Introduction to LaTeX

Jennifer Lin

Northwestern University

September 6, 2022

Contents

1 Overview of LaTeX

Throughout your life, you have likely written documents using software like Microsoft Word, Google Docs, and, for the Mac users among us, Pages. Each of these, in their unique, albeit similar, ways allows you to construct a document with proper formatting. When you use these, you likely click a button to do the various things you want, such as making text bold, changing font sizes, and so on. The software does the markup on the back end for you. While simple, these programs are just an overlay for markup languages like HTML, LaTeX and Markdown. In a markup language, you are writing in a plain text editor and you are in control of writing the code that allows things like bold text and font size changes to occur. Plain text simply means that the file contains text only, without any back end coding to enhance the layout and styles of the page. In this document, I will show you how to write documents using LaTeX.

To start, writing in LaTeX requires that you have a few things and what you need is based on your drafting preferences. You can draft on your own computer with a text editor (as you would with something like Microsoft Word) or you can draft online using Overleaf (which works like Google Docs). To write using Overleaf, simply create an account on Overleaf and you are set. You can collaborate on documents with one other person in the free version. If you want to draft on your computer, you will need a text editor. This depends on your preferences, but commonly used editors include R Studio, TeXStudio (my personal favorite), Visual Studio (VS) Code, or Atom. All of these editors are free or have usable free versions. In order to draft on your computer, you also need a TeX Live distribution. If you have a Mac, use MacTeX and if you use Windows or Linux (though this works for Macs too), use MiKTeX. These downloads are rather large and can take up a good amount of storage space on your computer. However, I would recommend having a localized LaTeX distribution if you are committed to using LaTeX for all of your document needs.

2 The LaTeX Frame of Mind

Attention to Detail is a value that we often hear about and probably have been told by a mentor or two over the years to develop and perfect in our work. After all, no employer likes sloppy work. But attention to detail is infinitely more important when writing in LaTeX because it can often be the line between whether your document compiles or fails. It can also be the difference between spending an hour on something versus spending a day on the same task.

So why is attention to detail so important that it can mean the difference between something working and something not working in LaTeX? To answer this question, we first need to understand how LaTeX works. As I mentioned in the introduction, LaTeX is a markup language, just as markdown. Like markdown, you write the code for how the text should appear in the compiled version. Unlike markdown, LaTeX is a bit more involved when it comes to programming. As we will see in the next sections, all functions require something that opens and something that closes it. **Everything has a beginning and an end!** Whether it is a

`\begin{*environment-name*}` or a `{}`, you need to make sure that when something opens, it closes. Most mistakes in LaTeX come from forgetting to `\end{*environment-name*}` any environment or close a brace, parentheses or square bracket. Therefore, pay attention to these details and the majority of your issues should be addressed. For other issues, consult Google, especially Stack Exchange. And do not worry, there will be a lengthier discussion below on what this all means.

3 Setting Up the Document

To this point, I have discussed LaTeX as a markup language, akin to markdown. However, it is also a programming language in the same way that R and Python are. When you write an R script, your file as the `.R` extension. In Python, it's `.py`. LaTeX has many document extensions, as we will see. The main one, for the document that you are editing, is `.tex`. When setting up the document, the process in rooted in programming even though the writing process is markup. In this section, we will see why. Setting up a document requires packages, custom commands (if applicable) and establishing document parameters.

3.1 Commenting

Before getting into the weeds of LaTeX, I should remind you, again that LaTeX is a programming language. Therefore, like R and Python scripts, there is a commenting option. In R or Python, comments are denoted with a number sign (hashtag), `#` but in LaTeX, they are denoted with a `%`. Like comments in R, comments with a `%` will not be compiled in the resulting PDF.

3.2 Document Parameters

First, let's start with the document parameters. When you open a Word document, it is assumed that you are writing an article. If you want to format it differently, say for a book or a letter, you can do that manually. In LaTeX, one of the first things you need to do is state what kind of document this is. This way, the program knows how to compile the document. There are many different types of documents, including `article` (what I am using now), `report`, `book`, `letter` and `beamer` (for slides).

To establish the document parameters, use the `\documentclass{class}` command. Specifically, the class setting for this document is as follows:

```
\documentclass[12pt]{article}
```

Here, the `article` designation tells the program to use article settings to compile the document. The `12pt` in the square brackets tells the program to format using 12 point font (10 point is the default). To use a different document class, simply replace `article` with the class you want.

As you become more advanced in LaTeX, you might see users using different document classes than the ones I mentioned above. And this is possible. Since LaTeX is a programming language, you can program your own document class. All the material is stored in a `.cls` or class file. When you install the TeXLive distribution, the article class file is already on your computer. But, if you desire to make changes to the defaults, you can do so and create your own class, and call it whatever you want, with the `.cls` extension.[1] To use it, just make sure that the `.cls` file is in the same folder as the `.tex` document.[2]

3.3 Packages

Before doing any statistical analyses in R or Python, you need to import packages. In R, the command for this is `library(package)` (for example, `library(dplyr)`) and in Python, its `import package as shorthand` (for example, `import pandas as pd`). The goal of packages in statistical software is to bring in extra tools that can help you in your computing needs. While LaTeX packages are not necessarily for computing, they are helpful in helping you format your documents and they make the writing markup process easier as well. In LaTeX, the command to load a package is `\usepackage{package}` (for example, to use the `graphicx` package for images, `\usepackage{graphicx}`). Table 1 lists commonly used packages. It is non-exhaustive! Just as you would not do `library(package1, package 2)` in R, you should use one `\usepackage{package}` for each package in LaTeX.

Table 1: Frequently used packages in LaTeX

Package	**What it Does**
`fullpage`	Full page typesetting – gives 1 inch margins
`setspace`	Allows for double spacing
`latexsym`	Nicely formatted symbols
`fontawesome`	tFont Awesome on paper (References here)
`rotating`	Allows for figure rotation
`longtable`	Tables that span multiple pages
`natbib`	Bibliographies
`authblk`	Author and Affiliation in title
`lscape`	Turn page landscape
`paflscape`	Make landscape pages show up as landscape in PDF compile
`abstract`	Abstract for paper formatting options
`titlesec`	Title and section heading styles
`hyperref`	Hyperlink text
`graphicx`	Graphics capacity

[1] When it comes to LaTeX, there are many document extensions, which we will review thoughout this document.

[2] Of course, if you do not feel the motivation to go this extra mile, setting up a template that you use for all of your writing is good enough and will serve you well in the long run.

`tikz`	Draw your own graphics
`tabulary`	Tables with wrap text feature
`booktabs`	Fancy tables
`fancyhdr`	Fancy page headings
`caption`	Captioning tables and figures
`float`	Force locate figures and tables
`amsmath`	Math functions
`amssymb`	Math symbols
`multicol`	Two column text
`listings`	Verbatum code formatting for R and Python in LaTeX

3.4 Commands

As emphasized many times in this section, LaTeX is a programming language and the programming is most evident in the custom commands that you can write. Writing custom commands is like writing your own function in R or in Python. In R, you might use the following format:

```
function_name <- function(arguments){
        command
}
```

In Python, it might be

```
def function_name(arguments):
        command
```

Just as you can write functions in R or Python to substitute writing things the long way or to have it make a calculation for you, LaTeX functions operate similarly. The format for commands in LaTeX is `\newcommand{cmd}{def}` where `cmd` is the name of the new command or function and `def` is what it is supposed to do.

One of the most classic ways to apply this is to generate shorthands for commonly used markup commands. Instead of writing `\emph{text}` over and over again for italics, you can do `\newcommand\e{\emph}`. Now, each time you need italics, you can just use `\e{}`.

3.5 Document Metadata

Now that you have set up all the packages that you want and commands that you need, you are ready to write a document, almost. If you want a title, author, date, institution or other metadata, you will need to specify it using the appropriate commands. These come from the `authblk` package and is quite straightforward.

- `\title{text}` – Sets the title
- `\author{names}` – Sets the author

- `\affil{affiliation}` – Sets author affiliation
- `\date{text}` – Sets the date – use `\today` to indicate the day that you compile the document

In practice, here is the document metadata that I used to draft this document:

```
\title{\textbf{Introduction to \LaTeX}}
\author{Jennifer Lin}
\affil{Northwestern University}
\date{\today}
```

* * *

Now you are *really* ready to write a document. Remember that everything in LaTeX has a beginning and an end so insert `\begin{document}` and `\maketitle` to generate the title and your affiliation details and let's get into it!

4 Formatting In Text

Now, we are moving past the programming and we are going into the markup. In this section, I will provide a non-exhaustive coverage of the possibilities that you can do in LaTeX. Remember, Google and Stack Exchange are excellent resources to learn the more fancy things.

4.1 Environments

Everything has a beginning and an end in LaTeX! And this is most evident in the case of **environments**. To do anything in LaTeX, you need an environment. The `document` is an environment, the ways to itemize and enumerate lists are environments, figures and tables are their own environments and so on and on. The first key to writing is the `\begin{document}` and `\end{document}`. This tells LaTeX to start and end the document. As we will see throughout, all things require environments and all environments have a beginning and an end. Learning what the environments are might take some time, but for now, just remember that if you `\begin` something, write the code to `\end` it right away. Same goes for parentheses, curly and square braces, by the way. Table 2 show all the commonly used environments that we will discuss in greater detail in the next sections.

Table 2: Frequently used environments in LaTeX

Environment	**What it Does**
document	Contains the main body of the document
singlespace	Single Spacing if `\doublespacing` is used in the document preamble

itemize	Bulleted list
enumerate	Numbered list
quote/quotation	Quote block (Same as `>` in markdown)
equation	Math equations
figure	Figures
table	Standard table
longtable	Requires `longtable` package – generates tables that can span multiple pages
verbatim	Block of plain text code that should not run
landscape	Landscape page orientation – requires the `lscape` package

4.2 Translations from Markdown

LaTeX markup follows, generally, the same conventions and frame of mind as markdown, with slightly different commands. This will be most evident when writing math equations. But, before we get there, formatting text from markdown to LaTeX is quire similar. Table 3 shows markdown conventions translated to LaTeX.

Table 3: Translating from Markdown to LaTeX

Markdown	**LaTeX**
`# Heading 1`	`\section{title}`
`## Heading 2`	`\subsection{title}`
`### Heading 3`	`\subsubsection{title}`
`**text**`	`\textbf{text}`
`*text*`	`\textit{text}`
`` `code` ``	`\verb`
`[](url)`	`\href{URL}{text}`
`[^ Footnote]`	`\footnote{text}`
`"quotes"`	``` ``quote ´´ ```

Below, I will demonstrate a paragraph formatted nicely, in markdown and in LaTeX.

Nicely:

Some people say that learning LaTeX is a difficult process. I am not going to pretend that it isn't. There *definitely* is a learning curve. You might encounter moments where you are struggling to figure out exactly what went wrong with your code to get your document to compile. I will remind you that "Tis you don't succeed, try, try again". **Patience** is perhaps the most important virtue.

In Markdown:

```
Some people say that learning \LaTeX \ is a
difficult process. I am not going to pretend that it
isn't. There *definitely* is a learning curve.
You might encounter moments where you are
struggling to figure out exactly what went wrong
with your code to get your document to compile. I
will remind you that "Tis you don't succeed, try,
try again". **Patience** is perhaps the most
important virtue.
```

In LaTeX:

```
Some people say that learning \LaTeX \ is a
difficult process. I am not going to pretend that it
isn't. There \textit{definitely} is a learning curve.
You might encounter moments where you are
struggling to figure out exactly what went wrong
with your code to get your document to compile. I
will remind you that ``Tis you don't succeed, try,
try again''. \textbf{Patience} is perhaps the most
important virtue.
```

In addition to the markup differences, there are also different ways that the programs handle lists, ordered and unordered. Below, I demonstrate the differences with itemized and enumerated lists in Markdown and LaTeX.

Markdown:

```
Here is the shopping list
for the week

- Apple
- Banana
- Lettuce
- Bread
```

LaTeX:

```
Here is the shopping list
for the week

\begin{itemize}
\item Apple
\item Banana
\item Lettuce
\item Bread
\end{itemize}
```

Formatted Nicely:

Here is the shopping list for the week

- Apple
- Banana

- Lettuce
- Bread

Markdown:

```
Here is how to do laundry:


1. Put clothes in washer
2. Add detergent
3. Turn on washer
4. Wait until done
```

LaTeX:

```
Here is how to do laundry:

\begin{enumerate}
\item Put clothes in washer
\item Add detergent
\item Turn on washer
\item Wait until done
\end{enumerate}
```

Formatted Nicely:

Here is how to do laundry:

1. Put clothes in washer
2. Add detergent
3. Turn on washer
4. Wait until done

4.3 Tables and Figures

As I mentioned before, tables and figures are their own environments and are a bit more difficult to include than in markdown.

To create a table in markdown, you simply needed some `|` lines and you could do it as such:

```
|Left|Center|Right|
|:---|:-----:|----:|
|1|Red|40|
|2|Orange|25|
```

But in LaTeX, you need the table environment. Here is the same table from above in LaTeX and its corresponding code:

Left	Center	Right
1	Red	40
2	Orange	25

```
\begin{table}
        \centering
        \begin{tabular}{lcr}
                \toprule
                \textbf{Left}&\textbf{Center}&\textbf{Right}\\
                \midrule
                1&Red&40\\
                2&Orange&25\\
                \bottomrule
        \end{tabular}
\end{table}
```

Now to inserting figures. The command to do so is `\includegraphics[keyvals]{imagefile}`. The figure should be in the same foldere as your `.tex` file and preferably be in a PDF or JPEG format. The mechanism of inserting the figure is demonstrated below. Notice that the `width` argument shows how large the figure should be in relation to how wide the text is. This can be adjusted to your needs based on space and formatting preferences.

Figure 1: Cute Dog

```
\begin{figure}
        \centering
        \includegraphics[width = .5\textwidth]{Goguma}
        \caption{Cute Dog}\label{fig:cutedog}
\end{figure}
```

From the above code, notice the `\caption{text}` and `\label{key}` commands. `Caption` inserts a caption for your figure and `label` is a identifier for the figure or table. To cross reference it in text, simply use the `\ref{label}` command and insert the key that you designate for the label and you will get a cross reference. For example, Figure 1 (`\ref{fig:cutedog}`) shows an image for a cute puppy.

4.4 Citations and Bibliographies

One of the most useful tools in LaTeX is automated bibliographies. You can do this in Markdown too, but the `natbib` package provides more options to make better bibliographies for articles written with the program.[3] Before getting into citations in text, we first should look at how to get sources into LaTeX in the first place. Just like everything we have discussed so far, bibliography files, or files with the `.bib` extension, are plain text files. Therefore, to create a bibliography, simply create a new file and replace the `.tex` file with a `.bib` extension. Place that file in the same folder as your `.tex` file and you are good to go.

Now, adding sources might be a bit more challenging. Instead of adding it in the format that you might use for Word, you need to program your sources in a way that LaTeX will understand it. The general format for sources usually goes as follows:

```
@article{cite-key,
        author  = {Peter Adams},
        title   = {The title of the work},
        journal = {The name of the journal},
        year    = 1993,
        number  = 2,
        pages   = {201-213},
        month   = 7,
        note    = {An optional note},
        volume  = 4
}

@book{cite-key2,
        author    = {Peter Babington},
        title     = {The title of the work},
        publisher = {The name of the publisher},
        year      = 1993,
        volume    = 4,
        series    = 10,
        address   = {The address},
        edition   = 3,
```

[3] If you use Google to learn about bibliographies in LaTeX, you may encounter other things like BibTeX, BibLaTeX, Biber, and others, which are all software for programming bibliographies. I personally find `natbib` to be the best and most intuitive, which is why I am focusing on it more in this section.

```
        month      = 7,
        note       = {An optional note},
        isbn       = {3257227892}
}
```

There are many others, but the above are the ones that you will more frequently use.[4] Let's unpack the contents of the citations. As we go into this, bear in mind that you do not always need to fill in all the arguments. The only things you truly need are the components that are required in the citation style of your choice.

- `@article` (or anything) – This is the kind of document you are citing. It can be a book, an article, conference paper, preprint, website, book chapter, and so on.
- `cite-key` – This is the reference to the citation. There *needs* to be a unique cite key for every document that you cite. This is how LaTeX will recognize what you are referring to in the main text.
- `author` – Author's name. If there are multiple authors, use `and` to separate each name. For example, if the following characters from *The Office* wrote a paper together, the corresponding entry for `author` would be {`Michael G. Scott and Dwight K. Schrute and James Halpert and Pamela M. Beesly and Andrew Bernard`}.
- `title` – Title of the paper, capitalization case does not really matter here because it will all be automatically formatted to the standards required by the style of choice.
- `journal` – Journal of Publication
- All other components should be self-explanatory.

Learning how to create a citation in BibTeX is a good skill to have. Fortunately for all of us, it is not something that you need to do by hand for everything. I would highly recommend, for Chrome users, to install the Google Scholar Chrome Extension. This is an extension where you have Google Scholar pinned on the menu bar on Chrome and it can locate any article you are viewing in your browser in Google Scholar. From there, you can generate a citation in BibTeX and paste it into your `.bib` file, preventing you from needing to do it all by hand.[5] If you do not have Chrome, you can still get automatic BibTeX citation from Google Scholar, but it might involve some more work on your end.

To demonstrate how to generate citations in-text using BibTeX, I will use a paper that I published with Kristin Lunz Trujillo and another paper that I published with Dino Christenson and Todd Makse. Bear in mind that all other document types operate in the same way. I select these two for demonstration purposes because (a) they are mine and (b) I can demonstrate the outcomes with 2 and 3+ authors using all the `natbib` citation commands.

[4] See the post on Verbosus for more examples.

[5] The Google Scholar Chrome extension, and Google Scholar in general, is a good resource for generating citations but I would always double check what you are pasting in as Google Scholar can interpret an article mistakenly as a book chapter, as it often does.

First, let's get the BibTeX citations. I am simply pulling these from Google Scholar.

```
@article{christenson2021ask,
        title={Ask Only What Your Country Can Do for You:
                Group Interests, Constituency Characteristics
                and Demands for Representation},
        author={Christenson, Dino P and Lin, Jennifer and
                Makse, Todd},
        journal={American Politics Research},
        volume={49},
        number={1},
        pages={17--29},
        year={2021},
        publisher={SAGE Publications Sage CA: Los Angeles, CA}
}

@article{lin2022urban,
        title={Urban-Rural Differences in Non-Voting
                 Political Behaviors},
        author={Lin, Jennifer and Lunz Trujillo, Kristin},
        journal={Political Research Quarterly},
        pages={10659129221119195},
        year={2022},
        publisher={SAGE Publications Sage CA: Los Angeles, CA}
}
```

Off the bat, notice a few things:

- Google Scholar automatically generates a cite key for each of the sources. It is in the format of `first author last name` + `year` + `first word of title`.
- There are mistakes in the above examples. For example, for the paper with the cite key `lin2022urban`, the page numbers do not resemble anything nearing a format for page numbers. It is an error, perhaps because the paper does not have an assigned page number at the time of citing this paper. To fix it, we can just remove it.

  ```
  @article{lin2022urban,
          title={Urban-Rural Differences in Non-Voting
                  Political Behaviors},
          author={Lin, Jennifer and Lunz Trujillo, Kristin},
          journal={Political Research Quarterly},
          year={2022},
          publisher={SAGE Publications Sage CA: Los Angeles, CA}
  }
  ```

In order to cite anything in text, we need to connect the bibliography file with the main file. To do this, simply use the command `\bibliography{bib␣file}` and import the name of your bibliography file. You might also want to declare a style, and you can do that with `\bibliographystyle{style}`. Both commands should be inserted inside the document environment, likely towards the end of the file, since that is where the bibliography usually goes. Here is how the bibliography is formatted for this paper:

```
\bibliographystyle{chicago}
\bibliography{References}
```

For this paper, I am using the `chicago` citation style and my bibliography file is named "References.bib". Most of the standard citation styles are built into your LaTeX distribution. However, if you need something that is not there, you will need a `.csl`[6] file, which you can get from the internet, specifically, the Zotero Style Repository. Simply download the file and place it in the same folder as your main document are you are good to go.

Now, let's look at commonly used `natbib` in-text citation styles. Table 4 demonstrates each command, its function and the outcome for two and three or more authors.

Table 4: `natbib` Citations in LaTeX

Command	**What it Does**	**2 Authors**	**3+ Authors**
`\cite{bibid}`	Default citation function	Lin and Lunz Trujillo (2022)	Christenson et al. (2021)
`\citet{bibid}`	In-text citation	Lin and Lunz Trujillo (2022)	Christenson et al. (2021)
`\citep{bibid}`	Fully enclose author names and year in parentheses	(Lin and Lunz Trujillo, 2022)	(Christenson et al., 2021)
`\citeauthor{bibid}`	Include author names only	Lin and Lunz Trujillo	Christenson et al.
`\citeauthor*{bibid}`	Writes out all author names	NA	Christenson, Lin, and Makse
`\citeyear{bibid}`	Includes year of publication only	2022	2021
`\citealp{bibid}`	Suppresses parentheses in citations	Lin and Lunz Trujillo, 2022	Christenson et al., 2021

It is important to note that citation commands have options as well. These options might come in handy if you want to add notes or cite a specific page. Suppose that you want to cite a quote from a certain page in an article, you can use the format `\cite[add.␣text]{keylist}` where the added text are the details that you want to cite. For example, if I am pulling

[6] As opposed to a class (`.cls`) file

something from page 18 of Christenson et al. (2021), I can do Christenson et al. (2021, p. 18) (`\citet[p.␣18]{christenson2021ask}`)

Once you have included the citation commands, you can scroll down to where you included the bibliography commands and see citations for the articles that you cited automatically written out in its long form.

4.5 Symbols and Special Characters

As you have seen throughout this document, many of the special characters that you would normally just type on your keyboard are already used for other purposes. For example, the & is for separating out columns on a table. The is for class file and other command designations. The % is for comments and the $ is for math equations. Therefore, to type any of these symbols, you will need a \ (backslash) in front of the symbol. For example, inserting the & is inserted as `\&`. The Comprehensive Symbol Guide is going to be your friend here, but bear in mind that most symbols can be inserted using the backslash preceding it.

4.6 Mathematics

Ah. We have finally arrived. The glorious section. The main point of LaTeX, as some might argue. Mathematics is often said to be the key of LaTeX mainly because it is arguably simpler to use than math typeset in Microsoft Word and the outcome is infinitely more beautiful. However, there is a catch. Typing math in LaTeX is perhaps easier than Word, but it is the part of using this software that requires the most attention to detail. You need to know how to keep your parentheses, square braces and curly brackets in order because one mistake and your document will fail to compile. As for typing math itself, it is very similar, if not the exact same, as typing in markdown. If anything, math in markdown is math in LaTeX. You can use the `$` for inline equations, `$$` for block equations, or the equation environment. Here, I am going to demonstrate the main skills that you need to type math equations.

Formatted Nicely:

There are some equations that have changed the world. Below are some examples of them. But I am not including equations here because they are beautiful. I am including them because they include something that contains great value to our understanding for how to type in LaTeX.

For starters, let's look at the definition of the derivative. Here, we can see that, to write this equation, we need to know how to insert limits and to write fractions. Fractions, in LaTeX follow the format $\frac{numerator}{denominator}$. The underscore (_) is the symbol for subscripts.

$$\frac{df}{dt} = \lim_{h \to 0} \frac{f(t+h) - f(t)}{h} \tag{1}$$

The Newton's Universal Law of Gravitation is another example of where subscripts and fractions are used. But this equation also introduces the exponent, or superscript, denoted with a carrot (^).

$$F = G\frac{m_1 m_2}{d^2} \tag{2}$$

Finally, the equation for the normal distribution applies the concepts that we saw earlier and adds square roots and greek letters.

$$\Phi(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}} \tag{3}$$

In LaTeX:

```
There are some equations that have changed the
world. Below are some examples of them. But I am
not including equations here because they are
beautiful. I am including them because they include
something that contains great value to our
understanding for how to type in \LaTeX.

For starters, let's look at the definition of the
derivative. Here, we can see that, to write this
equation, we need to know how to insert limits and to
write fractions. Fractions, in \LaTeX \ follow the
format $\frac{numerator}{denominator}$. The
underscore (\_) is the symbol for subscripts.

\begin{equation}
\frac{df}{dt} = \lim_{h \rightarrow 0}\frac{f(t + h) -
        f(t)}{h}
\end{equation}

The Newton's Universal Law of Gravitation is another
example of where subscripts and fractions are used.
But this equation also introduces the exponent, or
superscript, denoted with a carrot (\^{}).

\begin{equation}
F = G\frac{m_1m_2}{d^2}
\end{equation}

Finally, the equation for the normal distribution
```

```
    applies the concepts that we saw earlier and adds
    square roots and greek letters.

    \begin{equation}
    \Phi(x) = \frac{1}{\sqrt{2\pi\sigma}}e^{\frac{(x -
                      \mu)^{2}}{2\sigma^{2}}}
    \end{equation}
```

Beyond the pointers in the demonstration above, there are a few more tricks that might be useful in introductory statistics classes and the overall methods sequence for graduate students. To start, equations are often not as short and pretty as those demonstrated above. Often, we see this in linear models. In this case, the `aligned` sub-environment in the `equation` environment is particularly useful.

$$\begin{aligned} Y = \alpha + \beta_{\text{Party ID}} + \beta_{\text{Ideology}} + \beta_{\text{gender}} + \beta_{\text{age}} + \beta_{\text{Place of Residence}} \\ + \beta_{\text{race}} + \beta_{\text{Church Attendance}} + \beta_{\text{education}} + \beta_{\text{income}} + \varepsilon \end{aligned} \tag{4}$$

```
\begin{equation}
        \begin{aligned}
                Y = \ &\alpha + \beta_\text{Party ID} +
                        \beta_\text{Ideology} +
                        \beta_\text{gender} +\beta_\text{age} +
                        \beta_\text{Place of Residence}\\
                &+ \beta_\text{race} + \beta_\text{Church
                        Attendance} + \beta_\text{education} +
                        \beta_\text{income} + \varepsilon\\
        \end{aligned}
\end{equation}
```

From the above code, we notice that the outcome is aligned with the ampersand (`&`) and the text is broken where there is the two backslashes. This is how the aligned command works. In this demonstration, notice the `\text{text}` command. This function formats text in the standard font, rather than italics, when the code is in math mode.

In math, you sometimes might need to write w Probability Density Function and a Cumulative Density Function. I am not going to go into what these are, but the `cases` environment can be especially helpful for this task. Here is the format and it's code for several examples:

$$f(x) = \begin{cases} \binom{5}{x}.8^{x}(.2^{(5-x)}) & \text{for } x = 0, 1, 2, 3, 4, 5 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

```
\begin{equation}
        f(x) =
```

```
        \begin{cases}
                {5 \choose x}.8^x(.2^{(5-x)}) & \text{for} \
                x = 0, 1, 2, 3, 4, 5 \\
                0 & \text{otherwise}  \\
        \end{cases}
\end{equation}
```

$$f(x) = \begin{cases} 0.25 & \text{for } x = 1 \\ 0.25 & \text{for } x = 2 \\ 0.25 & \text{for } x = 3 \\ 0.25 & \text{for } x = 4 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

```
\begin{equation}
        f(x) =
        \begin{cases}
                0.25 & \text{for} \  x = 1 \\
                0.25 & \text{for} \ x = 2 \\
                0.25 & \text{for} \ x = 3 \\
                0.25 & \text{for} \ x = 4 \\
                0 & \text{otherwise}
        \end{cases}
\end{equation}
```

In addition, mathematics often has function notations for sum ($\sum$), product ($\prod$) and more. Table 5 a short list of these functions and other frequently used symbols and how to insert them into LaTeX.

Table 5: Frequently used Math Commands in LaTeX

Symbol	**Code**
$\sum$	`$\sum$`
$\prod$	`$\prod$`
$\lim$	`$\lim$`
$\int$	`$\int$`
$\iint$	`$\iint$`
$\iiint$	`$\iiint$`
$\rightarrow$	`$\rightarrow$`
$\times$	`$\times$`
$\pm$	`$\pm$`
$\approx$	`$\approx$`
$\ldots$	`$\ldots$`

⋮	`$\vdots$`

5 Compiling the Document

Now that you have your document fully set up, you are ready to compile. Compiling is as easy as pressing the "Compile" button for whatever IDE you use to write the document. Once you do, you will see a PDF output but you might also see a bunch of files with random extension come up, including `.aux`, `.log`, and others. Generally, these are not important and you should clear them, or program your LaTeX IDE to automatically clear them on each build. However, if you have a failed build, meaning that there is an error in your code, the `.log` file might be helpful in figuring out what went wrong. This file is a plain text file so you should be able to open and view it in any text editor.

As a general rule, it is always a good idea to proofread your documents and this is especially the case for LaTeX. You might have a document that compiles, but the formatting might be off. It is a good idea to check these before sending it out.

6 Concluding Thoughts

The way I learned LaTeX was through my advisor from undergrad, who gave me a functioning template and told me to start submitting my stats homework in this program. That semester, I spent a lot more time on the formatting of my assignments than I would have if I had used Word, but I think the outcome paid off. He taught me that patience is perhaps one of the greatest virtues when learning LaTeX and in life in general. The best way to become good ay anything is through practice. And he was right.

The material on LaTeX discussed here might seem like a lot, but through practice, LaTeX can become second nature. You will eventually find that formatting somethings in LaTeX is certainly easier than in Word or Google Docs and the outcome is simply more beautiful.

References

Christenson, D. P., J. Lin, and T. Makse (2021). Ask only what your country can do for you: Group interests, constituency characteristics and demands for representation. *American Politics Research 49*(1), 17–29.

Lin, J. and K. Lunz Trujillo (2022). Urban-rural differences in non-voting political behaviors. *Political Research Quarterly*.

A Greek Letters

Table 6: Greek Letters in LaTeX

Capital	Code	Lowercase	Code
α	`$\alpha$`	A	`$A$`
β	`$\beta$`	B	`$B$`
γ	`$\gamma$`	Γ	`$\Gamma$`
δ	`$\delta$`	Δ	`$\Delta$`
ϵ	`$\epsilon$`	E	`$E$`
ζ	`$\zeta$`	Z	`$Z$`
η	`$\eta$`	H	`$H$`
θ	`$\theta$`	Θ	`$\Theta$`
ι	`$\iota$`	I	`$I$`
κ	`$\kappa$`	K	`$K$`
λ	`$\lambda$`	Λ	`$\Lambda$`
μ	`$\mu$`	M	`$M$`
ν	`$\nu$`	N	`$N$`
ξ	`$\xi$`	Ξ	`$\Xi$`
o	`$o$`	O	`$O$`
π	`$\pi$`	Π	`$\Pi$`
ρ	`$\rho$`	P	`$P$`
σ	`$\sigma$`	Σ	`$\Sigma$`
τ	`$\tau$`	T	`$T$`
υ	`$\upsilon$`	Υ	`$\Upsilon$`
ϕ	`$\phi$`	Φ	`$\Phi$`
χ	`$\chi$`	X	`$X$`
ψ	`$\psi$`	Ψ	`$\Psi$`
ω	`$\omega$`	Ω	`$\Omega$`

B LaTeX Draft Template

```
% ---------------------------------
% Describe document characteristics
% ---------------------------------

\documentclass[12pt]{article}

% Classic packages
```

```
\usepackage{fullpage}
 %full page typesetting -- defaults to 1 inch margins
\usepackage{setspace}
 %allows for non-singlespacing
\usepackage{latexsym}
 %extra symbols
\usepackage{rotating}
 %rotation for figures
\usepackage{longtable}
 %tables that fill more than a single page
\usepackage{natbib}
%better bibliographies
\usepackage{authblk}
 %author and affiliation in opening
\usepackage{lscape}
 % For Landscape options

% Paper Abstract

\usepackage{abstract}
\renewcommand{\abstractname}{}
 % abstractname replaced with NULL
\renewcommand{\absnamepos}{empty}
 % removes the block where abstract would
 %   have been placed, originally 'center'

% Shortcut Commands

\newcommand\e{\emph} %Italics
\newcommand\tb{\textbf} %Bold
\newcommand\un{\underline} %Underline
\newcommand\txt{\texttt} %Code
\newcommand\tsc{\textsc} %Small Caps

% Change Section Header looks

\usepackage[rm, sc]{titlesec}
 % titles are non-bold, small, caps
\titleformat*{\subsection}{\itshape}
 % subsection titles are italic
\titleformat*{\paragraph}{\itshape}
```

```
  % paragraph titles are italic

% Links
\usepackage{hyperref}
 %hypertext links in the document
\hypersetup{
        colorlinks=true,
        linkcolor=blue,
        filecolor=magenta,
        urlcolor=cyan,
        citecolor=black
}

% Graphics
\usepackage{graphicx}
 %graphics capabilities
\graphicspath{ {Images/} }
 % Use the Images folder to hold images

% Tables
\usepackage{tabulary}
\usepackage{booktabs}
\usepackage{caption}
\usepackage{float}

% Math
\usepackage{amsmath} % math functions
\usepackage{amssymb} % math symbols


% Multiple Colmns
\usepackage{multicol}

% ---------------------------------
% Document Parameters
% ---------------------------------

% Spacing
\doublespacing

% Name and Title
\title{Title}
```

```
\author{Name}
\affil{Northwestern University}
\date{\today} %For Date to Today

% ---------------------------------
% Writing
% ---------------------------------

\begin{document}

% Title creation
\begin{singlespace}
        \maketitle
\end{singlespace}

\section{Literature Review}

\section{Methods}

\section{Results}

\section{Discussion}

% Bibliography Section

\begin{singlespace}
        \bibliographystyle{chicago}
        \bibliography{References}
\end{singlespace}

\end{document}
```