Introduction to Latex

AUG
2018

What is Latex and Why Bother?

Latex is a document preparation system that can create complex `.pdf` files that look great. A few examples of documents created with latex are all of the most recent papers published in economics.

In this class, we will solve various projects and write reports on the results. These reports will involve graphics, equations and Matlab code. There is no easy way of creating a good report using a regular text editor, like Word. Instead, we must use something more powerful, that allows us to easily type math equations, include and manipulate graphics and code, and that generates a report that looks good and is easy to read. This is why we will use Latex.

Installing Latex

Installing Latex is very easy, if done right. We will install Latex using a self-contained package. This package has all the binaries that are required to generate basic and complex `.pdf` files, a package manager (used to update packages), and a text editor that can compile latex files.

This package is very useful, since you can start using latex right away after the installation. The only downside is that it occupies a decent space in the hard drive (around 4gb). Keep in mind that the package takes a few minutes to download. Because the download size is also big, sometimes the installation files can get corrupted. If that happens, you will most likely get a mysterious error during install. In that case, just download the package again.

Mac

On a Mac, you should install the MacTex Distribution. Follow the link, and download the file MacTeX.pkg. The download size is approximately 3gb, so download it on a stable connection. After the download is complete, double-click the package file and follow the instructions.

Windows

On Windows, you should install the TexLive Distribution. Follow the link, which will download an executable file to your machine, and install it. If you need more instructions you can find them here.

Online

Nowadays you can also use Latex online. A good options is ShareLaTeX, which is fast and free.

Text Editors

All the Latex distributions mentioned before come with a text editor. For example, the Mac distribution comes with an editor called TeXShop, while the Windows distribution comes with TexWork. These editors are simple, but provide a powerful starting platform. They have a clean interface, with a simple to use typeset button (that transforms latex into pdf), and some form of error reporting in case something goes wrong (which will happen).

After you get used to writing latex documents using the basic text editor, you can look for other editors that might suit you better. However, the most important thing right now is to get good at working with the basic text editor. Nevertheless, here is a list of good editors that you might want to look at (in the future!):

- TexMaker (multi-platform, powerful and easy to use)

- LyX (multi-platform, powerful)
- Sublime Text 3 (multi-platform, paid, harder to use)
- texpad (mac-os, iOS, paid, powerful, easy to use)
- Emacs for Mac or Emacs for Windows (multi-platform, extremely powerful, very steep learning curve)

Again, first learn how to use Latex with the basic text editor.

Hello World!

Let's create our first document! To do so, open your Latex editor that came pre-installed with the package you just downloaded. Open a new file, and type the following:

```
\documentclass{article}

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

Save the file (.tex), and click on the "Typeset" button (or in the green play button). This will typeset your document, and generate a pdf file, which should open automatically. Congratulations, you have created your first pdf with latex!

The first line of the "code" says that the document we will generate is an article. Articles are used for scientific journals, presentations and short reports. This is the only document class we will work with. There are other document classes for slides, books and letters. More information on document classes can be found here.

The second line says that we are beginning our document. Everything that comes below this line is the content of our document, like the phrase "Hello world!". The last line says that we are ending the document.

This structure is the bare bones of any latex file. It states the type of file (article) and where the content is. Next, we will learn how to start organizing our file.

Sections

Now that we have our first document, we can add more text to it and organize it in sections and subsections. Type the following in your .tex file:

```
\documentclass{article}

\begin{document}
Hello world!
\section{My first section}
Some text. More text.

A new paragraph.
\section{My second section}
Now subsections!
\subsection{My first subsection}
Important text here.
\subsection{My second subsection}
Nobel prize winning argument goes here.
\subsubsection{Wow, a sub subsection!}
Some text.
\section{My last section}
Conclusion goes here.
\end{document}
```

The command `\section{Section Name}` creates a numbered section named "Section Name". When we typeset the file, the name of the section will automatically appear in a bold font and with a bigger font size. The section will also be automatically numbered, and everything that is typed below the section command is the content of that section. Creating a new section just require another `\section{New Section}` command. And to create a new subsection we use `\subsection{New Subsection}`. If we want to go a level deeper and create a sub-subsection, we just use the command `\subsubsection{New Sub-Subsection}`.

Now, we can type text, organize it in sections and subsection, and generate a good looking pdf. Next, we will learn how to type math!

Typing Math

Typing math in Latex is very natural. There are two types of equations that we can use in latex: in-line equations and "block" equations. The first type are equations that appear in the middle of lines, like this `$f(x)=x$`. We use dollar signs to indicate that what is in between them is a math equation. The second type of equations, are equations that are separate from the text, like in a paragraph by themselves. Let's see them in action:

```
\documentclass{article}

\begin{document}
\section{Inline Math}
An inline equation appears between dollar signs, like this $f(x)=x$.
\end{document}
```

Now, save the file and typeset it. You will see that the content between $ and $ was interpreted as a math equation, and displayed in an appropriate font and style.

The basic principle for typing math in latex is the following: type an equation as if you were reading it out loud. If you encounter a subscript, use `_` before it, and for superscripts use `^` instead. If you want to use a special symbol, like the symbol for an integral or for the square root, then type a backslash `\` and then the name of the symbol, like `int` for integrals. For example, to type `the integral from 0 to infinity of x squared` we would write `$\int_0^\infty x^2dx$`. The first piece is `\int_0^\infty`, which reads as the integral subscript 0 superscript infinity symbol. The second piece is `$x^2dx$`, which reads as x superscript 2 dx. Let's type that in the .tex file:

```
\documentclass{article}

\begin{document}
\section{Math Symbols}
Inline integral: $\int_0^\infty x^2dx$.
Now, with a square-root: $\int_{-\infty}^\infty\sqrt{x}dx$.
\end{document}
```

Save the file and typeset it. Notice that we used brackets for the `-\infty` subscript. We have to use brackets for subscripts and superscripts whenever there are more than one symbol that we want to put in the subscript or in the superscript. In the case of `-\infty` there are two symbols: the negative sign and the inifinity symbol. Also notice that the square-root used brackets to identify `x` as the argument.

Now, let's take a look at the second type of math equation:

```
\documentclass{article}
\usepackage{mathtools}

\begin{document}
\section{Block Equations}
We have some text here, and then an equation:
\begin{align}
\int_{-\infty}^{+\infty}\phi(x)dx=1
\end{align}
And more text down here.
\end{document}
```

Save it and typeset the document. Notice that now we have an equation that takes an entire line, and it is also numbered. To create this type of equation we had to use two things: a package (called mathtools) and an environment (called align). The package `mathtools` defines how latex should display the math equations inside the `align` environment. In this case, instead of using dollar signs, we use the `\begin{align}` and `\end{align}` commands to tell latex what is math.

Now, we know how to type and organize text, how to type inline equations, and how to type equations that appear separate from text. Next, we will learn how to add images to our files.

Adding Images

Now, we learn how to import figures we generate in Matlab to our latex file. Suppose we create a graph in Matlab and save it as `.png` image. For this example, you should download this .png image. In order to add it to our file, we type:

```
\documentclass{article}
\usepackage{mathtools}

\begin{document}
\section{Adding a Picture}
My figure:
\begin{figure}
    \includegraphics{figname}
\end{figure}
\end{document}
```

Save your file and typeset it. Now, you can see that the image was added to the pdf. Notice that we are still using the package `mathtools`, since it provides the functionality to add images to our latex file. There is a slight problem, the image is too big! To fix that, we tell latex that the figure should fit the width of the file:

```
\documentclass{article}
\usepackage{mathtools}

\begin{document}
\section{Fixing Figure Width}
My figure:
\begin{figure}
    \includegraphics[width=\textwidth]{figname}
\end{figure}
\end{document}
```

Save your file and typeset it again. Now, the image occupies the entire width of the pdf.

If the image resolution is low, we might want to make the image smaller than the width of the pdf. To do so, just "multiply" the `\textwidth` by a number smaller than 1:

```
\documentclass{article}
\usepackage{mathtools}

\begin{document}
\section{Smaller Figure}
My figure:
\begin{figure}
    \includegraphics[width=0.8\textwidth]{figname}
\end{figure}
\end{document}
```

The image is now smaller. However, it is not centered anymore. But that is easy to fix, just tell latex to center it:

```
\documentclass{article}
\usepackage{mathtools}

\begin{document}
\section{Center Figure}
My figure:
\begin{figure}
    \centering
    \includegraphics[width=0.8\textwidth]{figname}
\end{figure}
\end{document}
```

To add a caption to your image:

```
\documentclass{article}
\usepackage{mathtools}

\begin{document}
```

```
\section{Add Caption}
My figure:
\begin{figure}
    \centering
    \includegraphics[width=0.8\textwidth]{figname}
    \caption{5-minute prices of Morgan Stanley (MS) stock and the S\&P500 exchange-traded fund (SPY).}
\end{figure}
\end{document}
```

Now the figure will have a caption that identifies the number of the figure and what the figure represents. Notice that the figure is actually appearing before the text "My figure:". This happens because figures "float", and latex treat them as if they were not part of the text, displaying them whenever "it is possible". To avoid this behavior and make figures appear exactly where we expect them to appear, we use a package called `float` and tell latex that the figure should appear here `[H]`:

```
\documentclass{article}
\usepackage{mathtools}
\usepackage{float}

\begin{document}
\section{Fix Figure Position}
My figure:
\begin{figure}[H]
    \centering
    \includegraphics[width=0.8\textwidth]{figname}
    \caption{5-minute prices of Morgan Stanley (MS) stock and the S\&P500 exchange-traded fund.}
\end{figure}
\end{document}
```

And we get the final pdf, with figure centered, appearing in the correct place, and with a caption. Next, we learn how to add Matlab code to our file.

Adding Matlab Code

Suppose you have a Matlab script that you want to add to your report. In this example, we will use this script. To make latex understand that your code is a code, and is specifically a Matlab code, we use the packages: filecontents and matlab-prettifier.

```
\documentclass{article}
\usepackage{mathtools}
\usepackage{float}
\usepackage{filecontents}
\usepackage[numbered,framed]{matlab-prettifier}

\begin{document}
My code for cleaning Matlab's workspace and creating a figures folder:
\lstinputlisting[style=Matlab-editor,caption={clearWorkspace.m clears the workspace and create a figures folder}]{cle
\end{document}
```

The command `\lstinputlisting` takes a `style` argument, a `caption` argument, and the name of the Matlab script that you want to display. The `style` argument is always `Matlab-editor`, since we are only dealing with Matlab code. The `caption` argument is the caption that you want to add to your code, and it goes inside the curly brackets.

Save your file and typeset it. You will get a pdf that will display the code in the script you just downloaded, and it will also use the correct color for certain keywords, comments and functions.

Now, we know how to type and organize text, how to type and organize math equations, how to add figures with captions and that are centered, and how to add Matlab code to our latex files. We have all the tools that we need to write the reports for this class.

Next, we will put the pieces together and create a template for the reports.

A Template for Reports

Since we will be creating reports for every project, it makes sense to create a template file that we can re-

use. This template will include all packages that we have used so far,

Create a new `.tex` file, name it `preamble.tex` and write the following:

```
% PACKAGES
\usepackage[utf8]{inputenc}
\usepackage{mathtools} % math and figures
\usepackage{float} % make figure appear where we want with [H]
\usepackage{filecontents}
\usepackage[numbered,framed]{matlab-prettifier}
% these packages include more math symbols you might use
\usepackage{amsmath,amsfonts,amsthm,amssymb}


% PROJECT Specific Information to Fill Out
\newcommand{\LectureTitle}{High-Frequency Financial Econometrics}
\newcommand{\LectureDate}{\today}
\newcommand{\LectureClassName}{Project}
\newcommand{\LatexerName}{First and Last Name}
\author{\LatexerName}


% CONFIGURATIONS to make the report look better
\usepackage{setspace}
\usepackage{Tabbing}
\usepackage{fancyhdr}
\usepackage{lastpage}
\usepackage{extramarks}
\usepackage{afterpage}
\usepackage{abstract}

% In case you need to adjust margins:
\topmargin=-0.45in
\evensidemargin=0in
\oddsidemargin=0in
\textwidth=6.5in
\textheight=9.0in
\headsep=0.25in

% Setup the header and footer
\pagestyle{fancy}
\lhead{\LatexerName}
\chead{\LectureClassName: \LectureTitle}
\rhead{\LectureDate}
\lfoot{\lastxmark}
\cfoot{}
\rfoot{Page\ \thepage\ of\ \pageref{LastPage}}
\renewcommand\headrulewidth{0.4pt}
\renewcommand\footrulewidth{0.4pt}
```

The first part (PACKAGES) is what we have used so far, it contains the packages for typing math and include figures and Matlab code. The second part (PROJECT) contains information that we should fill in. It has a field for the class name and for your name. Go ahead and change the `First and Last Name` to your first and last name. This information will appear in the `.pdf` file when we typeset our latex file. And the third part (CONFIGURATIONS) contains configurations that make the report look better, and that we should not alter.

This file contains all the configurations that we will use to generate the reports. We created it so that we don't have to retype all the `\usepackage`'s every time we want to create a new latex file. To actually use this file, we have to include it in the file that will hold our project report. For example, create a file named `report.tex`, and type the following:

```
\documentclass{report}
\include{preamble}

\title{\LectureTitle: Project 1}

\begin{document}
\maketitle
\newpage

Content.

\end{document}
```

Let's go line by line. The first line is like always, with the difference that instead of having an `article` we have a `report`. Try switching from one to the other and typesetting the file to see what changes (not

much).

The second line includes the `preamble.tex` we just created. It basically imports all the configurations and packages into our new file, so that we do not have to retype everything again.

The third line is used to create a title page for your report.

The fifth line begins the document, telling latex where the content is. The sixth and seventh line create the actual title page.

You are now ready to write reports using latex!

But I Don't Know How to ...

That's alright. Just google it. Seriously, you are not the first latex user, and your question has most likely already been asked and answered by other people. Besides Google, another website that you can use to look for help is the TEX StackExchange. There you can searh for answered questions, and you can also ask questions to people that use latex everyday.

Conclusion

In this quick introduction to Latex we learned how to install latex, create files, add text and organize paragraphs into sections and subsections. We also learned how to type math, both inline and in "blocks", and also how to add figures to our reports. We also learned how to add Matlab code to our files, and we created a template for our projects.
