Latex lecture

Darren Homrighausen

August 31, 2015

1 What is LaTeX?

It is pronouned 'la' + 'tech' (the 'X' is actually a χ).

1.1 Summary of what it is/does

LaTeX is a compiled mark-up language. The main idea is that you write a document that includes all the formatting instruction and actual text. This document gets compiled by LaTeX into some device-independent file format, one of the following

- .dvi (which isn't used very often)
- .ps (which is not used very much and is relatively large)
- .pdf (pretty much the standard)

I'll go into some more details on this process later.

2 Software

- For Mac OS X, use MacTex.
- For Linux, you don't need to do anything (it is already installed).
- For Windows there are two major options:
  - MikTex.
  - TexLive.

  Unfortunately, I haven't used Windows since version 3.11 (and even then, I mostly used the DOS interface). So, I am essentially useless on trouble shooting problems for either of these programs. If you have any questions regarding their installation, please contact our system administrator Zube (Zube@stat.colostate.edu).

It is (somewhat) important to note that the software both provide the *compiler* and *editors*. The compilers are crucial, you can choose to use the editor or not.

2.1 Editors

- MacTex comes with TeXShop, which is a nice editor.
- Alternatively, there is Aquamacs (which is a Mac OS X version of Emacs). This has a very versatile interface and includes the awe-inspiring Emacs interface.
- Choose whatever editor you want and compile the source directly.

3 The basics of LaTeX

A basic LaTeX document is composed of several ingredients: Preamble, Top matter, body, and bottom matter.

3.1 Preamble

The preamble sets up all of the formatting instructions for you. Here is an example:

```
\documentclass{article}
\usepackage{amsmath,amsthm,amssymb,amsfonts}

\begin{document}
```

There are three components to a basic document (there are far more advanced things).

- documentclass: Determines what type of document you are making. You will mainly use article or specific classes developed by journals (we will return to this later).
- usepackage: this is how to load packages that contain functions you want to use. Here, we have loaded all of the math packages for making the appropriate math symbols.
- begin{document}: Tells latex to start looking for text.

As an aside, you can make LaTeX display things exactly as written (instead of formatting it) by using[1]:

```
begin{verbatim}
...by using\footnote{By the way, this is about as meta as I can stand.}:
end{verbatim}
```

[1] By the way, this is about as meta as I can stand.

3.2 Top matter

Here, you put all the meta information for the document.

```
\title{Example}
\author{Darren Homrighausen}
\date{\today}
\maketitle

\begin{abstract}
This is all so abstract.
\end{abstract}
```

3.3 Body

This is were the things you will see in the compiled document get written. For example:

```
\section{Introduction}
In the beginning...
```

3.4 Bottom matter

Fairly simple:

```
%Bibliography stuff

\end{document}
```

The bibliography stuff we will return to later. The ‘end document’ statement needs to go at the end of every document. Anything placed after it will not be compiled.

4 Math stuff

The main utility (though far from the only) of LaTeX is in displaying math. There are three main ways of doing math formatting:

There are two great resources for learning the syntax. These are the general and advanced wikibooks entries and the not-so-short introduction to LaTeX.

4.1 In-line

Suppose you want to write the math-equivalent of ‘let x be a d-dimensional vector’, then you could write

```
Let $x \in \mathbb{R}^d$ be given
```

Let $x \in \mathbb{R}^d$ be given

4.2 Broken-out, single line

Sometimes, you want to write some math on its own line. There are two main options for this:

```
\begin{equation*}
Y = \mathbb{X}\beta + \sigma Z
\end{equation*}
```

$$Y = \mathbb{X}\beta + \sigma Z$$

```
\begin{equation}
Y = \mathbb{X}\beta + \sigma Z
\end{equation}
```

$$Y = \mathbb{X}\beta + \sigma Z \tag{1}$$

4.3 Broken-out, multiple lines

Other times, you want to write some math, but it takes too much space. In this case, use

```
\begin{align*}
Y_i & = \beta_0 + \beta_1 X_{i1} + \ldots + \beta_1 X_{ip} + \sigma Z_i \\
    & = \mathbb{X}\beta + \sigma Z_i
\end{align*}
```

$$\begin{aligned} Y_i &= \beta_0 + \beta_1 X_{i1} + \ldots + \beta_1 X_{ip} + \sigma Z_i \\ &= \mathbb{X}\beta + \sigma Z_i \end{aligned}$$

```
\begin{align}
Y_i & = \beta_0 + \beta_1 X_{i1} + \ldots + \beta_1 X_{ip} + \sigma Z_i \\
    & = \mathbb{X}\beta + \sigma Z_i \\
\end{align}
```

$$\begin{align} Y_i &= \beta_0 + \beta_1 X_{i1} + \ldots + \beta_1 X_{ip} + \sigma Z_i \tag{2} \\ &= \mathbb{X}\beta + \sigma Z_i \tag{3} \end{align}$$

```
\begin{align}
Y_i & = \beta_0 + \beta_1 X_{i1} + \ldots + \beta_1 X_{ip} + \sigma Z_i \notag \\
    & = \mathbb{X}\beta + \sigma Z_i \\
\end{align}
```

$$\begin{align} Y_i &= \beta_0 + \beta_1 X_{i1} + \ldots + \beta_1 X_{ip} + \sigma Z_i \notag \\ &= \mathbb{X}\beta + \sigma Z_i \tag{4} \end{align}$$

4.4 Warning

It is tempting to use other equation formatting functions in LaTeX (eqnarray, $$,..). Don't due it. They will someday not be supported and it is just bad practice.

5 Figures

It is crucial to get the flow of figures into your documents going smoothly.

The first step is to put the graphicx package into the preamble:

```
%Preamble
\documentclass{article}
\usepackage{graphicx}
\usepackage{amsmath,amsthm,amssymb,amsfonts}
```

Let's make a sample figure:

```
x = rnorm(100)
y = 10 + 2*x + rnorm(100)

out = lm(y~x)
plot(x,y,xlab='X',ylab='Y')
abline(a = out$coeff[1],b = out$coeff[2])

pdf('regressionPlot.pdf')
plot(x,y,xlab='X',ylab='Y')
abline(a = out$coeff[1],b = out$coeff[2])
dev.off()
```

We can put this into LaTeXvia:

```
\begin{figure}[h!]
  \centering
  \includegraphics[width=3in]{./figures/regressionPlot.pdf}
  \caption{Caption for figure.}
  \label{fig:greenRed}
\end{figure}
```

5.1 Things not to do (and what to do instead)

Two common desires when writting papers is to want to put many figures in a single figure environment and to want to include a legend. The legend one is easy: don't do it. Just describe the figure in a caption[2].

[2]This is of course just my opinion/experience.

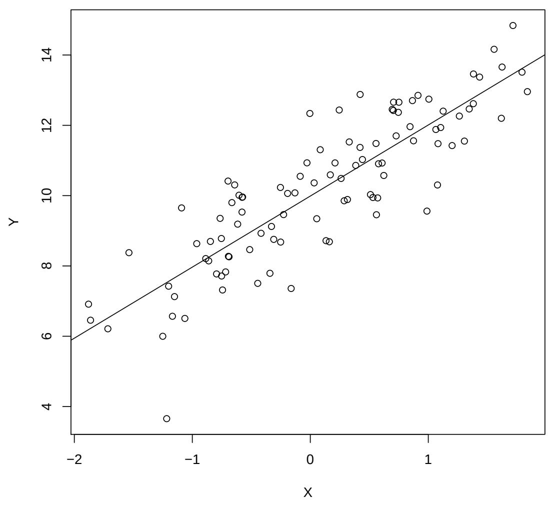


Figure 1: Caption for figure.

5.1.1 Multiple figures

It is tempting to try and include multiple figures in the same device through R, like so:

```
pdf('regressionPlotMultiple.pdf')
par(mfrow=c(2,2))
plot(x,y,xlab='X',ylab='Y',main='Regression')
abline(a = out$coeff[1],b = out$coeff[2])

plot(x,y - fitted(out),xlab='X',ylab='Y',main='Residuals')
abline(h=0)

plot(x,cooks.distance(out),xlab='X',ylab="Cook's distance",main="Cook's D")
abline(h=0)

Quantiles = y-fitted(out)
qqplot(x,Quantiles,main='QQplot')
abline(a=0,b=1)
dev.off()
```

I encourage you to **not** to do this, as you are 'locked in' to formatting (you can't change margins, rearrange figures, etc.). Instead, there are a couple of better options. First, generate 4 separate devices (not shown, but it should be clear how to do this).

You can put them into LaTeXvia:

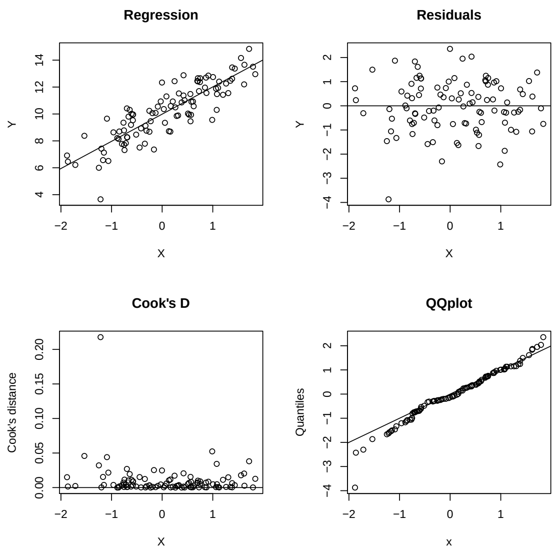


Figure 2: Putting all figures together with R.

```
\begin{figure}[h!]
  \centering
  \includegraphics[width=1.5in]{./figures/regressionPlotMultiple1.pdf}
  \includegraphics[width=1.5in]{./figures/regressionPlotMultiple2.pdf} \\
  \includegraphics[width=1.5in]{./figures/regressionPlotMultiple3.pdf}
  \includegraphics[width=1.5in]{./figures/regressionPlotMultiple4.pdf}
  \caption{Caption for figure.}
  \label{fig:greenRed}
\end{figure}
```

Alternatively, we can use the package ‘subfig’.

```
%Preamble
\documentclass{article}
\usepackage{graphicx}
\usepackage{subfig}
\usepackage{amsmath,amsthm,amssymb,amsfonts}
```

Now, instead we can put them into LaTeXvia:

```
\begin{figure}[h!]
  \centering
  \subfloat[Regression fit]{ \includegraphics[width=1.5in]{./figures/regressionPlotMultiple1
  \subfloat[Residuals plot]{ \includegraphics[width=1.5in]{./figures/regressionPlotMultiple2
  \subfloat[Leverage plot]{  \includegraphics[width=1.5in]{./figures/regressionPlotMultiple3
  \subfloat[QQ-plot]{  \includegraphics[width=1.5in]{./figures/regressionPlotMultiple4.pdf}
```

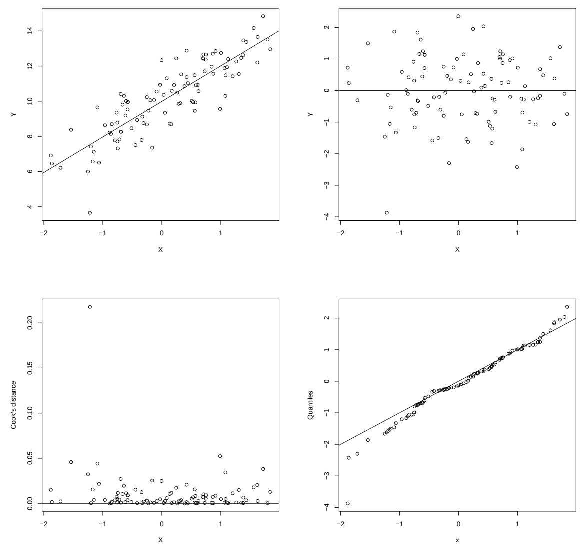


Figure 3: Caption for figure.

```
  \caption{Caption for figure with subfig.}
  \label{fig:greenRed}
\end{figure}
```

5.2 File Formats for Figures and Compilation

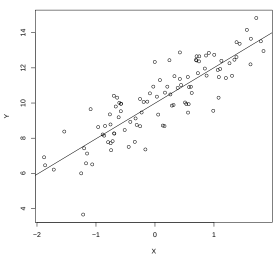
The contents of this section are becoming less important. However, it still comes up and it is better to know what you are getting youself into. The main idea is that LaTeX was designed to do the following:

1. The .tex file gets compiled to a .dvi file.
2. The .dvi file gets converted to a .ps file (the unix command is dvips, incidently)
3. The .ps file gets converted to a .pdf

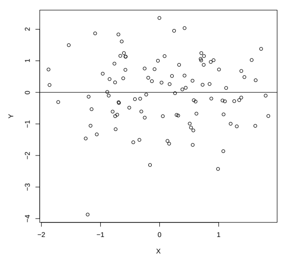
The reason for this is that while Adobe Systems made the PDF specification available free of charge in 1993, PDF remained a proprietary format, controlled by Adobe, until it was officially released as an open standard on July 1, 2008.

Now, most (all?) LaTeX interfaces come with `pdflatex`, which simply creates a .pdf from a .tex file directly.

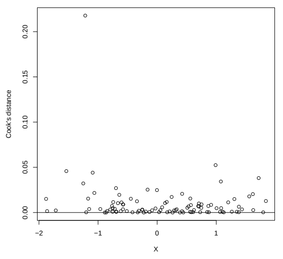
This is meaningful for you as it determines what file format you need your figures in. If you use the original specification, your figures must be .eps (the 'e' stands for encapsulated, which essentially has additional header information



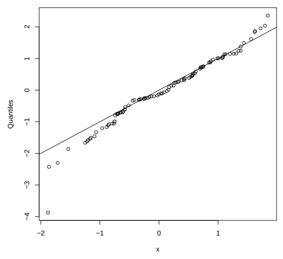
(a) Regression fit



(b) Residuals plot



(c) Leverage plot



(d) QQ-plot

Figure 4: Caption for figure with subfig.

that gives a bounding box for the figure). Alternatively, if you use `pdflatex`, the figures must be .pdf. Note: I'm aware that other file formats can be read in (e.g. .png, .jpg, ...). Don't use these as they are not vector graphics and hence can't be scaled.

Conclusion: The take-away message is to use .pdf figures and `pdflatex`. Some stodgier journals still require you to use .eps figures, though, so be careful what you're doing from the beginning so you don't have to regenerate a bunch of things (as an aside, you can always go .ps to .pdf. The opposite can be trickier).

6 Referencing

A crucial part of any scientific document will be referring to previous or forthcoming parts in the paper. This could of course be hard coded in, but that is almost surely going to cause problems at some point. Luckily, LaTeXallow for self-consistent and automatic referencing. To wit:

```
\begin{equation}
Y = \mathbb{X}\beta + \sigma Z
\label{eq:regression}
\end{equation}
```

```
As we can see in equation \eqref{eq:regression}, ...
```

$$Y = \mathbb{X}\beta + \sigma Z \tag{5}$$

As we can see in equation (5), ...

Now, if we move this equation or add ones before it, we will always be referencing the correct equation. As an aside, LaTeX actually compiles parts of the document at a time. So, it is common to see the following instead:

as we can see in equation (?), ...

Simply compile the document another time and this should go away (this is because LaTeX first builds an index of references and then actually adds the references after a second compilation). If not, check the output for references to non-existing labels.

Note that we can refernce anything that is an environment (theorems, equations, sections, tables, figures, ...).

7 Functions and Redefining Commands

Suppose we are writing a paper and we are continually writing

```
\mathcal{G}
```

which looks like $\mathcal{G}$. Now, two things could happen. The first is that a referree or coauthor could say 'I don't like script G for this, let's make it Ω, instead.' We could find and replace it, but what if we are using Ω for something else? A second things is that we get sick of writing the whole thing out. We can solve both of these problems at the same time. Returning to the preamble

```
%Preamble
\documentclass{article}
\usepackage{subfig}
\usepackage{graphicx}
\usepackage{amsmath,amsthm,amssymb,amsfonts}
%Ignore for now%%% \usepackage[options]{natbib}
%Ignore for now%%% \renewcommand{\abstractname}{Not abstract}
\usepackage[colorlinks]{hyperref}

\newcommand{\G}{\mathcal{G}}
```

Now, if we type

```
$\G$
```

we get $\mathcal{G}$.

Additonally, we can do more complicated things, as well. What if we want to talk about a likelihood function at a fixed data vector evaluated at a parameter, conditional on other parameters.

Returning to the preamble

```
%Preamble
\documentclass{article}
\usepackage{subfig}
\usepackage{graphicx}
\usepackage{amsmath,amsthm,amssymb,amsfonts}
%Ignore for now%%% \usepackage[options]{natbib}
%Ignore for now%%% \renewcommand{\abstractname}{Not abstract}
\usepackage[colorlinks]{hyperref}

\newcommand{\G}{\mathcal{G}}
\renewcommand{\L}[3]{\mathcal{L}_{#1}(#2|#3)}
```

This gives:

```
\L{Y}{\theta}{\eta}
```

$\mathcal{L}_{Y}(\theta|\eta)$

7.1 Using .sty files

Often, you will end up with many such definitions. Or, you will end up using the same series of definitions in multiple documents. It is very useful to keep such information in a central file and import it.

Create a document called ‘exampleDefs.sty’ and have it contain:

```
\newcommand{\G}{\mathcal{G}}
\renewcommand{\L}[3]{\mathcal{L}_{#1}(#2|#3)}
```

Then, add this to your preamble

```
%Preamble
\documentclass{article}
\usepackage{subfig}
\usepackage{graphicx}
\usepackage{amsmath,amsthm,amssymb,amsfonts}
%Ignore for now%%% \usepackage[options]{natbib}
%Ignore for now%%% \renewcommand{\abstractname}{Not abstract}
\usepackage[colorlinks]{hyperref}

\usepackage{exampleDefs}
```