LaTeX Guide

James

May 20, 2019

1 What is LaTeX ? Why should I care?

Our story really begins with a man named Donald Knuth [1], probably *one*[1] of the most important mathematicians of the 20^{th} century. Originally, in the early 1900's, books were made using the hot metal typesetting set. Knuth, a young mathematician (and prototypical computer scientist), realized that this was an antiquated way of doing things. The method not only was slow and painful, but it also was in no way really "standardized." As a result, many early mathematical books had a very disgusting typesetting, which was not only displeasing to the eye but also just plain hard to read[2]

In his frustration at the current system, he designed a markup language called TeX. The goal was to have a language which allowed people to create beautiful looking books with minimal effort, as well as provide a standard for all math books. Knuth's language took off almost instantly, but it was not without it's faults – most of the language was not built with the future of computers in mind. Soon, an offshoot language called LaTeX was developed by Leslie Lamport, who wanted to create a version of TeXthat was not only "future proof" (albeit, nothing is future proof) but was also more accessible to the average academic.

In some ways, Leslie and Knuth achieved their goal; LaTeX (and by extension TeX) is now a standard among any STEM field. However, it is not without it's faults. It is a language built in the 1980s, and it's clear in many ways. Many features are not standard (and have to be added in manually), many libraries are now grossly overbuilt and break when importing one another, and it is definitely not an easy language to learn. However, it does produce high quality documents with "minimal"[3] effort.

My brother, an academic but on the more liberal arts side of things, once asked me why he should use LaTeX . I was a huge fan of the language at the time, and I was writing almost every document I could with it. However, I was unable to really come up with a good answer to the question outside of, "It looks really good." In trying to teach him some LaTeX , I had a sort of realization that maybe this wasn't the best tool for every job. He struggled greatly with it,

[1] Emphasis on one, there are many contenders and each are important for different reasons.

[2] For my favorite contender of worst typesetting, take a look at Milnor's book on Morse Theory [2]. You can find it online just by googling it.

[3] Once you know some LaTeX .

and towards the end I realized that he could accomplish the exact same thing with Word or any other WYSIWYG[4] writing program as he could with LaTeX .

On the flip side of things, when I was a freshman I tried to write up comprehensive math notes on integrals using Word. The process is probably one of the most painful things, and has scarred me for life from using Word for Math. So, my advice for you is that you should learn LaTeX with the goal in mind of using it for mathematical/statistical/computer science documents, and really for nothing else. While the end result is beautiful and consistent, it's not always the most efficient thing to use (as you'll eventually learn when you try to add figures or position things manually).

My goal with writing this is to hopefully streamline the more complicated parts of writing a document with LaTeXIt took me many years to get to where I am, and it was all due to there not being any good guides on how to write in LaTeXWhile I am not expecting you to master LaTeX after reading this, you should be able to at least write a homework set or a paper in it's entirety after reading this.

2 Overleaf

When I started using LaTeX I had a horrible time trying to get a decent compiler. At the time I didn't really know much about programming, but based on all the guides it seemed like the language or the IDE came with some compiler, and yet I was unable to compile anything. I'll start by saying this is not how things work. To get LaTeX to work on your computer, you need to download three things – the language itself, which can be found here, a compiler, which you will need to find on your own as it depends on your operating system, and an IDE, because writing LaTeX without one is one of the worst punishments I can think of. *Some* IDEs have compilers built into them or come with them[5] but for the most part you will need to install things separately.

The installation process is also not easy for some reason. I recall running into many errors, and I have had to help many friends through the process of debugging the installer, setting up the IDE, trying to locate the compiler, etc. In short, nothing about this part is easy.

However, we can shorthand it by using Overleaf. Overleaf is an online compiler/IDE for LaTeX meaning you can write the document in the browser, compile it, and download the .PDF at the end. Not only that, you can also collaborate with people, connect it to GitHub/Dropbox, and it comes with other features that are supremely useful. I have written most of my homework sets using Overleaf, and I highly recommend it. Due to this, I'm going to skip over the installation process. If you decide to download it to your own machine, you should contact me for help.

[4] What You See Is What You Get.

[5] I think maybe Texstudio does this?

3 Initializing the Document

To get started, we need to first declare the document. If you are using the format.tex file I sent, then you can skip over this process (as I've done all the work for you).

We first start by initializing the document class. This is done via

```
\documentclass[]{}
```

This command is telling the compiler what kind of document you are creating. Your options here are things like **article**, which is for articles or short papers (this document is an example of an article), **book**, which is for things that will have chapters and need a table of contents, and **slides**, which is used for Beamer (more on that later). For this example, we will generate an **article** We put the type of document in the curly braces, as so:

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

In the square brackets, we can indicate some options about our document. The most important thing is setting the size of the font. For most papers, we will use 11pt font. We indicate this via:

```
\documentclass[11pt]{article}
```

We can also tell it some other things, like to have it prepared for a4paper (a special kind of paper), and have it prepped to be printed on two sides. We indicate this via:

```
\documentclass[11pt, a4paper, twoside]{article}
```

Now that we've specified what kind of document, we want to introduce the skeleton of the document. We need a **wrapper** which will contain where the document beings and where the document ends. If you know HTML, this is like a tag which indicates where something begins and ends. We will indicate this via

```
\documentclass[11pt, a4paper, twoside]{article}
\begin{document}

\end{document}
```

We can now compile this document and something will print! Well, not really. It will most likely tell you that the document has compiled, but still run an error since we haven't put anything in the document. Let's do a standard "Hello world!"

```
\documentclass[11pt, a4paper, twoside]{article}
\begin{document}
```

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

Now, if we compile this we should see a blank page with just Hello world! at the top. This is the basic layout you should use for every document you write. You can play around with the options and document class to see what fits your mood best.

We now want to include a **title** and **author** with our document. We do this *before* we run the begin document command – this is in what is called the **header** of our document. Let's say our title is going to be "First Latex Project"

```
\title{First Latex Project}
```

and the author is "James,"

```
\author{James}
```

In the context of the document, we get

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First Latex Project}
\author{James}
\begin{document}
Hello world!
\end{document}
```

Notice if you compile this nothing will change. This is because we need to declare the make title command. Make sure to do this after the begin document command – this is now something we would like to include in our document. The title command is

```
\maketitle
```

and in the context of the document so far, we have

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First Latex Project}
\author{James}
\begin{document}
\maketitle
Hello world!
\end{document}
```

Notice as well that this auto includes the date. If we would like to manually change the date, we can do this with a date command,

```
\date{}
```

similar to author and title. In context of the document, we have

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First Latex Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
Hello world!
\end{document}
```

Now, we would like to include **sections** in our document, to separate different parts. Let's say we would like to make an introduction section. We do this by calling section (this is with the backslash) and in curly braces putting our title; i.e., something like

```
\section{TITLE}
```

To put it in context, here's the document so far:

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First Latex Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section{Introduction}
Hello world!
\end{document}
```

Notice that the section has a number associated to it. Sometimes, we would like to drop this number. We can do so with an asterik (*) following the command before the curly braces. This tells LaTeX to drop the counter associated to the object.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First Latex Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world!
\end{document}
```

If we made another section, say Body, after the Introduction setting, it would pick up the number that we left off of introduction.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First Latex Project}
\author{James}
```

```
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world!
\section{Body}
\end{document}
```

Notice there is now a 1 next to Body.

Now, you'll notice that the LaTeX doesn't look fancy (like the LaTeX I've been writing throughout). This is because it's actually a built in command. To run it, we need to have a backslash behind Latex, as well as capitalize every other letter.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world!
\section{Body}
\end{document}
```

This is now starting to look like a real LaTeX document.

4 Math Mode

As mentioned earlier, the real reason you should care about LaTeX is for the math mode. This allows you to write mathematical equations with relative ease and speed. Let's say we wanted to try writing the fundamental theorem of calculus. This would be a hard thing with Word, as I would have to go through and find the integral in math mode, click on the upper and lower limits, and then write everything out. On top of that, it would not fit with the rest of the document. However, in LaTeX it's a matter of just writing a few words.

We cannot just write it out in the open, though. In order to write it, we need to enter what is called **math mode**. This can be done in two ways – either on a new line or inline. Generally, if it's a short equation, you will use the inline math mode. To do so, we open and close it with dollar signs. So, if I wanted to talk about a limit after hello world, I would add dollar signs and then use the **lim** command.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
```

```
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim f(x).$
\section{Body}
\end{document}
```

Now, let's say I wanted to write the limit as x goes to infinity. I would use an underscore to indicate that this goes underneath the limit, and then in curly braces use the right arrow command. So

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow infinity} f(x).$
\section{Body}
\end{document}
```

However, this doesn't look quite write, as I'm writing out the word infinity instead of the symbol ∞. To write out the symbol, we just use a backslash with infty.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x).$
\section{Body}
\end{document}
```

This is now starting to bring back good memories of Calculus. The backslash will be very important throughout – anytime we want to write a symbol or Greek letter, we will use the backslash to do so. For example, if I wanted to write out alpha, I would put a backslash behind it and get α.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
```

```
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
\end{document}
```

Now, let's go back to writing out the Fundamental Theorem of Calculus. Here, I would want to use a new line in order to write this out. The rule of thumb is if you're using big symbols (think integral or union), then you will want to go to a new line, as it otherwise looks out of place and is too small to read. To go into math mode for new lines, we use a backslash and a square bracket. As a quick aside, to do comments we will use a percentage sign.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\[ \]
\end{document}
```

Now, to do the integral we simply write **int**. To do the upper limit of an integral, we use the upper caret, and to do the lower limit we again use the underscore.

```
\documentclass[11pt, a4paper, twoside]{article}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\[ \int_a^b f(x) dx = F(a)-F(b) \]
\end{document}
```

We now have the Fundamental Theorem of Calculus! There are many more commands for math mode (think fractions, partial derivatives, and more) but to go through all of them would take me a book. For the most part, I usually just Google the symbols name followed by Latex in order to figure out the appropriate command.

5 Libraries/Packages

Like any programming language, there a bunch of different packages that you can use with LaTeX. I will not really discuss any of them, as again there are too many to cover in a short tutorial. However, a very important package is the **amssymb** (short for American Mathematical Society symbols). These are some important symbols and equations used a bunch throughout math. To import it, we use the usepackage command in the header.

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amssymb}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\[ \int_a^b f(x) dx = F(a)-F(b) \]
\end{document}
```

We would also like to import all of the ams packages. These are given as below

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\[ \int_a^b f(x) dx = F(a)-F(b) \]
\end{document}
```

We can now have numbered equations in our document. Let's say I wanted to refer to the Fundamental Theorem of Calculus later on, and in order to do that I wanted to number it. Instead of using the brackets for math mode, I would use the **align** command.

```
\documentclass[11pt, a4paper, twoside]{article}
```

```
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
\end{document}
```

Now it will be easy to refer to it.

We now touch a little on the **amsthm** package. This will be useful for writing out theorems and comments. Sadly, you can't do much out of the package. Trying to use

```
\begin{theorem}
```

First, we need to decide on what style to use. The link here talks a little more about that, but there are three styles we can use – **plain**, **definition**, and **remark**. Within the link are examples of what this does. Let's say we want to make theorem a plain style. In the header, we declare first the theoremstyle:

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\theoremstyle{plain}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
\end{document}
```

Next, we create a link to the theorem itself. This is done with a newtheorem command, i.e.

```
\newtheorem{}{}
```

In the first curly braces, we decide how we want to call it. Let's say we want to just call it using the name theorem. Then we would put

```
\newtheorem{theorem}{}
```

We then need to decide what will appear in the title. By this, I mean when we call this we will get something of the form

```
[Name] 1. [data here]
```

(if this is confusing, just hang on with me). So, let's have hte name Bananas (just so it's clear what's going on). Then we write

```
\newtheorem{theorem}{Bananas}
```

To put it in context,

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\theoremstyle{plain}
\newtheorem{theorem}{Bananas}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
\end{document}
```

Now, in the body of the document, we can write

```
\begin{theorem}
Theorem information
\end{theorem
```

and it should look something like

Bananas 1. Theorem information

For context, the document code will look like

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
```

```
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\theoremstyle{plain}
\newtheorem{theorem}{Bananas}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
\begin{theorem}

\end{theorem}
\end{document}
```

Now, you should take some time to explore the other theorem styles and see which ones you like and under what context. Personally, I use plain for everything. Let's assume you're writing this for some homework assignment. We would want a **problem** and **solution** wrapper. We can do this with the theoremstyle. We would write

```
\theoremstyle{plain}
\newtheorem{problem}{Problem}
\newtheorem{solution}{Solution}
```

so our document would look like

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\theoremstyle{plain}
\newtheorem{problem}{Problem}
\newtheorem{solution}{Solution}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
\end{document}
```

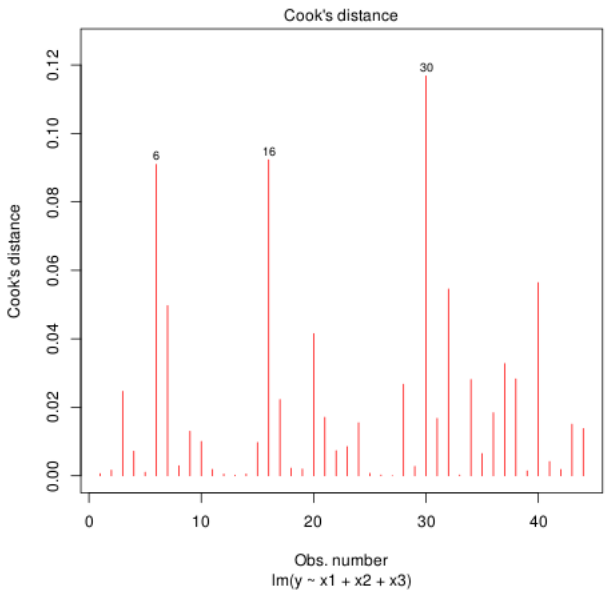
6 Figures

Figures are probably my least favorite part of LaTeX. They are clunky at best, and outright broken at worst. But they are important for things like statistics, when you need to include pictures on different graphs and models.

To get around some of the clunkiness, we'll use the **graphicx** and **float** packages. So, we'll start by including them in our document.

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\usepackage{graphicx}
\usepackage{float}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\theoremstyle{plain}
\newtheorem{problem}{Problem}
\newtheorem{solution}{Solution}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
\end{document}
```

Next, let's use a picture off the internet. I'll use this picture from a recent assignment:



Now, I'll want to put it in a figure. To do so, I can use the figure wrapper:

```
\begin{figure}[H]
\end{figure}
```

The "H" at the end is just to specify that I want the figure to float a certain way, or in other words I want it to sit in the document in a certain way. This is why we included the float package. Generally, LaTeX will choose the worst position by itself, and this lets you somewhat sidestep that.

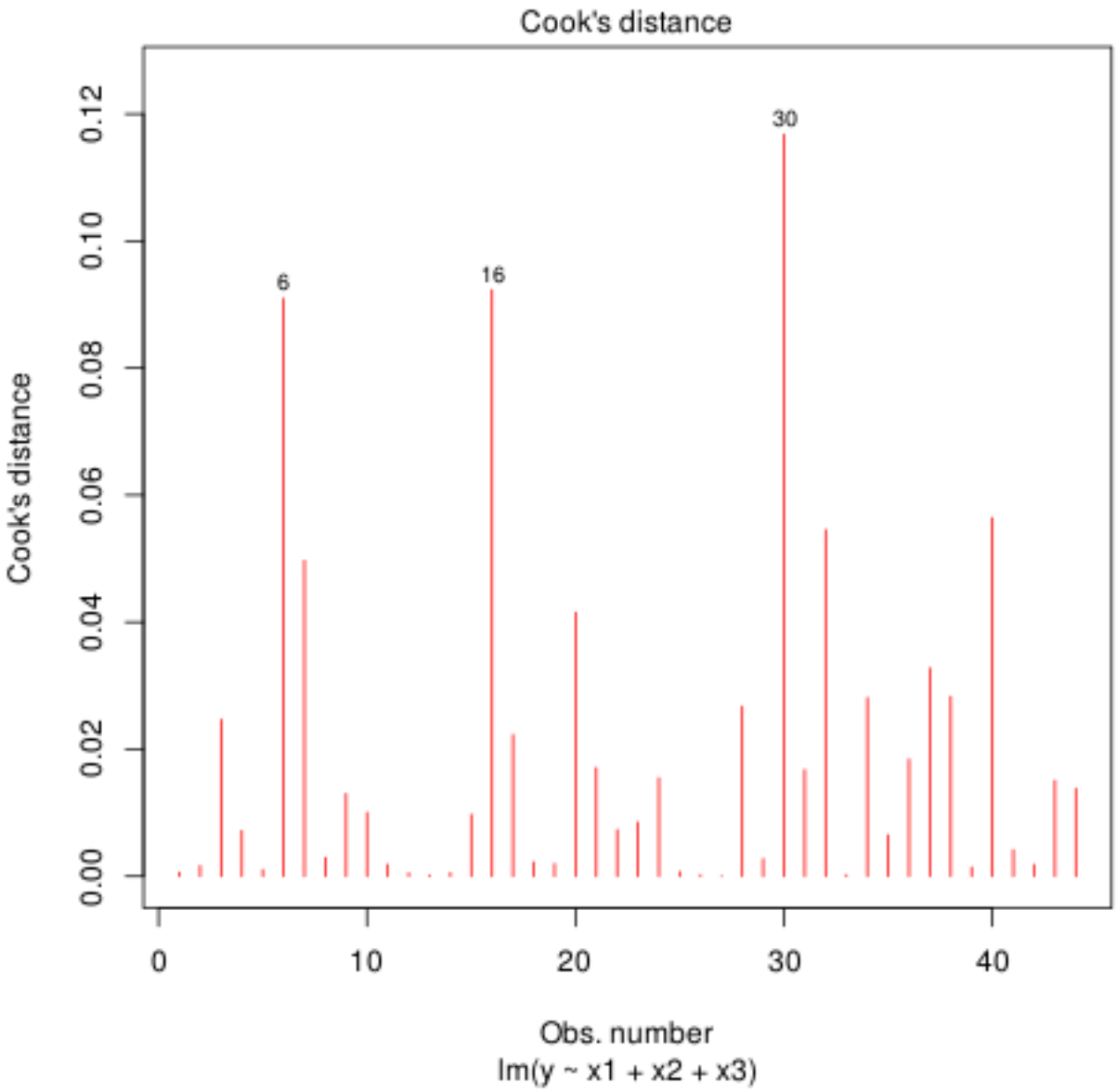
Now, in the figure wrapper I'll need to include some key information. First, we will want to center our figure. I can do this by just calling a generic

```
\centering
```

command, which will center everything in the figure. Next, I'll want to add the picture. I can do this by using the

```
\includegraphics[]{}
```

command. In the square brackets, I'll specify the scale I want the picture to be. I scaled the picture prior, so if you rip it off of this .PDF you'll want to use the full scale. For context, this is the picture unscaled:



Notice it overwhelms everything without making it smaller. So for my code, I would include

```
\includegraphics[scale=0.6]{}
```

You'll want to put the file location in the curly braces. For me, the code and

the picture are in the same folder, so it's sufficient to just put the file name.

```
\includegraphics[scale=0.6]{cooksdistance.png}
```

Finally, we'll want to add a caption for our figure. To do so, we include

```
\caption{}
```

In the curly braces, I'll write what I want the caption of the figure to be. In this case, I'll just set it to be

```
\caption{Cooks Distance of the Model}
```

Next, I'll want to include a label. This allows me to link the figure within the document. The wrapper for this is

```
\label{}
```

In the curly braces, you'll include the name of the figure. This wil be more important later on. For now, let's just label it cooksdistance.

```
\label{cooksdistance}
```

So putting it all together, I have

```
\begin{figure}[H]
    \centering
    \includegraphics[scale=0.6]{cooksdistance.png}
    \caption{Cooks Distance of the Model}
    \label{cooksdistance}
\end{figure}
```

In the context of the document, it'll look like

```
\documentclass[11pt, a4paper, twoside]{article}
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
\usepackage{amsthm}
\usepackage{graphicx}
\usepackage{float}
\title{First \LaTeX \ Project}
\author{James}
\date{Today}
\theoremstyle{plain}
\newtheorem{problem}{Problem}
\newtheorem{solution}{Solution}
\begin{document}
\maketitle
\section*{Introduction}
Hello world! $\lim_{x \rightarrow \infty} f(x) = \alpha.$
\section{Body}
```

```
% Math Mode on new line
\begin{align} \int_a^b f(x) dx = F(a)-F(b) \end{align}
% Figure!
\begin{figure}[H]
    \centering
    \includegraphics[scale=0.6]{cooksdistance.png}
    \caption{Cooks Distance of the Model}
    \label{cooksdistance}
\end{figure}
\end{document}
```

The float package will try to put the picture relative to where you put in the document, but if it's too big it will put it on the next page. To avoid this, try playing with the scaling of the image to find a sweet spot.

To link figures, we use the

```
~\ref{}
```

command. In the curly braces, we use the title given by the **label** command. So, if we wanted to refer to "cooksdistance," we would write

```
~\ref{cooksdistance}
```

The label command is not limited to just figures. You can attach it to pretty much anything that is an object, and then use the ref command to call it later.

7 Tables

It turns out this is actually one of the more logical things in my opinion. To create a table, we call

```
\begin{tabular}{}
\end{tabular}
```

In the second curly braces, we're going to not only determine how many columns we have, but also the type. The sort of style we're going to use is

```
\begin{tabular}{c|c}
\end{tabular}
```

The line inbetween the cs denote a line inbetween the columns. We could drop this if we wanted to, although I don't really recommend doing that. An example of what this looks like is

Test	Test
Test	Test

We could also put bars on the outside, via

```
\begin{tabular}{|c|c|}
\end{tabular}
```

This gives us more of a "boxy" look, as follows:

Test	Test
Test	Test

You can customize this as you'd like. We now want to start filling in the cells. We have two columns in our table so far, so let's fill them in with the numbers 1 and 2;

```
\begin{tabular}{c|c}
    1  & 2
\end{tabular}
```

Notice the ampersand is what differentiates cells here. To get to the next row, we use a double backslash;

```
\begin{tabular}{c|c}
    1 & 2 \\
\end{tabular}
```

Now we can fill in the next row with 3 and 4:

```
\begin{tabular}{c|c}
    1 & 2 \\
    3 & 4
\end{tabular}
```

and we could keep going. While you need to specify the number of columns, the number of rows is never specified, and you can keep going as much as you'd like.

We could also add lines between rows. We do this with the **hline** command, via

```
\begin{tabular}{c|c}
    1 & 2 \\
    \hline
    3 & 4
\end{tabular}
```

This looks something like

1	2
3	4

So we could keep going and make our table look like a box table:

```
\begin{tabular}{|c|c|}
    \hline
    1 & 2 \\
    \hline
    3 & 4
    \hline
\end{tabular}
```

which looks like

1	2
3	4

You may be wondering why I used **c** when declaring the columns. This actually specifies the kind of column I'm declaring; in other words, everything in the column will follow this format. The options here are **c** for centered, **l** for left float, **r** for right float, and **p** for a paragraph column. With the paragraph column, we have some options as well; for more information, look here.

While on the topic of tables, I'll talk a little bit about matrices. Matrices are essentially tables called in math mode. To do so, we run

```
\[ \begin{pmatrix}
1 & 2 \\
3 & 4 \\
\end{pmatrix}\]
```

which looks like

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

The difference is I used **pmatrix** instead of tabular, and I didn't have to specify the number of columns. The p in pmatrix stands for parenthetical; we could also use bmatrix, where the b stands for bracket. This looks like

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

8 Final Remarks

This is just a basic introduction to everything. Hopefully you are able to use this information to write a basic document in LaTeX. For more nuanced things, I've found just googling the problem to be extremely productive; since the language is so old, nearly every problem you'll run into has been solved at some point. Moreover, there are a great number of things you can do with LaTeX that I did not cover here (BibTex, Tikz, and more), and googling things will help you explore exactly what you need out of it. Most of the excess information will start to become specific to what YOU need out of the language.

I hope this is useful in some way. If not, email me at jamesmarshallreber@gmail.com and I'll try my best to help you out.

References

[1] Donald Knuth's Webpage (2018)
https://www-cs-faculty.stanford.edu/ knuth/

[2] Stein E. and Milnor, J.W. and Spivak, M. and Wells, R. and Mather, J.N. and Griffiths, P. (1963) *Morse Theory*