Introduction to LaTeX

Brendan Skip Mark

bmark2@binghamton.edu

S T A T I S T I C S

BINGHAMTON
UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Part I

The Program

1 Giving Credit Where It Is Due

I would like to thank Kerem Ozan Kalkan who ran that Latex workshop at ICPSR 2015 for a lot of this code and many of the examples. I'd recommend sitting in on this class if you forget everything about LaTeXby the time you go.

I would also like to thank Mert Moral who has taught me quite a bit about LaTeXand helped me fix an error in this document that was preventing it from compiling. Whether it is for LaTeX, STATA, R or any other problem one of the best resources you have are your fellow colleagues. Run problems by them or ask myself, Dave, or another person in the department. Most people are readily willing to help when you run into trouble or get stuck.

2 A shortcut to some FAQs

For those who are having trouble with Latex a useful guide for beginners can be found CLICK HERE which goes through a lot of the basics and is highly recommended

3 Introduction and Download Guide

A lot of this has been given in the readme file that Dave provided. Here is a quick recap

3.1 Distribution of LaTeX

(note everyone often signals that they use LaTeX by using this font for Latex)

Windows:
(MiKTeX Hyperlink) or (proTeX Hyperlink) guides you through the installation
(TeXLive Hyperlink)
Max:
(MacTex Hyperlink)
(XeTex Hyperlink) has additional font support

These are the "machines" that typeset in LaTeX

3.2 Text Editors

This is the main part of LaTeXwhich you will be working in. This is where you will enter the text and compile your document. This is a GUI (graphical user interface) that translates the commands you type into a finalized version which can be turned into a pdf or .tex file.

For Mac there is a free version which is the one I am using called (TexShop Hyperlink) (Mac only) this is included in the package from MacTex (see above)

For Windows users the most common text editor is (WinEdt hyperlink) it is $40 for student licenses and may be used for free for a short trial period.

Finally you will need a pdf viewer like adobe acrobat or preview on mac.

Part II
Parts of the LaTeX document

LaTeXdocuments have two main parts: the preamble and the body. The preamble sets up all the packages you will be using (this is similar to the library() command in R). This is where you pull up packages, set titles, fonts, and organize how the document will look. Below you can see an example of what the preamble looks like. We will go through each of these but the best source for questions about each package is google! Googling a package and looking around for questions about it or descriptions of it will often be the best source of information. There are too many packages to cover and each template you find will likely have different packages. Also note there are many different packages and commands which accomplish the same thing.

1 Preamble

The preamble is everything in between \documentclass[]{} and \begin{document}

to begin a .tex file we start with the {documentclass}[12pt]{artticle} function.
is a cool symbol
This tells LaTeXwhat kind of environment it will create. Next we load a number of packages with the usepackage command.

Below is an example of what a preamble in a document might look like.
[12pt]article report, slides, letter, memoir, and beamer
amsfonts, amsmath, amssymb, bm symbols which you can use. It essentially allows latex to understand math notation dcolumn, multirow aligns the text based on the decimal location
usepackage[colorlinks=true]hyperref you insert into the text

graphicx, subfigure, float [margin=1in]geometry setspaceverbatimto print exactly what you type in.

natbib

[,](),a,

the end of the preamble begins when we type \begin{document}. This marks a shift towards the beginning of the body, which is the second part of a .tex document

2 Body

This is where your graphics, papers, and all of the meat of your document goes. Remember that at the end of a document you must type \end{document}. In this .tex file you can see that all of this is included in the body. This is what we actually see when everything is compiled.

3 Compiling a document

Compiling a document translates all of the code you see into a pdf. This is a necessary step as the code alone is useless if it doesn't compile. This necessitates a warning and best practices suggestion: compile early and often. There is nothing more frustrating than writing an entire LaTeX file and hitting compile only to see numerous errors. Compiling at each stage of the document takes little time and will save you numerous headaches. Another suggestion is to ensure that the template you start with compiles properly before you begin.

In TeXShop we can compile a document by clicking the Typeset button in the top left.

Note that you will want to be working in pdflatex. "pdflatex" compiles your document as a pdf and can typeset a lot of the extensions (such as .jpg) your figues will be saved under. "latex" will compile your document into a .dvi file which needs to be converted to ps and then saved as a pdf. Long story short pdflatex saves you a lot of time.

LaTeX may often return errors and stop compiling the document. Sometimes you can force LaTeX to compile the document by entering q when the errors come back (or holding enter). As long as it is not a fatal error LaTeX will ignore any non-essential errors. It may also help to "Trash Aux Files" before trying to compile again.

The most common errors occur because you are using `[ ] instead of { } or [},` it may also occur because you open an environment but do not close it. Or you fail to add `\end{document}` at the end of your file. The console can provide useful information about where you errors are.

You can also try entering the code from the body of your document into programs like LaTeXiT which checks the code for errors. These programs act like a spelling check in microsoft word. Mixed results as they word imperfectly.

Part III
Useful commands

- \\ and \newline - move whatever comes after these to a new line. Make sure there is a line to end. If you simply type \\ you will get an error
- \newpage will move text to the next page
- \newpage moves onto the next page
- quotation marks are a bit different. Use the ` symbol under the key to create the left open quotation. To close a quotation use two single quotation marks. This will look like this in your command file ``' '
- each symbol we use in math has its unique code. For a list of common symbols and their commands see Symbols. An example of this is *epsilon* which is entered by typing \epsilon which produces the symbol ϵ. You need a $ before and after this for the command to work
- to create new sections and subsections use the \section \subsection and \subsubsection commands
- you will sometimes want to manually insert spaces between items. This can be done with the \ command. A single \ command produces a space. Alternatively you can use to \hspace{} or \vspace{} commands within the brackets enter a distance such as 1cm or 2cm.
- sectioning can be useful. in LaTeX we type the `\section{section 1}` command. We can also use subsections and subsubsections. If you use the `\subsection` command it will automatically generate a number next to it. However, if you would prefer no numbers you can add an asterisk as such: `\subsection*{}` . Below I will briefly show the code for both.

```
\section*{Section 1}
\subsection{Subsection 1}
\subsubsection{Subsection 1}
```

This will produce the following:

Section 1

0.1 Subsection 1

0.1.1 Subsection 1

Note that in order to reset the counter for sections we can use the \usepackage{chngcntr} and \counterwithin*{section}{part} commands in the preamble. This allows you to reset the section numbering everytime you start a new part which is done by entering the \part command

- \footnote{} puts a footnote with text between the brackets at the bottom of the page. If you would prefer to use endnotes you can use the endnotes.sty package to convert footnotes to endnotes.

Part IV
Useful environments

There are a lot of useful environments that can make your life a lot easier. Whenever you open an environment make sure to write the command to close it immediately. Also be careful when you keep environments open for too long as these can cause problems compiling your code.

- Enumerate uses numbers and letters.

```
The code for enumerate is as follows

\begin{enumerate}
\item First Bullet Point
\end{enumerate}

This produces the following
```

1. First Bullet Point

You can also enumerate environments. This code for example has multiple new enumerations. One thing to be careful of is to keep track of where enumerate begins and ends. If you miss one it will prevent the document from compiling.

```
\begin{enumerate}
\item First item
```

```
\begin{enumerate}
\item First sub-item
\begin{enumerate}
\item First sub-sub-item
\begin{enumerate}
\item First sub-sub-sub-item.
\end{enumerate}
\end{enumerate}
\end{enumerate}
\end{enumerate}
```

This produces the following

1. First item
    (a) First sub-item
        i. First sub-sub-item
            A. First sub-sub-sub-item.

Please note that the same can be done for itemize the only part of the code that changes is replacing the word enumerate with itemize.

You can change the itemize symbol by entering the symbol you want in the [] after the \item command. Try some of these

```
\item[$\circ$]
\item[$\bullet$]
\item[$\star$]
\item[$\blacklozenge$]
\item[$blacksquare$]
```

• The Verbatim Command is very useful. Look through this document's .tex file to see how often I use it. It tells LaTeXto replicate exactly what is posted

Part V

Text size and spacing

- You can alter the size of text within the document.

S T A T I S T I C s

This is done by using the following code

{\tiny} tiny

{\scriptsize} scriptsize

{\footnotesize} footnotesize

{\small} small

{\normalsize} normalsize

{\large} large

{\Large} Large

{\LARGE} LARGE

{\huge} huge

{\Huge} Huge

- You can use the \doublespace command to make text double spaced

"When his father told him about his alarm at having forgotten even the most impressive happenings of his childhood, Aureliano explained his method to him [...] with an inked brush he marked everything with its name: table, chair, clock, door, wall, bed, pan. He went to the corral and marked the animals and plants: cow, goat, pig, hen, cassava, caladium, banana. Little by little, studying the infinite possibilities of a loss of memory, he realized that the day might come when things would be recognized by their inscriptions but that no one would remember their use. Then he was more explicit. The sign that he hung on the neck of the cow was an exemplary proof of the way in which the inhabitants of Macondo were prepared to fight against loss of memory: ”This is the cow. She must be milked every morning so that she will produce milk, and the milk must be boiled in order to be mixed with coffee to make coffee and milk.” Thus they went on living in a reality that was slipping away, momentarily captured by words, but which would escape irremediably when they forgot the values of the written letters.” Gabriel Gárcia Márquez - One Hundred Years of Solitude 3.14

This is a wonderful quote not just because it is from one of my all time favorite books, but also because it is exactly how you should think of and annotate your .tex .do and .R files. Label everything because like Aureliano you will forget it all, even the simplest of commands. I cannot stree how important this is in any software you will work in this semester. There are countless times when a past .do file has saved me hours of re-creating code and countless times when poor annotation on a .do file has cost me hours trying to recreate what I had done in the past.

Part VI
Tables and Tabular Materials

These go in their own environments. Pay attention because for many of the homework assignments you will need to include a table of some sort.

the ampersand (&) separates the contents of each cell and each line is ended with \\

we use the c command to center something, l to left-center, and r to right-center

For example if we wanted to tabulate something we might use this code

```
\begin{center}
\begin{tabular} { l c r}
first cell & second cell & cell3 \\
second line & middle cell & 6th cell \\
you & get the & point
\end{tabular}
\end{center}
```

This code would produce the following output

first cell	second cell	cell3
second line	middle cell	6th cell
you	get the	point

Some useful commands for tables

- multiple columns and rows \multicolumn and \multirow commands can combine columns and rows. For the \multirow command you will need to add \usepackage{multirow} to your preamble

- multi-page tables \usepackage{longtable} can be used when your table takes up two or more pages

- table positioning positioning tables is easy when you put them in a float table environment.

  - commands h (approximately here) , t (top of page), b (bottomg of page) , p (put table in special page) , ! override internal LaTeXparameters, H (place table in this exact spot).

Ok lets create a table!

```
\begin{table}[ht]
\caption{Nonlinear Model Results}
\centering
\begin{tabular}{c c c c}
\hline\hline
Case & Method\#1 & Method\#2 & Method\#3 \\ [0.5ex] % inserts table %heading
\hline
1&50&837&970 \\
2&47&877&230 \\
3&31&25 &415 \\
4 & 35 & 144 & 2356 \\
5 & 45 & 300 & 556 \\ [1ex]
\hline
\end{tabular}
%\label{table:nonlin}
\end{table}
```

Table 1: Nonlinear Model Results

Case	Method#1	Method#2	Method#3
1	50	837	970
2	47	877	230
3	31	25	415
4	35	144	2356
5	45	300	556

Let's go through some of the code here. For the most part this is the only code you will need to create tables for your homework.

- \hline this creates a horizontal line.
- & This tells LaTeXto move to a new cell
- c lc c c this centers every cell and creates a vertical line in between the first and second column
- caption is how we label the table.
- t says to put the graph at the top of the page
- centering puts the table in the middle of the document

Part VII

Including graphs and other graphics

Including graphics such as a graph you created or a picture can be accomplished by loading the \usegraphics command in the preamble. The easiest way to make sure that the directories line up is by including the graphic in the same directory as the .tex file. If this is done then you can simply insert in the document \includegraphics[options]{graphich name} in your file.

If your .tex file is in a different directory than you .tex file then you can load the \graphicsdirectory{c:user/documents/} package and include the directory (note you can drag the folder into the brackets for the correct directory). Best practices would be to make sure these files are in the same location as directories can be easy to mess up and moved files can cause problems.
Some useful options include
• [width=2cm] or [height=3cm] allows you to adjust the scaling of the picture
• [width=textwidth] creates an image the same width as the text. • **keepaspectratio** keeps the original dimensions of the figure
• **angle** allows you to rotate the image

1 figure command

You can include figures by using the **figure** command. The figure environment is useful for creating a title, and automatically numbering your figures. Graphics can be included without the **figure** command but they cannot be labelled automatically without it.

Here is a sample of what the **figure** environment and syntax looks like

```
\begin{figure}[h]
\caption{Pugs are wonderful dogs}
\centerline{\includegraphics[keepaspectratio]{pug}}
\end{figure}
```

This will produce the following. Getting figures to properly align in the document can be tricky. So play around with the various float commands.

Figure 1: Pugs are wonderful dogs

Part VIII
Equations

LaTeXprovides a ton of flexibility for writing equations. However, there is a big startup cost. Learning the symbols and the way LaTeXinterprets them will take time. Once you have it down however, it will make it much easier to put equations in your papers and other documents.

A good online tutorial with plenty of examples can be found here `http://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf`

One useful application for Mac users is called Equation Maker. It has an easy to use palette and it converts the math symbols to latex language for you. You can copy the code right into latex and it is a great way to learn. However, it is not free.

Let's go through some equations and the syntax behind them.

If I were to enter the code:

```
$ Y*= \beta_1x^2_2 + \beta_2x_2 + \gamma_1W
+ \varepsilon_{1}, if \ Y* >0,  \ 0  \ \ otherwise $
```

We get this equation in LaTeX:

$Y* = \beta_1x^2_2 + \beta_2x_2 + \gamma_1W + \varepsilon_{1}, if \ Y* >0, \ 0 \ \ otherwise$

Or if we wanted to look at say the error structure of a bivariate probit model we might show it like this:

```
\left (
\begin{matrix}
\varepsilon_{1}   \\
\varepsilon_{2}
\end{matrix}
\right ) ~ N \left (
\begin{matrix}
 0 \\
 0
\end{matrix}
\right ), \left (
\begin{matrix}
 1 & \rho  \\
 \rho & 1
\end{matrix}
\right )
```

Which would produce this output:

$$\begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \end{pmatrix} \ N \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

Or if we wanted to replace some of the variables with their variable names we might do something like this:

```
$ Domestic \ Discontent = \alpha_{1}+\beta_{1}HumanRights + \beta_{2}Austerity
+ \gamma Z \ + \varepsilon  $
```

Which would produce:

$Domestic \ Discontent = \alpha_{1}+\beta_{1}HumanRights + \beta_{2}Austerity + \gamma Z \ + \varepsilon$

There are a ton of math symbols and practice and google will be your best friends for figuring this out.

Part IX
Bibliography

This may be one of the most useful functions of LaTeX. Even if you decide to use microsoft word to write your document you may want to consider using BibTex for your bibliography. BibTex creates a file that saves all of your citations.

I use the natbib package (see the preamble). Create a master .bib file. This is where all of your references should be stored. BibTex will take out the ones it need in any given document. This means you can autoformat your bibliography and spend much less time on this (with fewer mistakes) once you have entered a citation.

- you only need to type each reference once. Once you have it in your .bib file it can be included in any future document by simply typing its label
- you can alter you citation style easily and BibTex automatically changes every citation in a paper.

- if you cite multiple papers by the same authors it automatically orders them and adds "Arrow (1972a)" and does the same in your bibliography.

- your bibliography is automatically compiled, ordered, and put in the right citation style by simply typing \bibliography{research}. BibTex takes all of the information from you master .bib file and does all the work

Creating a .bib file

Mac

On a Mac you can place you bibliography file in the following folder `~/Library/texmf/bibtex/bib/`

PC

On a PC, using MikTex, you can put your bibliography file in the following folder:
`C:\Program Files\ Miktex 2.9 \bibtex\bib\ or`
`C:\Program Files\texlive\2012\texmf-dist\bibtex\bib`
(or appropriate path where you latex is)

I am using a program called BibDesk which is quite useful. You can attach the article you are citing to the citation. This makes it easy to organize and find every article you cite. It also makes it easy to enter bibliographical information as we will see.

You will need three lines of code to use the natbib package. First you will need to include `\usepackage{natbib}` in your preamble. Next you will need `\bibliographstyle` and finally you will need `\bibliography{bibliography}` . Without all three parts you cannot compile your bibliography. Calling the package forward is straightforward. The bibliography style tells LaTeX which style you want your bibliography to take (APA, Chicago, MLA). This will work as follows:

One of my favorite articles from Will's class was Olson. Olson (1993) argued that leaders give up their power in exchange for money. A good development argument which meshes well with Bates (2001) who says that when leaders are broke they give up their power in exchange for money. Both of which are the foundation of many development arguments made in political science.

Some common natbib citations include

- `\citep{olson1993dictatorship}` produces (Olson 1993)
- `\citet{olson1993dicatorship}` produces Olson (1993)
- `\citep[See][569]{olson1993dictatorship}` produces (See Olson 1993, 569)

- `\citeyearpar{olson1993dictatorship}` produces (1993)

You can find a list of all the common natbib references at natbib

In order to include the bibliography and citations you must first compile using LaTeX then compile using BibTex and finally compile again using LaTeX˙Thus there are three steps. Compiling with BibTex can also give you useful information about what is wrong with citations.

Some useful tips

Google scholar is a very useful resource. It can give you bibtex citations (always check these as they are hit or miss and you will have to enter your own citekey).

Find someone else's .bib file and steal all their citations! No use in starting from scratch.

you can find a number of useful LaTeX files at `http://spia.uga.edu/faculty_pages/monogan/computing/latex/` including a master bib file created by Jamie Monogan (called master bib) and an APSR bib style entitled apsr.bst (place this in /bibtex/bib folder).

Keep a consisten citekey. I use authors last name, year, first non-article work. So olson1993dictator is a citekey.

Tagging you citations with keywords for broad literatures will come in handy down the road. So if an article talks about development and you tag it for development then creating a lit review on development articles becomes much easier as they are all in one place with that tag.

This is just the tip of the iceberg! Hopefully this is enough to do all of your assignments but there is an almost endless number of packages and commands to learn. Good luck!

References

Bates, Robert H. 2001. *Prosperity and violence.* WW Norton.

Olson, Mancur. 1993. "Dictatorship, Democracy, and Development." *American Political Science Association* 87(3):567–576.