Learning LaTeX

A Soft Introduction to the Language of LaTeX

Philip D. Waggoner*

*This document is based on the *Learning LaTeX Weekly Workshops*, taught by Philip Waggoner (philip.waggoner@gmail.com) and hosted by the Department of Political Science, University of Houston. *Updated: November 6, 2017.*

Contents

1 Introductory Remarks

The popularity of LaTeX is rapidly increasing, both in and out of political science. As such, to be prepared to engage with this increaisngly popular and quite valuable tool, this document is designed to be a soft introduction to the program. Note that there are dozens of books, websites, and blogs dedicated to explaining and teaching LaTeX. This is for two main reasons. First, like `R`, LaTeX is open source, meaning any programmers can write packages and upload them to `CTAN` (The Comprehensive TeX Archival Network). Thus, the program is changing daily. Second, and most importantly, LaTeX is very complex and can quickly get technical. As this document will only scratch the surface of everything the program can do, I highly recommend you take a look at some of these additional resources to help along the way.

This is not to scare you, but rather motivate you to do two things. First, practice and apply, practice and apply, practice and apply. Use LaTeX everyday for everything you do. You will learn best and most quickly by forcing yourself to climb the steep learning curve. Once you do, you will likely be surprised by how quickly you pick it up. And second, Google any and all error messages and questions you have about whether the program can do something. As noted earlier, there are tons of sources to address problems, from the common to the technical. The best and most easily accessible resource is `https://tex.stackexchange.com/`. For those familiar, this is a lot like the stack exchange for `R` Also, this is an excellent resource for a variety of tutorials, issues, and problems: `https://www.sharelatex.com/`.

To start, our goals are:

- Opening the program
- Creating and typesetting a LaTeX document
- Use footnotes
- Use basic punctuation, typeface, and commands

2 Getting Started

It is important to note that LaTeX is not a word processor, but is rather a formatter or typesetter. This means you give it a series of commands, words included, and then you tell the file to print a document for you based on the commands you give it. And as many have said before, it bears repeating: computers are simultaneously brilliant and stupid; though they can do a ton, they are only as valuable as the information they are told to process. LaTeX is no different, where it will only do what you tell it to do, and will not "autocorrect", as millenials have become so accustomed to today. Thus, it is imperative to learn the mechanics of what's going on behind the scenes, to ensure proper formatting, resulting in a nice, pretty document.

As such, there are **two** key things you need to know about the mechanics of how LaTeX works. First, like `R`, most commands are possible through the use of a package. Packages are collections of commands that provide shortcuts to do stuff, on average (though some packages exist for specialized commands that are only possible with those commands). For example, the package "threeparttable" is a good table creation package, which helps with things like putting footnotes on tables within the text. But you can also do this by inserting a simple textbox in a "tabular" environment (more on this later).

This brings us to the second key thing you need to understand about LaTeX which is that the entire program functions through something called an "environment." An environment is simply a space where you can type commands, beginning with the command "begin", and ending with the command, "end." And environments can and almost always are nested, where you work within a "document" environment, to creatae a "list" environment. While we will unpack these concepts here, the key thing to remember is, to do virtually anything in LaTeX you must be operating within an environment.

3 Simple Documents & Typesetting

Before we create a document, we need to learn, first how commands work, and then some basic commands to get us started.

First, a command is how you "call" something, or tell LaTeX what you want it to do. The way we do any command is be preceding the substance of the command with the backslash. Then you follow it with whatever you want to do.

Second, to see commands in practice, let's walk through starting a simple LaTeX document by creating a document environment. Type...

```
\documentclass{article}
\begin{document}
Here is our test document.
\end{document}
```

To actually create the document, you need to typeset or "compile" the document, which automatically generates a PDF, along with several other compilation files. For our purposes, you don't need to bother with these ancillary command logs and additional files, including error logs.[1] Thus, in `TeXStudio`, click (or in TeXShop, you would click the big button aptly called, "Typeset"),

Then, the output looks like...

[1] A quick word on errors: You will encounter many error messages in LaTeX as you do and will in `R`. Though these are not nearly as useful in LaTeX for helping you diagnose the problem, they are useful when they pop up in `TeXStudio`, in that they provide the line where the error is. When you click on some of the detailed information surrounding an error message, it will actually take you to the entry to help locate the problem. Beyond this, though, when you encounter an error message, simply go to the line it says, and figure out your mistake on your own. If you can't, Google the error message and likely something will come up for it.

Here is our test document.

Let's complicate things a little bit by including a heading with some more identifying information, sections, subsections, and subsubsections. These are some valuable and simple commands to produce a clean, organized document or paper. Let's walk through the following code together.

```
\documentclass[11pt]{article}
\title{Let's Expand our Simple \LaTeX\ Document}
\author{Philip D. Waggoner \\
            University of Houston \\
                Department of Political Science
}

\date{\today} % the command for today's date makes sense... Also, use the percent
sign (%) to insert a comment anywhere
\begin{document}
    \maketitle

\section{Here is the First Section (Preamble)}
Usually \LaTeX\ documents begin with a ``preamble." For our purposes, this is useful
to demonstrate sections.

    \subsection{Here is a Subsection}
Now that we have created our first section, let's create a nested subsection.

    \subsection*{Unnumbered sections}
If you don't want sections numbered, simply type ``*" after ``subsection" or
``section", but before the braces. This simply removes the numbering, but keeps the
section and subsection (and subsubsection) formatting the same.

        \subsubsection{Here is a subsubsection}
\LaTeX\ commands for generating sections, sub, and subsubsection are clearly
intuitive. Take advantage of this feature if you want to try and figure out a
command or something on your own.

\end{document}
```

Then, the output looks like...

Let's Expand our Simple LaTeX Document

Philip D. Waggoner
University of Houston
Department of Political Science

September 14, 2017

1 Here is the First Section (Preamble)

Usually LaTeX documents begin with a "preamble." For our purposes, this is useful to demonstrate sections.

1.1 Here is a Subsection

Now that we have created our first section, let's create a nested subsection.

Unnumbered sections

If you dont want sections numbered, simply type "*" after "subsection" or "section", but before the braces. This simply removes the numbering, but keeps the section and subsection (and subsubsection) formatting the same.

1.1.1 Here is a subsubsection

LaTeX commands for generating sections, sub, and subsubsection are clearly intuitive. Take advantage of this feature if you want to try and figure out a command or something on your own.

4 Footnotes

We now know how to create a document in LaTeX, as well as how to make it look a little bit nicer with sections and appropriate organization. Given that most of us will be using LaTeX primarily for paper writing, at least in graduate school, the next extremely useful command is learning how to insert footnotes. Importantly, we could also use endnotes, but for two reasons I will pass on introducing this at this point. First, to use endnotes in LaTeX, you need to install another package. Given the newness of the program at this point, let's keep things as simple as possible to start. Second, footnotes are much more common in political science, and in general, much less frustrating when reading a paper. This is my humble opinion, but I don't enjoy flipping to the end of a paper to reference an endnote, when the information could just as easily be in front of me on the relevant page as a footnote.

You may be shocked to find out that the footnote command is... "footnote." The only different from word processesors, that may take some getting used to, but that is actually much more intuitive, is the footnote is placed directly in the text of the processor, though in the typeset document it looks like a normal footnote that you would expect to see in a published paper. To get a better sense of precisely what I mean, let's first look at including a footnote in a block of text, and then we will look at the output.

Start by including a footnote in our previous document with the sections. We will include the footnote in the beginning. Type...

```
\documentclass[11pt]{article}
\title{Let's Expand our Simple \LaTeX\ Document}
\author{Philip D. Waggoner \\
            University of Houston \\
                Department of Political Science
}

\date{\today} % the command for today's date makes sense... Also, use the percent sign (%) to insert a comment
anywhere
\begin{document}
    \maketitle

\section{Here is the First Section (Preamble)}
Usually \LaTeX\ documents begin with a ``preamble." For our purposes, this is useful to demonstrate sections.
However, if you want to use subsections, you will need to see the subsequent portions of this
document.\footnote{However, I would use a footnote here to bring something to the attention of the reader as a
caveat, but that which doesn't deserve a prominent place in the text.}

    \subsection{Here is a Subsection}
Now that we have created our first section, let's create a nested subsection.

    \subsection*{Unnumbered sections}
If you don't want sections numbered, simply type ``*" after ``subsection" or ``section", but before the braces. This
simply removes the numbering, but keeps the section and subsection (and subsubsection) formatting the same.

        \subsubsection{Here is a subsubsection}
\LaTeX\ commands for generating sections, sub, and subsubsection are clearly intuitive. Take advantage of this
feature if you want to try and figure out a command or something on your own.

\end{document}
```

Then, our output looks like...

Let's Expand our Simple LaTeX Document

Philip D. Waggoner
University of Houston
Department of Political Science

September 14, 2017

1 Here is the First Section (Preamble)

Usually LaTeX documents begin with a "preamble." For our purposes, this is useful to demonstrate sections. However, if you want to use subsections, you will need to see the subsequent portions of this document.[1]

1.1 Here is a Subsection

Now that we have created our first section, let's create a nested subsection.

Unnumbered sections

If you dont want sections numbered, simply type "*" after "subsection" or "section", but before the braces. This simply removes the numbering, but keeps the section and subsection (and subsubsection) formatting the same.

1.1.1 Here is a subsubsection

LaTeX commands for generating sections, sub, and subsubsection are clearly intuitive. Take advantage of this feature if you want to try and figure out a command or something on your own.

[1] However, I would use a footnote here to bring something to the attention of the reader as a caveat, but that which doesn't deserve a prominent place in the text.

5 Some Final Notes on Fonts and Other Commands

Before looking at a few other commands, note that you can code the type face and font size, if you are unsatisfied with the default option when typesetting. To do so, the commands are also quite intuitive.

For type face, here are the most common:

```
\textit{This is italic} \\
\textbf{This is bold face} \\
\textsf{This is sans-serif} \\
\textsl{This is slanted} \\
\texttt{This is typewriter} \\
\textsc{This is Small Caps}
```

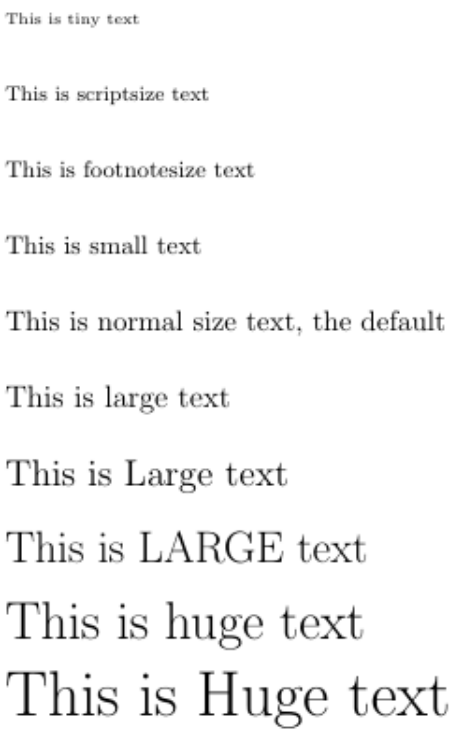
Then our output...

This is italic
This is bold face
This is sans-serif
This is slanted
This is typewriter
This is Small Caps

For font size, note the change can come from either lower or uppercase for the first letter of the command (e.g., "large" versus "Large.") Here are some examples:

```
\tiny{This is tiny text} \\
\scriptsize{This is scriptsize text} \\
\footnotesize{This is footnotesize text} \\
\small{This is small text} \\
\normalsize{This is normal size text, the default} \\
\large{This is large text} \\
\Large{This is Large text} \\
\LARGE{This is LARGE text} \\
\huge{This is huge text} \\
\Huge{This is Huge text}
```

And our output...



To avoid overburdening you with too much information, yet out of a need to make you aware of the many other basic commands you will likely need to know, here is a great reource for a ton of commands: `https://www.ntg.nl/doc/biemesderfer/ltxcrib.pdf`.

6 Supplemental Resources & Readings

A few websites (though there are many out there):

1. `https://tex.stackexchange.com/`
2. `https://www.sharelatex.com/`
3. `https://www.ntg.nl/doc/biemesderfer/ltxcrib.pdf`
4. `https://www.latex-project.org/publications/`
5. `http://www.math.harvard.edu/texman/`
6. `https://tobi.oetiker.ch/lshort/lshort.pdf`

And a few books:

1. Leslie Lamport. LaTeX: A Document Preparation System. AddisonWesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201- 52983-1.
2. Donald E. Knuth. The TeXbook, Volume A of Computers and Typesetting, Addison-Wesley, Reading, Massachusetts, second edition, 1984, ISBN 0-201-13448-9.
3. Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, Chris Rowley. The LaTeX Companion, (2nd Edition). Addison-Wesley, Reading, Massachusetts, 2004, ISBN 0-201-36299-6.
4. Michel Goossens, Sebastian Rahtz and Frank Mittelbach. The LaTeX Graphics Companion. Addison-Wesley, Reading, Massachusetts, 1997, ISBN 0-201-85469-4.

7 Introductory Remarks: Tables & Graphics

Drawing tables in LaTeX like most other things, can be simple or quite difficult. The idea behind drawing tables is in line with doing anything else in LaTeX where we need to start with environments, and then proceed to complicate as much as we like.
It can be frustrating to draw tables, becuase each cell must be in alignment, though there are no boundaries keeping each cell in place, like there is in Excel, for example. Thus, drawing tables requires meticulous care and attention. But the result is well-worth the trouble.
There are many different table environments in LaTeX (e.g., table, tabular, tabu, tabularx, and so on). For current purposes, and for most research, table and tabular should be sufficient. The difference between these two environments in that *tabular* is the default table maker in LaTeX, and *table* allows for a "floating" environment, where we can move tables around more efficiently and easily. We will be using each of these environments in for Tables and Graphics.
Our goals for this section:

- Draw and manipulate tables, positions, & content
- Reference tables within text
- Insert notes within tables
- Insert graphics, plots, & figures

8 Tables

Let's get started with a simple table. To do so, we need to know a couple of useful commands and syntax. First, columns are generated by a position indicator such as "c" (for "center", or "l", "r" and so on), cell delineators are generated by "|", horizontal lines are inserted into the table with the command "hline", and we tell LaTeX where to put our values with the use of the ampersand, "&".
Thus, for a simple table, let's start with the following code,

```
\begin{document}
        \begin{tabular}{ |c|c|c| }
            \hline
            1 & cell A & cell Y \\
            2 & cell B & cell Z \\
            \hline
        \end{tabular}
\end{document}
```

And this code gives us,

1	cell A	cell Y
2	cell B	cell Z

So from the output, we can see that we told LaTeX to generate a table with three columns, with dividing lines between each column. And there should be two rows, given the two rows of information we provided. Let's change some of these parameters to see how the simple table changes. Specifically, let's add another line on top of the table, addional lines on the ends, but remove the column dividers within the table, and let's also add another row of information.
Try this new code,

```
\begin{document}
    \begin{center} % we can use the center environment to center our table
        \begin{tabular}{ ||c c c|| }
            \hline
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{document}
```

Now our table looks like,

1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

8.1 Combining Columns

Now that we have the basic intuition behind creating a table, let's complicate things just a little to give us more flexibility for customizing tables. We will start by combining columns in only part of the table. To do so, we need a new command, "multicolumn". This means, then, that we will also need to add a new line to our original table. Let's start with the following code,

```
\begin{document}
    \begin{center} % we can use the center environment to center our table
        \begin{tabular}{ ||c c c|| }
            \hline
                \multicolumn{3}{|c|}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{document}
```

Then, we get the updated table,

A Combined Column		
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

We can see our new table has a new row added to the top of the table, which is comprised of three columns. This is useful for delineating between different models presented in a single table, for example. We can also combine several columns within the table, rather than the full length of the table. So for our simple table,

```
\begin{document}
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{document}
```

And our output looks like,

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

Notice the combined two columns over the right two columns in the table. This is possible through the addition of the "{2}{c||}", which tells multicolumn to combine only 2 columns, even though there are three, and then to place the double lines to the right of it, to be consistent with our table format. Also notice the blank space in front of the & in the multicolumn line. This tells the tabular environment to not place anything in that space, but to still include it as a column in the table.

8.2 Position Tables Using the "Table" Environment

To this point, we have only been using the tabular environment to draw a table in LaTeX. But let's say we want more flexibility over precisely where our table is placed in the body of the text. The default position using only the tabular environment, is exactly where you insert the code. Thus, the table environment lets us specify the table's location.
To get started, we simply nest the tabular table within the table environment, and then specify the location using brackets after "table", and then one of the following positional identifiers: h ("here"), t ("top"), b ("bottom"), p (specific "page"), ! ("override internal position from LaTeX), H ("exactly here", which also equals, "h!"). To update our simple table code, type,

```
\begin{document}
 \begin{table}[h!]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{table}
\end{document}
```

And here's the output,

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

But, to see the precise impact of these positional identifiers, let's see all positions at once. Try several of them back to back,

```
\begin{document}

 \begin{table}[t]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{table}
%
\begin{table}[h!]
        \begin{center}
            \begin{tabular}{ ||c c c|| }
                \hline
                & \multicolumn{2}{c||}{A Combined Column} \\
                \hline
                1 & cell A & cell X \\
                2 & cell B & cell Y \\
                3 & cell C & cell Z \\
                \hline
            \end{tabular}
        \end{center}
\end{table}
%
\begin{table}[b]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
            & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{table}

\end{document}
```

And here's the output,

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

1

8.3 Captions, Labels, & In-Text Referencing

One of the best features of LaTeX is the ability to keep track of all tables, figures, and equations, regardless of how the text may change. For example, let's say you have a document with four tables in it. But you update the paper and decide you need to add a table of descriptive statistics. So instead of having to go back and manually change all table numbers, and also change any place in the text you reference the old table numbers, LaTeX allows you to imply reference the table with an in-text command based on the caption you gave the table. Thus, adding this fifth table is no problem at all. Nothing in your text changes, and LaTeX does the work of updating all references to table as well as table numbers throughout your text.
To get a better sense of this, we need to use the "caption" and "label" commands. The caption command gives the table a name, which is attached to the table wherever it goes in the text. The label command provides the reference we can cite within the text to keep us on track with our table numbering. To actually cite within the text, we type "ref" after the backslash to call the reference command.
So let's update some code for two tables in order to see precisely how these commands work.

Type,

```
\begin{document}

\noindent First, I would like to talk about my first simple table seen below in Table \ref{table:simple.tab1} with
the caption on top. But then, we should discuss oursecond simple table below in Table \ref{table:simple.tab2}, where
the caption is placed below.


 \begin{table}[h!]
    \begin{center}
     \caption{This is the Title of Our First Simple Table}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}

     \label{table:simple.tab1} % note the difference in our labels
    \end{center}
\end{table}
%
%
\begin{table}[h!]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
            & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
        \caption{This is the Title of Our Second Simple Table}
        \label{table:simple.tab2}
    \end{center}
\end{table}


\end{document}
```

Then, we can see the ouput as,

First, I would like to talk about my first simple table seen below in Table 1 with the caption on top. But then, we should discuss oursecond simple table below in Table 2, where the caption is placed below.

Table 1: This is the Title of Our First Simple Table

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

Table 2: This is the Title of Our Second Simple Table

8.4 Table Notes

Finally, you almost always see some kind of note at the bottom of a table (e.g., denoting levels of statistical significance, references to standard errors, and so on). There are many ways to do this (e.g., threeparttable - you should look this up and use it on your own), but the simplest way for our purposes, is to add subsequent lines to the bottom of our table and manually format to resemble a proper table footnote using our multicolumn command from earlier, where the syntax following the multicolumn command is, "{columns}{position}{text}."
To do this, let's update our code to make our simple table look a little nicer with our new footnote,

```
\begin{document}

 \begin{table}[h!]
   \begin{center}
       \caption{This is the Title of Our First Simple Table}
       \label{table:simple.tab1}
       \begin{tabular}{ c c c }
           \hline
           Obs. & \multicolumn{2}{c}{A Combined Column} \\
           \hline
           \hline
           1 & cell A & cell X \\
           2 & cell B & cell Y \\
           3 & cell C & cell Z \\
           \hline
           \textit{Note:}  & \multicolumn{1}{r}{Here is our table note.}
       \end{tabular}
   \end{center}
\end{table}

\end{document}
```

Now, we can see our updated table is starting to look a bit more like something we would see in a journal (though we still have a ways to go),[2]

Table 1: This is the Title of Our First Simple Table

Obs.	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z
Note:	Here is our table note.	

[2]In R, we can generate base tables of results very easily in many ways. The two main ways to get you started are using either "stargazer" or "xtable" packages. In stargazer, you can simultaneously print and save results, while also generating LaTeX code to simply copy and paste into LaTeX. In xtable, you can print the LaTeX code as well.

9 Graphics, Plots, & Pictures

To conclude, we will take a look at inserting and manipulating figures/plots/graphics/images (hereafter, *images*) into a LaTeX document. First, *we must save all images in the same location as our main TeX document.* So for example, if we have a paper saved in the folder called, "Class 1 Final Paper," all images we want to include in that paper for Class 1 must also be saved in the the "Class 1 Final Paper" folder. Once we have saved all images in the same location, we need to now need to load a package called "graphicx." This let's us place images in a file, as well as allows us to tell LaTeX where to track down our images. Once the package is included in our preamble (at the top of our TeX document), below it we need to set the "graphicspath." This is simply the file path to the location of your main document, set within braces, which specifies the location of both your main document and your images. The synax would look like,

```
\usepackage{graphicx}

\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 2}
```

Once we have our preamble set, and our graphics path specified, we can now insert images easily into any location in our paper using the "figure" environment.[3] The figure environment, like the "table" environment, allows us to specify precisely where we want to place our images using the same positional identifiers (e.g., *h*, *t*, *b*, etc.).
Let's first go to the internet and find an image of anything you want to include in your document. Once you have this, place it in the same location as your main document, whereever that is. Then, to call and place this image, we simply type,

```
\begin{document}

\begin{figure}[!h]
    \includegraphics{UHSEAL}
\end{figure}

\end{document}
```

[3] You can simply use the command "includegraphics." But this limits our flexibility for manipulating the image as we see fit. Thus, let's focus on using the figure environment instead.

Then, our output is the image,

9.1 A Few Notes on Scaling & Sizing

Now there are a few things to consider here. First, I always make the names of my files as simple and clear as possible to limit the threat of confusing LaTeX, such as removing file types from the title (.png, etc.). Also, for our purposes, let's stick with only .png files. Its cleaner and easier for the figure environment, though we can include other image types. As we are using the figure environment, we can manipulate the alignment of the opbject as with a table, using commands such as "center" (see above on centering our tables). Also, and importantly, notice how our image is a little fuzzy. This can easily be corrected by either changing the dimensions of height and width of the image manually using brackets just after the "includegraphics" command (e.g., "width=12cm, height=11cm"), or for a more careful and simpler fix, we can simply alter the scale of the image. To get a sense of this, let's update our code with a few options of rescaling,

```
\begin{document}

\begin{figure}[!h]
    \includegraphics[scale=.4]{UHSEAL}
\end{figure}
%
%
\begin{figure}[!h]
    \includegraphics[scale=.25]{UHSEAL}
\end{figure}
%
%
\begin{figure}[!h]
    \includegraphics[scale=.15]{UHSEAL}
\end{figure}

\end{document}
```

Now, let's see each of the three iterations of the rescaled University of Houston seal,

10 Introductory Remarks: Equations, Lists, & Numbering

In the social sciences, we often write and use equations in our work. Whether calling a functional form of a model, generating proofs, or creating new measures of our own, we will need to be familiar with the ability to both write and include equations in the body of our text. This implies the need to understand at least basic Greek letters, as well as their appropriate usage in writing papers. As I will not teach all Greek letters and specific equations (I leave that to your statistics professors), I assume you have some level of knowledge of these letters and their proper use in statistics and scientific writing.

An important note: when reading equations, we read them as part of a sentence. Therefore, we need to include normal punctuation as we include equations in our work (e.g., after an equation is included, we would place a comma after it, to segue into a block of text describing each term in the equation). Further, we will walk through how to align numerous equations, as well as where equations are placed (e.g., set apart or "displayed" versus inserting a text within the main text). Pivoting a bit, we will then discuss how to create "bulleted" lists as well as numbered lists. These come in handy in a variety of contexts, while may require different placement and formatting within the document. As such, we will walk through proper formatting of lists in written work (e.g., centering or left-justified).

Our goals for this section (made using the "itemize" environment):

- Write, insert, and reference equations in text
- Generate, manipulate, and format *ordered* lists
- Generate, manipulate, and format *numbered* lists

11 Equations

11.1 Basics of Equations

To begin, we have two main options or modes to work with when including equations in our documents. We can either keep them in line with the text, or we can set them apart (i.e., "display"). To include equations within the text, we need specific delimiters (e.g., ( ), $ $, and so on). So for example, type:[4]

```
Here is our first paragraph that includes an equation. To see an equation placed
within the text, the median voter theorem from Anthony Downs (1957) provides a
useful starting place. The (rational) ultility of voting is expressed as,
$R=(B*P)-C$, where $R$ is the reward you get from voting, $B$ is the benefit of
having your candidate win over another candidate, $P$ is the probability your
vote matters, and $C$ is the cost of voting.
```

This code gives us a well-formatting, simple equation placed directly in the text,

> Here is our first paragraph that includes an equation. To see an equation placed within the text, the median voter theorem from Anthony Downs (1957) provides a useful starting place. The (rational) ultility of voting is expressed as, $R = (B * P) - C$, where R is the reward you get from voting, B is the benefit of having your candidate win over another candidate, P is the probability your vote matters, and C is the cost of voting.

So we can see that the equation is include in the same font and size right within our text as we told it to do. Notice that the italicizing of each term is done automatically by placing each term within the "$" delimiter. This especially comes in handy when you have more complicated terms as we will see below.
Now, let's see the same thing, but with a different delimter. Let's use parentheses:

```
Here is our first paragraph that includes an equation. To see an equation placed
within the text, the median voter theorem from Anthony Downs (1957) provides a
useful starting place. The (rational) ultility of voting is expressed as,
\(R=(B*P)-C\), where $R$ is the reward you get from voting, $B$ is the benefit of
having your candidate win over another candidate, $P$ is the probability your
vote matters, and $C$ is the cost of voting.
```

This code gives us,

> Here is our first paragraph that includes an equation. To see an equation placed within the text, the median voter theorem from Anthony Downs (1957) provides a useful starting place. The (rational) ultility of voting is expressed as, $R = (B * P) - C$, where R is the reward you get from voting, B is the benefit of having your candidate win over another candidate, P is the probability your vote matters, and C is the cost of voting.

[4] Downs, Anthony. 1957. "An Economic Theory of Political Action in a Democracy." *Journal of Political Economy*, 65(2): 135–150.

As we would expect, the output is identical.[5]
Now if we wanted to display the equation by setting it apart from the text, we would need a different "mode", which is simply using brackets instead of the parentheses used above. So, to update our previous example, type:

```
Let's see the (rational) ultility model of voting displayed rather than placed in
text. As before, \[R=(B*P)-C,\] where $R$ is the reward you get from voting, $B$
is the benefit of having your candidate win over another candidate, $P$ is the
probability your vote matters, and $C$ is the cost of voting.
```

Now, this code gives us,

Let's see the (rational) ultility model of voting displayed rather than placed in text. As before,

$$R = (B * P) - C,$$

where R is the reward you get from voting, B is the benefit of having your candidate win over another candidate, P is the probability your vote matters, and C is the cost of voting.

See that the equation is now separated from the text and displayed more prominently as we told LaTeX to do by calling the brackets as our delimiters instead of parentheses. Also, note that to read the equation with proper syntax like a normal sentence, we need to place the comma within the bracket delimiter in order to keep the comma associated with the equation (displayed), rather than separated from it.

11.2 Greek Letters & Commonly Used Equations

With the basic intuition behind equations, let's first take a look at a few commonly used Greek letters, and see how these work while building out equations.
First, as with many things in LaTeX, Greek letters are quite intuitive. Simply type the name of the letter following the command backslash. For a few examples, type a few commonly used Greek letters:

```
$\alpha$, $\beta$, $\gamma$, $\sigma$, $\delta$, $\epsilon$
```

[5] Note, that you can display equations also using a double "$" or also the "math" environment, typed directly into the text. Try these out to see whether (if) things change.

As we would expect, this code gives us,

$$\alpha, \beta, \gamma, \sigma, \delta, \epsilon$$

Feel free to experiment (or look up) as many different letters as you wish. You can find many resources online.
Let's look at a few commonly used expressions and equations. A few are fractions, large operators (summation, products), exponents, and subscript. To write a fraction, we simply use the command "frac". The numerator goes in the first set of braces and the denominator in the second set. To use large operators, depending on our needs, we write the command for whatever the operator does. So for summation (captial Sigma), the command is "sum," and for products we call command "prod" (or captial "Pi"). For exponents, we use the carrot, the same symbol as in R. And naturally for subscript, we use the underscore, "_".[6]
To see each of these in commonly used formulas in statistics, type:

```
\(SE_{\bar{x}}=\frac{\sigma}{\sqrt{N}}\) \\ % Standard error of the mean

\(S^2=\frac{\sum(X-\bar{X})^2}{N-1}\) % Variance
```

Now we get the well-formatted equations,

$$SE_{\bar{x}} = \frac{\sigma}{\sqrt{N}}$$

$$S^2 = \frac{\sum(X-\bar{X})^2}{N-1}$$

11.3 Numbering Equations

To number equations, rather than placing an equation directly in the text (either inine or displayed), we would need to use the "equation" environment. In LaTeX, the equation environment automatically numbers (and keeps up with) equations, as long as they are all written in the equation environment.
Also, as with tables and figures, we can give equations labels, allowing us to keep track and organized throughout, as well as reference specific equations in the main text, without worrying how many we have. We do this with the "ref" commend. To see this and everything else we have done in practice, we will use an example from my research on measuring partisan issue prioritization:

[6] Indeed, there are many symbols used in math. These are just a few major examples. For more on this, see: http://web.ift.uib.no/Teori/KURS/WRK/TeX/symALL.html.

```
The PIP Score for $Legislator_{i}$ is given in Equation \ref{eq:pip} as,

\begin{equation}
\label{eq:pip}
{PIP_{it}}=(\frac{BILL_{it}^{P}}{\sum_{\forall{j\neq{i}}}BILL_{jt}^{P}} -
\frac{BILL_{it}^{NP}}{\sum_{\forall{j\neq{i}}}BILL_{jt}^{NP}}),
\end{equation}

where, $BILL_{it}^{P}$, is the sum of partisan priority bills sponsored by an
individual legislator, $i$, belonging to a specific party in a single Congress,
$t$, divided by the sum of partisan priority bills introduced by all other
legislators, $j$, in that same Congress, $BILL_{jt}^{P}$, when $j\neq{i}$.
Subtracted from this total, the second term is similar, but reflecting the sum of
non-priority bills introduced by the same legislator in the same Congress,
$BILL_{it}^{NP}$, relative to the non-priority bills summed across all other
legislators in that same Congress, $BILL_{jt}^{NP}$, when $j\neq{i}$.
```

And now our output with everything altogether is,
The PIP Score for $Legislator_i$ is given in Equation 1 as,

$$PIP_{it} = (\frac{BILL_{it}^{P}}{\sum_{\forall j\neq i} BILL_{jt}^{P}} - \frac{BILL_{it}^{NP}}{\sum_{\forall j\neq i} BILL_{jt}^{NP}}), \tag{1}$$

where, $BILL_{it}^{P}$, is the sum of partisan priority bills sponsored by an individual legislator, i, belonging to a specific party in a single Congress, t, divided by the sum of partisan priority bills introduced by all other legislators, j, in that same Congress, $BILL_{jt}^{P}$, when $j \neq i$. Subtracted from this total, the second term is similar, but reflecting the sum of non-priority bills introduced by the same legislator in the same Congress, $BILL_{it}^{NP}$, relative to the non-priority bills summed across all other legislators in that same Congress, $BILL_{jt}^{NP}$, when $j \neq i$.

11.4 Aligning Multiple Equations

Finally, if we have multiple equations that are especially complicated, or at least that we want aligned at a specific point within the equation, we need to access another nested environment. As you may expect, it is called "align." Importantly, it works similar to the "tabular" environment in making a table, where "&" symbols are used to align text. So the alignment occurs where the ampersand is placed. Also, and importantly, the align environment simultaneously numbers equations, as well as centers them in the page, similar to the "equation" environment. And you can control the numbering with "*" as you would with sections, subsections, and so on (see Week 1 for more on this).
To see this in practice, consider the following code:

```
\begin{align}
2x^2 - 7y &=  19 \\
3x^2 + 6y &=  24 \\
4x^2 + 5y &=  30
\end{align}
```

This gives us the output,

$$2x^2 - 7y = 19 \tag{1}$$
$$3x^2 + 6y = 24 \tag{2}$$
$$4x^2 + 5y = 30 \tag{3}$$

12 Lists

12.1 Basics of "Itemize"

In LaTeX, creating bulleted lists is a very straightforward tool to fill big needs we often have in research and writing. To create lists, we simply leverage the "itemize" environment, where each item on the list is categorized with the "item" command. To see this in practice, type the following:

```
\noindent Here is my first simple list of a few bulleted items: % I am using
``noindent" to left-justify my first sentence

\begin{itemize}
    \item This is item \#1
    \item This is item \#2
    \item This is item \#3
\end{itemize}
```

This gives us the output,

Here is my first simple list of a few bulleted items:

- This is item #1
- This is item #2
- This is item #3

As we would expect, we have a bulleted list with each of our items. Notice my comment after the main setnece (followed by the "%" sign). LaTeX automatically indents, so if you don't want this, simply type the command "noindent" following the backslash as we would with any other command. Also, note that we have to essentially treat the pound sign ("#") as a command, to avoid errors. This is the same as using the dollar sign ("$"), the ampersand ("&"), and so on. Otherwise, it will treat those as delimiters as we have seen throughout.

12.2 Centering Lists

Now let's say we wanted to center our list under some text. As with drawing tables, we would simply include the command "centering." To see this, let's update our simple code with the following,

```
\noindent I am going to increase the amount of text to stretch the length of the
page to give you a point of reference to see the ``centering" command in action.
Now, here is our updated simple list of a few bulleted items:

\begin{itemize}\centering
    \item This is item \#1
    \item This is item \#2
    \item This is item \#3
\end{itemize}
```

This gives us the output,

I am going to increase the amount of text to stretch the length of the page to give you a point of reference to see the "centering" command in action. Now, here is our updated simple list of a few bulleted items:

- This is item #1
- This is item #2
- This is item #3

12.3 Nesting Lists

A final note on itemized lists (or also for numbered lists addressed below). As they are generated using environments, remember back to the beginning when we noted that environments can be nested. Thus, if we wanted to nest a list within a list, we would simply start a new environment within our base "itemize" environment. Let's update our code,

```
\noindent I am going to increase the amount of text to stretch the length of the
page to give you a point of reference to see the ``centering" command in action.
Now, here is our updated simple list of a few bulleted items:

\begin{itemize}
    \item This is item \#1
    \item This is item \#2
    \item This is item \#3
    \begin{itemize}
        \item Here is item \#3.1
        \item Here is item \#3.2
        \item Here is item \#3.3
\end{itemize}
\end{itemize}
```

This gives us the output,

I am going to increase the amount of text to stretch the length of the page to give you a point of reference to see the "centering" command in action. Now, here is our updated simple list of a few bulleted items:

- This is item #1
- This is item #2
- This is item #3
  - Here is item #3.1
  - Here is item #3.2
  - Here is item #3.3

13 Numbering

13.1 Basics of "Enumerate"

Finally, what if we want to number our list? Well, all we need to do is change the environment from "itemize" to "enumerate." All other syntax remains the same. So, let's see this by updating our nested original table previously used for our itemized list. Type,

```
\noindent Let's see the enumerate environment, which generates numbered lists.
Here is our wonderful list.

\begin{enumerate}
    \item This is numbered item 1
    \item This is numbered item 2
    \item This is numbered item 3
    \begin{enumerate}
        \item Here is numbered item 3.a
        \item Here is numbered item 3.b
        \item Here is numbered item 3.c
\end{enumerate}
\end{enumerate}
```

Now we see our same list, only numbered,

Let's see the enumerate environment, which generates numbered lists. Here is our wonderful list.

1. This is numbered item 1
2. This is numbered item 2
3. This is numbered item 3
   (a) Here is numbered item 3.a
   (b) Here is numbered item 3.b
   (c) Here is numbered item 3.c

13.2 A Few Extras on "Enumerate"

If you wanted to "list" certain items, but didn't want any counter or bullets (from "itemize"), you simply type an open and closed bracket after each item in your environment. Also, if you wanted to change the sub-list from letters (default) to numbers, you need to first load the enumerate package (as opposed to just accessing the environment as we have been doing), and then type in the brackets placed outside the enumerate environment beginning, the number or letter you want to start.
So first, before we see everything together, type the following to load our "short labels" options,

```
\usepackage[shortlabels]{enumerate}
```

Now, with that loaded, we can do both of the additions/changes to our list as addressed above, with the following code. Type,

```
\noindent Let's see the enumerate environment, which generates numbered lists.
Here is our wonderful list, with the changes we made.

\begin{enumerate}
    \item [1]This is numbered item 1
    \item [2]This is numbered item 2
    \item [3] This is numbered item 3
    \begin{enumerate}[1]
        \item Here is numbered item 3.1
        \item Here is numbered item 3.2
        \item Here is numbered item 3.3
\end{enumerate}
\end{enumerate}
```

Now we see our updated numbered list,[7]

Let's see the enumerate environment, which generates numbered lists. Here is our wonderful list, with the changes we made.

This is numbered item 1

This is numbered item 2

This is numbered item 3

1 Here is numbered item 3.1

2 Here is numbered item 3.2

3 Here is numbered item 3.3

[7]Note that you may get a warning message saying it couldn't find or use "shortlabels." But check to see that your output is as you think it should be. If it looks fine, then just disregard this warning from TeX Studio.

14 Introductory Remarks: Presentations & Beamer

For classes and conferences, we all have a need for creating presentations. While the most common way to produce quick presentations these days is Microsoft PowerPoint, I would suggest LaTeX produces a much better, cleaner, and more professional looking presentation. And as with most other things in LaTeX, it sends a good signal to folks and also is simple to use on any computer as the presentation is in a PDF format.
As such, we will be focused on creating clean, high-quality presentations in LaTeX, using the "Beamer" document class. While we will touch on many useful things and provide a good starting place, there is much more you can do and add to Beamer to create some good looking presentations. In a word, this is scratching the surface to all Beamer can do. Here we will be focused on starting out and getting you acquainted with the basics of Beamer (e.g., frames, basic formatting, multiple slides, boxes, etc.).
Our goals for this section:

- Create a basic presentation using Beamer
- Add multiple frames
- Be familiar with the basics of Beamer

15 Basics of Beamer

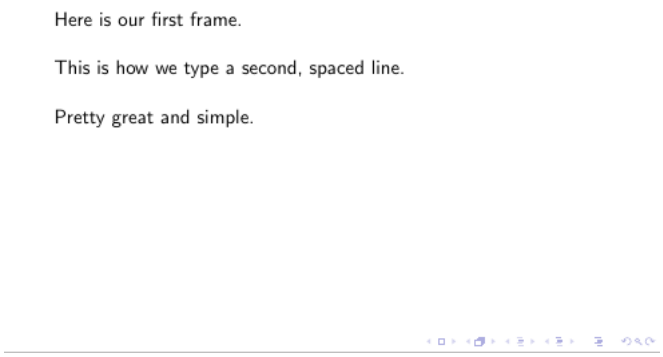
We create presentations in LaTeX using a different document class or a package, *not* by opening a new program as we would in other programs (e.g., Microsoft Word ⇝ Microsoft PowerPoint). Though there are many classes for creating presentations, such as `Fancyslides`, `Powerdot` or `KOMA-script`, by far the most common is `Beamer`. This presentation tool is possible by changing our documentclass to "Beamer," then starting our document enivonrment to add information. So let's begin by typing the following,

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

\begin{document}


\end{document}
```

As there is nothing to compile, let's now turn to add information and material to our presentation. To do so, let's look at slides, or "frames."
To add information into a single slide, we simply start a frame environment, type whatever we want, and then end that environment. This gives us a single slide. Then, we repeat for however many frames we need for our presentation (*Note*: there are many ways to build and design slides, which we will get into in a bit).
So let's update our code with a single, simple frame,

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

\begin{document}

\begin{frame}
Here is our first frame. \\~\\

This is how we type a second, spaced line. \\~\\

Pretty great and simple.
\end{frame}

\end{document}
```

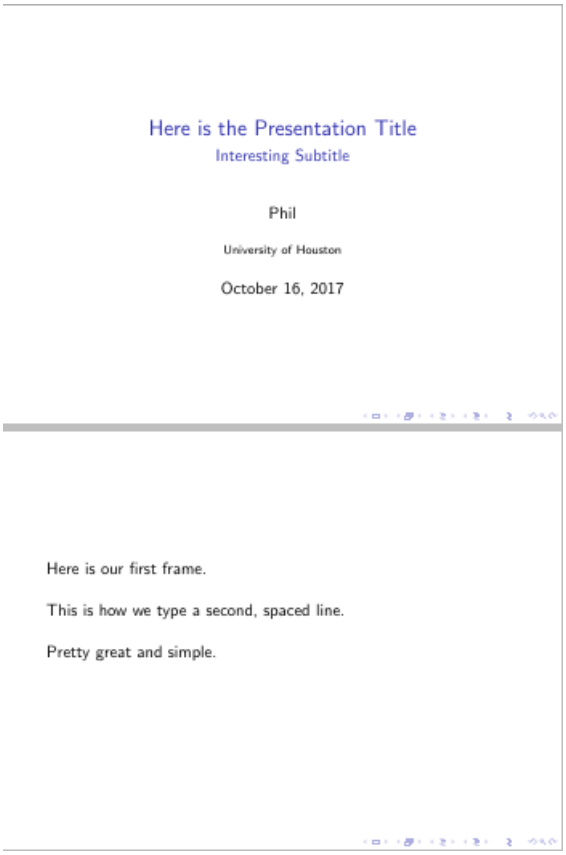
And this gives us a single slide with the information we included,



Now, let's say we wanted to add a title slide. To do so, we need to add the identifying information *prior* to starting the document (i.e., in our preamble). Then, *after* we begin the document environment, we create a new frame that tells LaTeX to make the information in the preamble the title for the whole presentation. To do so, let's type the following,

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

    \author{Phil} % Some basics; much more
    \title{Here is the Presentation Title}
    \subtitle{Interesting Subtitle}
    \institute{University of Houston}
    \date{\today}

\begin{document}

\begin{frame}
\maketitle
\end{frame}


\begin{frame}
Here is our first frame. \\~\\

This is how we type a second, spaced line. \\~\\

Pretty great and simple.
\end{frame}

\end{document}
```

This gives us our two frames: title and the original frame...



15.1 Building a Single Frame

Now that we have a title and basic frame, let's create multiple frames based on a *single* frame. We need to think of it this way, because we often add information a bullet at a time when presenting. Though you could build multiple frames manually (.e.g, a new frame environment for each new bullet), this can be a giant waste of time. Therefore, we will need to update twe things in our original presentation: a new frame environment within braces, and then the "pause" command after each line, denoting a new frame after that line. Let's type the following code to get started,

```
{
\begin{frame}
    Here is our second frame \\~\\ \pause

    Then, we have a second line \\~\\  \pause

    Now we have our final line, but all in a \textit{single} frame environment.
\end{frame}
}
```

Though the output is too large to place in this document, the output should be three separate slides, each adding the successive item, as you coded using the "pause" command.

15.2 Adding Frame Titles

To this point, we have only made simple, blank frames with information. Let's add a title to the frame to see the automatic formatting by LaTeX. Simply include the command, "frametitle" along with the specific title to add it to the frame. Start with,

```
\begin{frame}
\frametitle{Now We Have a Title}

Here is our frame with a nice title.

\end{frame}
```

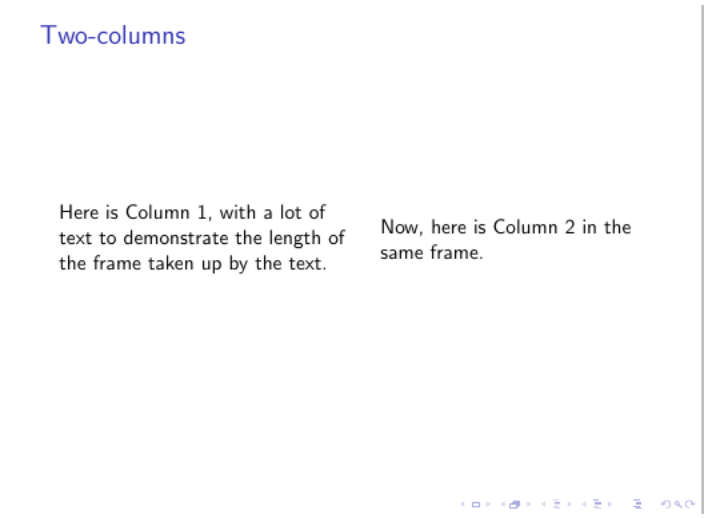
And this gives us,

15.3 Multiple Columns

We are starting to get better presentations. Let's add to this by splitting a frame into multiple columns. To do so, we will work within the "columns" environment within our frame environment. Within this environment, we will specify the amount of columns we want by telling LaTeX the amount to split the frame by (e.g., for two columns, we split it by .5; for three columns we split the frame by .33, etc.).[8] Let's type the following to see how this works starting with two columns, three columns the same length, and then three columns different lengths,

```
\begin{frame}
\frametitle{Two-columns}

\begin{columns}

    \column{0.5\textwidth}
    Here is Column 1, with a lot of text
    to demonstrate the length of the frame
    taken up by the text.

    \column{0.5\textwidth}
    Now, here is Column 2 in the same frame.

\end{columns}
\end{frame}
```

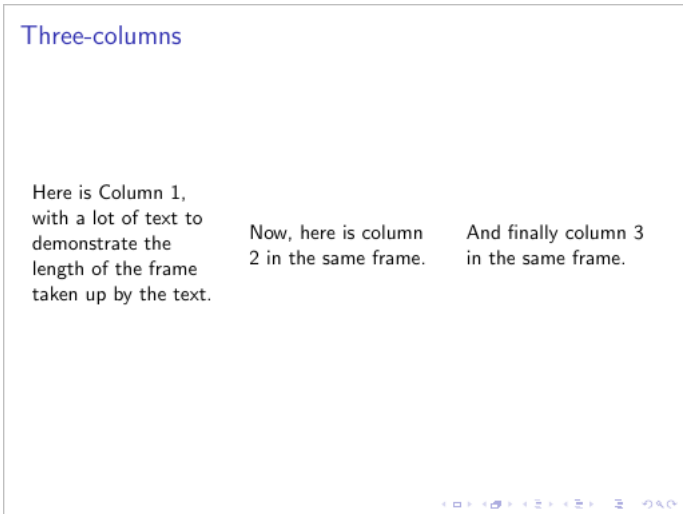
And this gives us a two-column frame as we expect,



Next, for three same-length columns type:
And this gives us a three-column frame,

[8] You could also vary the size of columns to make one small and the other large for example, as long as it adds up to 1. Try out options and different columns on your own.

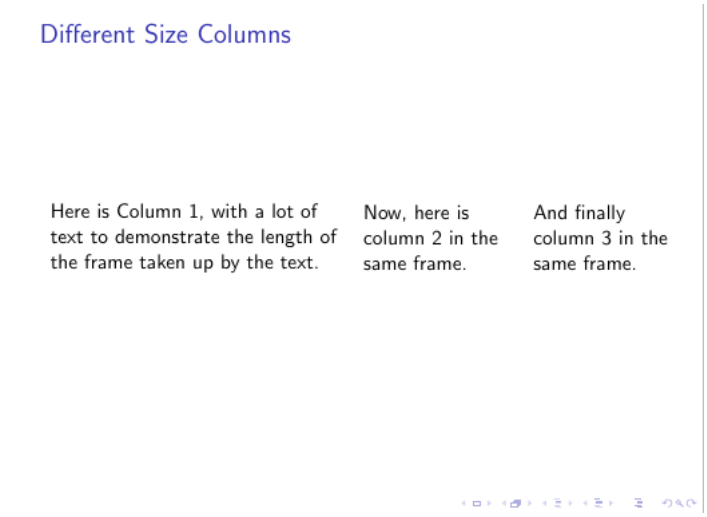
```
\begin{frame}
\frametitle{Three-columns}

\begin{columns}

    \column{0.33\textwidth}
    Here is Column 1, with a lot of text
    to demonstrate the length of the frame
    taken up by the text.

    \column{0.33\textwidth}
    Now, here is column 2 in the same frame.

    \column{0.33\textwidth}
    And finally column 3 in the same frame.

\end{columns}
\end{frame}
```



And finally, for three columns, but of different lengths, type:

```
\begin{frame}
\frametitle{Different Size Columns}

\begin{columns}

    \column{0.5\textwidth}
    Here is Column 1, with a lot of text
    to demonstrate the length of the frame
    taken up by the text.

    \column{0.25\textwidth}
    Now, here is column 2 in the same frame.

    \column{0.25\textwidth}
    And finally column 3 in the same frame.

\end{columns}
\end{frame}
```

And this gives us a three-columns frame with different lengths,



15.4 Adding Figures, Tables, and Equations

Finally, to add Figures, Tables, and Equations in Beamer, you do this the same as you would in a document. This is a great feature of doing all work in LaTeX whereby you can easily move stuff from a paper to a presentation and vice versa, without changing or updating formatting (for the most part). Thus, given that we have already done all of these things, but for articles, we are going to do a quick overview of each showing their applicability in Beamer.
Let's start with Figures. Remember to include the "graphicx" package in your preamble, as well as setting your graphics path to where ever your image is stored to tell LaTeX where to pull it. Thus, remember to save your image (a PNG if possible) in that folder, with a name including no spaces or punctuation). Start with the following,

```
\documentclass[11pt]{beamer}
\usepackage{graphicx}
\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 4}

\begin{document}

\begin{frame}
\frametitle{Adding a Figure}

Here is a frame with a giant UH Seal:

\begin{figure}[!h]
    \includegraphics[scale=.5]{UHSEAL}
    \centering
\end{figure}

\end{frame}

\end{document}
```

Now we have our slide with a giant UH seal included (remember you can resize the image using the "scale" subcommand in brackets):
Next, let's add a table to our frame. To do so, remember we need the tabular environment exactly as before, only this time we include it within the frame environment. Start with the following,
And here is our frame with a simple table,
Finally, let's see the inclusion of an equation in a frame, using the "equation" environment. Start

Adding a Figure

Here is a frame with a giant UH Seal:


```
\begin{frame}

\frametitle{Adding a Table}

Here is a frame with a simple table:

\begin{table}
    \centering
    \begin{tabular}{l c c c}
        Obs. & Party & Votes & Gender \\
            \hline
        1    & 0     & 45    & Male \\
        2    & 0     & 73    & Female \\
        3    & 1     & 75    & Female \\
        4    & 0     & 89    & Male \\
        5    & 1     & 22    & Female
    \end{tabular}
\end{table}

\end{frame}
```

Adding a Table

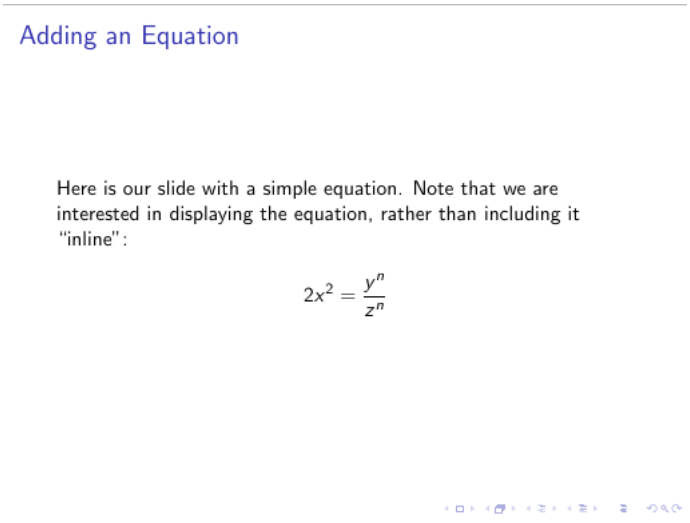
Here is a frame with a simple table:

Obs.	Party	Votes	Gender
1	0	45	Male
2	0	73	Female
3	1	75	Female
4	0	89	Male
5	1	22	Female

with,

```
\begin{frame}

\frametitle{Adding an Equation}

Here is our slide with a simple equation. Note that
we are interested in displaying the equation, rather
than including it ``inline":

\[ 2x^2=\frac{y^n}{z^n}\]

\end{frame}
```

And here is our frame with a displayed equation, rather than "inline,"



16 Design & Themes

To make a nice looking, presentable presentation in Beamer, we have two options: 1. we can either manually change the formatting, or 2. we can use "themes." As this is an *introduction* to TeX let's stick with the simplest approach of using prepackage themes that folks have already done the hard work of making consistent and nice.
As such, we will need some new code to add to our preamble. Specifically, we need two new commands, "mode" and "usetheme." The mode command is accompanied by the word presentation as well as a set of braces, into which the usetheme command will be placed. Essentially, this combination of commands in the preamble tells LaTeX that we are working with a presentation and want to access a pre-packaged theme for our presentation. It will then automatically provide colors and unique frame design.
Importantly, there are *many* theme options to choose from. For a full list, visit, `http://deic.uab.es/ iblanes/beamer_gallery/`. We will only use a few for our examples. Let's launch with my personal favorite for a variety of reasons: "CambridgeUS."
So let's start with the following addition to our preamble:

```
\documentclass[11pt]{beamer}
\usepackage{graphicx}
\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 5}
\mode<presentation> {
    \usetheme{CambridgeUS}
}
```

Make sure you change the graphicspath to be the location where you main document is stored. Now, with our preamble properly specified, let's create a simple title slide using the CambridgeUS theme. Let's start with the following code:

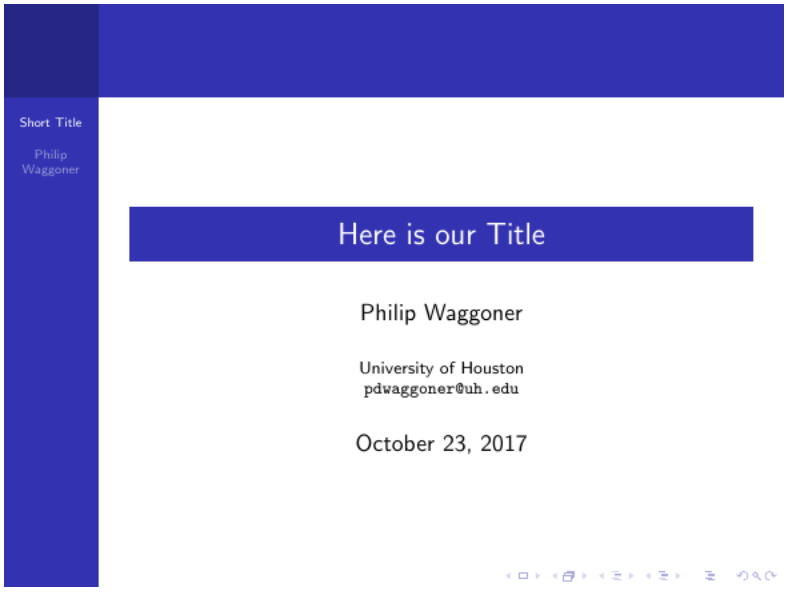
```
\documentclass[11pt]{beamer}
\usepackage{graphicx}
\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 5}
\mode<presentation> {
    \usetheme{CambridgeUS}
}

\title[Short Title]{Here is our Title}

\author{Philip Waggoner}
\institute[UH]{University of Houston \\
    \texttt{pdwaggoner@uh.edu}
}
\date{\today}

\begin{document}

    \begin{frame}
    \titlepage % Print the title page as the first slide
\end{frame}

\end{document}
```

This now gives use the following title slide using the base color scheme in the CambridgeUS theme, as well as the basic frame foramtting (we will address more formatting below):

Let's see another major theme, "Berkeley." So let's update our original code by only changing the theme. *Leave everything else the same to make a direct comparison.* Start with:

```
\documentclass[11pt]{beamer}
\usepackage{graphicx}
\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 5}
\mode<presentation> {
    \usetheme{Berkeley}
}

\title[Short Title]{Here is our Title}

\author{Philip Waggoner}
\institute[UH]{University of Houston \\
    \texttt{pdwaggoner@uh.edu}
}
\date{\today}

\begin{document}

    \begin{frame}
    \titlepage % Print the title page as the first slide
\end{frame}

\end{document}
```

And this gives us the same information but on a completely differently formatted frame,



16.1 Color Themes

Importantly, within a theme, you can also change the color theme if you like the general layout of a theme, but don't like the colors. To do so, we add the new command, "usecolortheme" to our preamble *below* the theme command. So to update our original example using the Berkeley theme, let's use the "crane" color theme:

```
\documentclass[11pt]{beamer}
\usepackage{graphicx}
\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 5}
\mode<presentation> {
    \usetheme{Berkeley}
    \usecolortheme{crane}
}

\title[Short Title]{Here is our Title}

\author{Philip Waggoner}
\institute[UH]{University of Houston \\
    \texttt{pdwaggoner@uh.edu}
}
\date{\today}

\begin{document}

    \begin{frame}
    \titlepage % Print the title page as the first slide
\end{frame}

\end{document}
```

And this gives us a new frame with a new color scheme,

Refer to the website above to see a full list of pre-programmed color themes. Note, the same process should be followed for different font themes. See the above website for a full list of those as well.

17 Table of Contents

Tables of contents can be quite useful in Beamer presentations to provide signposts for the audience, as well as help you stay organized as you create the presentation. It is straightforward to insert a table of contents. To do so, we create a new frame with the command "tableofcontents." This adds a table of contents automatically, based on the sections and subsections in your presentation.
Importantly, there are two main caveates in using a table of contents. First, you must be using the "section" command as you would in a document. And second, you must be using a theme that is compatible with the table of contents. Some themes do not include these, but some do. So the first thing to check if you don't see a table of contents when you told it to be included, is the theme you are using. Try another them as a baseline troubleshooting step.
With that, let's see how this works. Start with the following code updating to use the "Hannover" theme,

```
\documentclass[11pt]{beamer}
\usepackage{graphicx}
\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 5}
\mode<presentation> {
    \usetheme{Hannover}
}

\begin{document}

\begin{frame}
\frametitle{Table of Contents}
    \tableofcontents
\end{frame}

%-------------------------------

\begin{frame}
\frametitle{First Section}
\section{Here is the first section}

Here is frame 1 with a section for our TOC.

\end{frame}

\end{document}
```

And see our output,

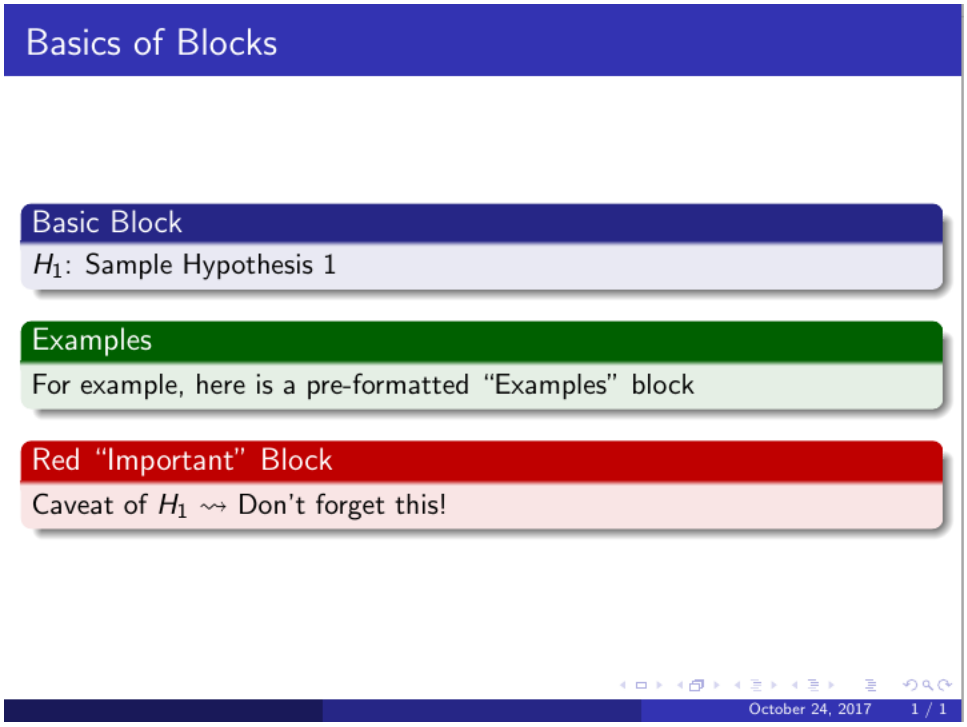
Note a few things: we dropped out title frame for the sake of space, the frame title shows up on the leftside column, and when we get to the specific frame in the presentation, the text automatically gets darker. Feel free to explore and try other themes and colorthemes.

18 Inserting Blocks

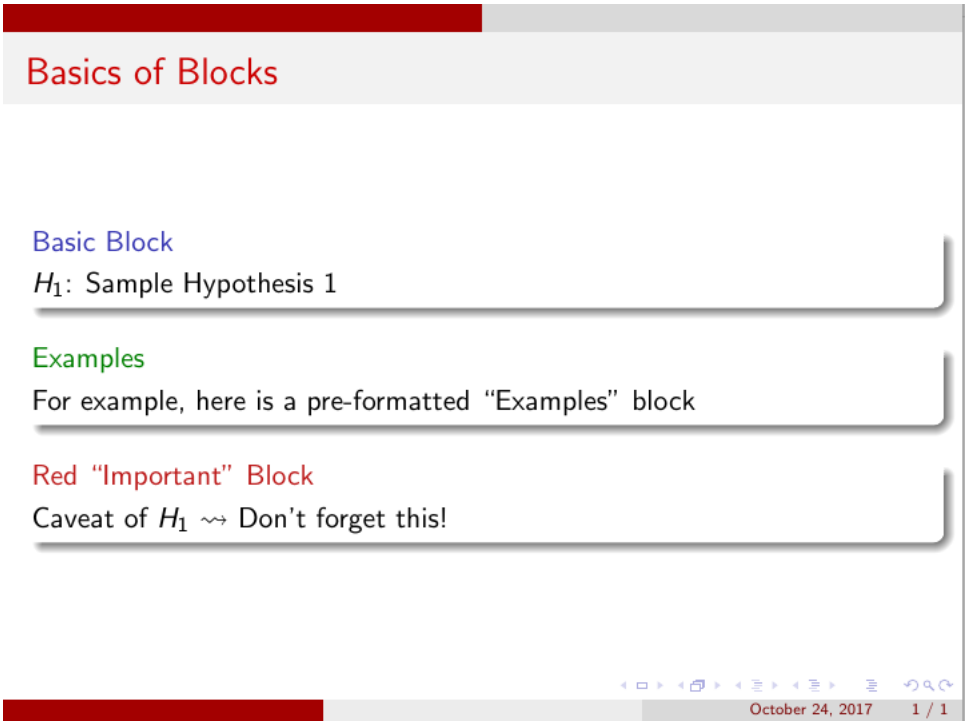
A cool feature of Beamer and LaTeX presentations is adding a block to highlight something in a different way (e.g., hypotheses, equations, etc.). Different colors are associated with different blocks, dependning on the needs. As a quick caveate, similar to the table of contents, some templates allow blocks (or have different colors and formatting associated with different blocks) and others don't. As with table of contents, if your typesetting doesn't include what you thought it should, start by trying another template to see if this fixes the problem. If need be, you can always resort to `Share Latex` or `Stack Exchange`.
There are several ways to add blocks. We can just use the "block" environment (e.g., begin and end). We could also use the "examples" environment to get a different color pre-formatted to stand out from our text. Or we could also use the "alertblock" environment to make the color red. Note that using the base "alert" command will turn any associated text red. The same is true for the blocks and the "alertblock" environment specifically. To add these and get a sense of the differences, start with the following code using two different environments to see the differences in how the base themes format blocks differently. Type,

```
\begin{document}

\begin{frame}
\frametitle{Basics of Blocks}

\begin{block}{Basic Block}
$H_1$: Sample Hypothesis 1
\end{block}

\begin{examples}
    For example, here is a pre-formatted ``Examples" block
\end{examples}

\begin{alertblock}{Red ``Important" Block}
Caveat of $H_1$ $\rightsquigarrow$ Don't forget this!
\end{alertblock}

\end{frame}

\end{document}
```

Using the "Madrid" theme first, we see the colors are intense and the blocks are clearly separated from the rest of the frame as well as from each other,



But if we change the theme back to our original "CambridgeUS" theme, we see the blocks, though still clearly fromatted, are a bit more subtle and blending in with the frame a bit more,



19 Miscellanea & Final Words

As a final word, there are a few additional simple commands to allow you a little more flexibility to customizing your Beamer presentations. We will look at only three, though there are many more ways to customize your presentation. Include the following commands within the "mode" command braces in your preamble (remember to include the backslash as the command operator):

- Remove navigation symbols ⇝ "setbeamertemplate{navigation symbols}{ }"
- Remove the footer line in all frames ⇝ "setbeamertemplate{footline}"
- Include only a frame count at the bottom ⇝ "setbeamertemplate{footline}[page]"

I hope this soft introduction to Beamer has been useful, and that you feel confident to make well-formatted, good looking presentations using LaTeX. As always, there are so many resources online. Starting with a simple Google search is always a good place to begin looking for additional updates, designs, templates, customizations, and so on.