is that "enumerate" numbers the objects and itemize just uses bullets. Here is an example of how to use enumerate; itemize is used almost identically:

1. The first item
2. The second item
3. The third item

This list is given by:

```
\begin{enumerate}
\item{The first item}
\item{The second item}
\item{The third item}
\end{enumerate}
```

5.5 Bibliographies

As every academic document needs a bibliography, LaTeX has several automatic ways to construct one. The easiest way, which is sufficient for most normal papers, is to use an environment called "thebibliography." Below is an example of LaTeX's automatic implementation of such an environment. Normally this would occur after the last section.

References

[1] Deer, J. *Triangles, an Introduction.* **A Real Math Journal** 117 (2014), 1-25.

[2] Doe, J. *An Introduction to Numbers.* **Not a Fake Math Journal** 6 (2014), 20-33.

If we wanted to cite the first reference, it would appear as [1]. Now, here is how we compose such a bibliography:

```
\begin{thebibliography}{1}

\bibitem{triangles} Deer, J. \textit{Triangles, an Introduction}.
{\bf A Real Math Journal} 117 (2014), 1-25.

\bibitem{numbers} Doe, J. \textit{An Introduction to Numbers}.
{\bf Not a Fake Math Journal} 6 (2014), 20-33.

\end{thebibliography}
```

As usual, we first establish the environment. The {1} tells LaTeX that we will need at most one digit for citations in the paper, that is, there are at most 9 references (it looks at the number of digits in the braces, not the number itself). That just helps with spacing and margins in the bibliography. Then, like in the enumerate environment, we have items listed, but right after the `\bibitem` command, we name the reference. The names allow us to easily cite the references. If we wanted to cite the first reference, we would type `\cite{triangles}` and it would appear as [1]. After listing the references, we just end the environment. You can also see examples in the code of how to bold or italicize text.

There is a much more complicated system called BibTeX which is more efficient for larger projects and more experienced users. It composes the citations automatically, whereas here we must do it manually. We refer the reader to `http://en.wikibooks.org/wiki/LaTeX/Bibliography_Management` for an introduction.

6 Figures and Tables

Many mathematics papers will have a few figures, and occasionally you'll see a table or two. However, this section is perhaps more useful for other fields - lab reports for example look excellent in LaTeX, and you will obviously need lots of figures and tables there. As you might guess, one can insert and customize figures and tables in several ways. Here we will see the most common way to do so.

6.1 Figures

Figures have a special "figure" environment which gives a label, specifies the file, gives a caption, and specifies the position of the object. It is usually preferable to save a copy of the image in the same directory as the .tex file. You can also specify a pathway for the compiler to check, but it is usually easiest to just put the image in the same folder. We will now look at an example of an image in a document:

This is given by the following:

```
\begin{figure} [h]
\centering
\includegraphics[width = .3\textwidth]{unitdisk.png}
\label{unitdisk}
\caption{The unit disk in $\mathbb{C}$ consists of all complex numbers
inside the circle, not including the boundary.}
\end{figure}
```

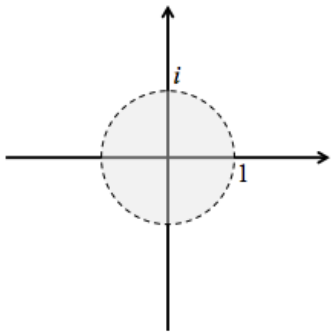

Figure 1: The unit disk in $\mathbb{C}$ consists of all complex numbers inside the circle, not including the boundary.

Let's look at this line by line. The first line introduces the figure environment, and the [h] portion tells LaTeX to put the image *here* ("h" for here). That means the location of the image should correspond to the location of the code for the image relative to the other code. If we put "t" instead of "h", the image would be located at the top of the page, regardless of where the code for the image is. There are other options one can put in the brackets, which are easy to look up online.

The next line just tells LaTeX that we want the image in the center of the page. Next is the line that tells LaTeX that I wanted the figure to be 0.3 times the width of the text, and gives the filename. Following that is the label for the image, and then the caption, and then the end of the figure environment. It looks a little complicated at first, but as you can see, including images is fairly straightforward. If I wanted to refer to the image, I would type `\ref{unitdisk}` and it would appear as 6.1. There are other commands which can make the text of the document wrap around the image, or change the size of the caption, and so on. They are easy to learn about online.

6.2 Tables

Tables aren't as straightforward as images, since there are so many options available. Here, we will explore a very simple example of a small table and examine it as we did for the figure. The reader is referred to `http://en.wikibooks.org/wiki/LaTeX/Tables` to see the plethora of options available.

Consider the following table, with the code for it below:

1	2	3
4	5	6

```
\begin{table} [h]
\begin{tabular} {|c|c|c|}
\hline
1 & 2 & 3 \\ \hline
4 & 5 & 6 \\ \hline
\end{tabular}
\end{table}
```

The first line opens the table environment and has the same position code as in the figure example. The next line begins the data of the table. The code |c|c|c| means we want three columns, all center justified ("c" for center) and the vertical bars mean that we want vertical lines between the columns. The next line is just putting a horizontal line on the top of the table. Then, we have the data of the table, similar to matrices. Then, we end the table. As you can see, there are a lot of places to customize the table. We can have multiple or no vertical lines, change position or justification, and many other options.

7 Concluding Remarks

Hopefully you have seen that LaTeX is an extremely powerful and efficient typesetting system. It is highly recommended that anyone in STEM fields at least be familiar with the basics.

As has been mentioned, LaTeX is used for many different types of documents in addition to just articles. Some students choose to write up all of their homework sets in all of their math and science classes in LaTeX. Most professionals write all of their papers in LaTeX, and many books are even composed using LaTeX. It can also be used for reports, proposals, CVs and other documents we would normally use a word processor for. Another important use is presentations. I encourage anyone giving a technical talk with slides to read about the document class "beamer" to learn how to make beautiful presentations with all of the benefits of LaTeX. We will not be discussing the other document classes here, but it is very easy to find free templates and tutorials online.

As with learning any new software, it can take a little time to get used to the LaTeX environment. However, after a little practice, it becomes very natural, and you might want to use LaTeX for all of your documents!