A Short Introduction to LaTeX

Petra Harwin

October 2, 2012

Contents

1 Introduction to LaTeX and Unix

Aim: To give the student a basic working knowledge of using LaTeX to write mathematical documents.

Recommended books:

- F. Mittlebach & M. Goossens, "The LaTeX Companion" (Second edition), Addison-Wesley, Harlow, England (2004).
- Helmut Kopka & Patrick W. Daly, "A guide to LaTeX" (Third edition), Addison-Wesley, Harlow, England (1999).
- David F. Griffiths, "Learning LaTeX", Society for Industrial and Applied Mathematics, (1997).
- Leslie Lamport (illustrations by Duane Bibby), "LaTeX a document preparation system user's guide and reference manual", Addison-Wesley, Harlow, England (1994).

1.1 Why learn LaTeX?

- It typesets mathematical documents much better than Microsoft Word does.
- Making small alterations to the style of long documents is easy.
- Referencing of equations, figures and tables is fully automated.
- Long bibliographies can be dealt with easily using BibTeX.
- The `.tex` source files are ASCII text - platform independent.

- Versions of LaTeX are available for almost every computer platform.
- The output file can be either `.pdf` (Adobe Portable Document Format), `.ps` (PostScript) or `.dvi` (Device Independent Format).
- Most journals encourage the use of LaTeX.

1.2 Getting started with LaTeX

Which text editor?

- In UNIX and Linux I use emacs because it knows a lot about LaTeX and BibTeX.
- In Windows I use Texnic center and Miktex (a Windows version of TeX).
- For the purposes of this course I shall assume that you will all use emacs.

2 Getting started with Unix

For the purposes of these sessions we will be writing and compiling our LaTeX documents using the BUCS Unix servers. To open a Unix terminal on a BUCS machine click

Start $\rightarrow$ All programs $\rightarrow$ Xming $\rightarrow$ LCPU

The first time you do this a small box will appear asking you if you still want to connect. Click OK. You will be prompted to enter your BUCS username and password. Once you've done this a terminal will appear with the prompt `bash-3.2$` in it. This is the Unix shell. We type all our commands here and execute them by pressing return. If you wish to close the Unix terminal then type "exit" and press return.

`ls` (list)

When you first login, your current working directory is your home directory. Your home directory has the same name as your user-name, for example, mappjh, and it is where your personal files and subdirectories are saved (your H drive on Windows). To find out what is in your home directory, type

```
ls
```

The `ls` command (lowercase L and lowercase S) lists the contents of your current working directory. There may be no files visible in your home directory, in which case, the Unix prompt will be returned. Alternatively, there may already be some files inserted by the System Administrator when your account was created or you may have created some yourself in Windows.

`mkdir` (make directory)

We will now make a subdirectory in your home directory to hold the files you will be creating and using in this course. To make a subdirectory called LaTeXlab in your current working directory type

```
mkdir LaTeXlab
```

To see the directory you have just created, type

```
ls
```

`cd` (change directory)

The command `cd directory` means change the current working directory to 'directory'. The current working directory may be thought of as the directory you are in, i.e. your current position in the file-system. To change to the directory you have just made, type

```
cd LaTeXlab
```

Type `ls` to see the contents (which will be empty) .

Note: typing cd with no argument always returns you to your home directory. This is very useful if you are lost in the file-system.

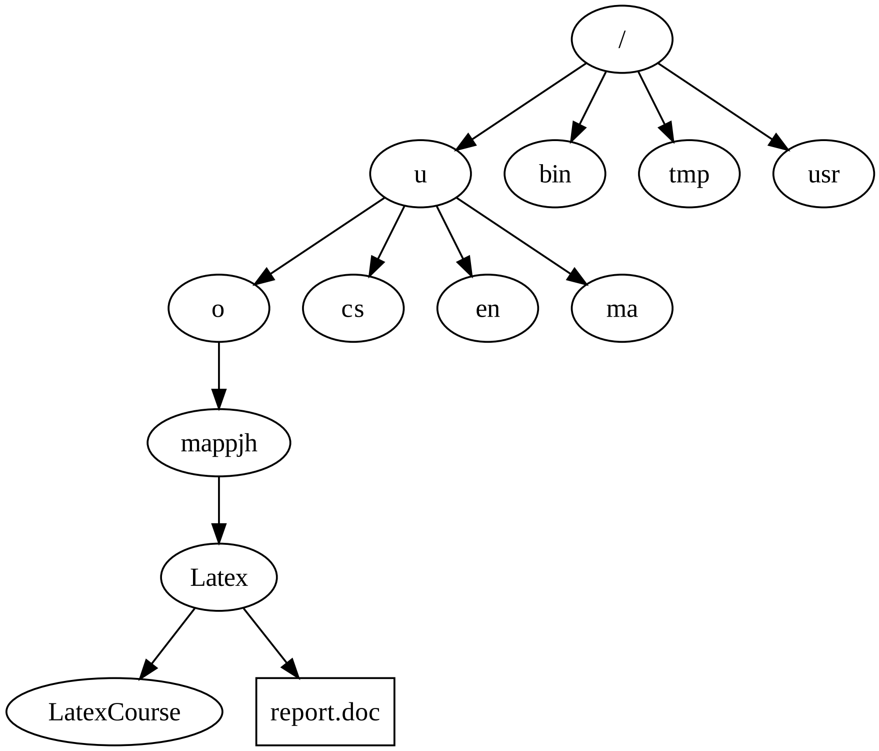
Exercise

Make another directory inside the LaTeXlab directory called backups.

`pwd` (print working directory)

All files are grouped together in the directory structure. The file-system is arranged in a hierarchical structure, like an inverted tree. The top of the hierarchy is traditionally called root (written as a slash /).

In the diagram below, we see that my home directory "mappjh" contains a sub-directory called Latex and this subdirectory contains a subdirectory called LatexCourse and a file called report.doc. The full path to the file report.doc is “/u/o/mappjh/Latex/report.doc”



Pathnames enable you to work out where you are in relation to the whole file-system. For example, to find out the absolute pathname of your home-directory, type cd to get back to your home-directory and then type

```
pwd
```

The full pathname will look something like

```
/u/o/mappjh
```

which means that mappjh (my home directory) is in the sub-directory ‘o’ (the group directory), which in turn is located in the ‘u’ sub-directory, which is in the top-level root directory called ‘/’.

First type cd to get back to your home-directory, then type

```
ls LaTeXlab
```

to list the contents of your LaTeXlab directory. Now type

```
ls backups
```

You will get a message like

```
backups:  No such file or directory
```

The reason is that the backups directory is not in your current working directory. To use a command on a file (or directory) not in the current working directory, you must either move to the correct directory using `cd`, or specify its full pathname:

```
ls LaTeXlab/backups
```

If you do this now you will see that nothing is returned because the backups directory is currently empty.

Now we've got some basic knowledge of Unix we can learn some LaTeX and begin to write a skeleton document to work with.

3 Basic Structure

Every LaTeX document has the following basic structure

```
\documentclass[options]{class}
    preamble
\begin{document}
    the body of the document
\end{document}
```

where *class* determines the style of the document and can be `article`, `book`, `report` or `letter` among others; while *options* allows you to specify the font size, format and paper type.

3.1 How LaTeX interprets spaces

- LaTeX treats single spaces, multiple spaces and newlines as a single space.
- Leaving a blank line (or multiple blank lines) tells LaTeX to start a new paragraph.

- LaTeX automatically leaves the right amount of space after the various types of punctuation even if only a single space is typed.
- To create an unbreakable space, i.e. one which will not be used to break the line, use the character `~`. For example to stop Latex from splitting P. Harwin after the P., type `P.~Harwin`.
- If you type the character `%` on its own then LaTeX will ignore the rest of the text until it encounters a new line. For example ``` ``You can't see %this text'' ``` produces “You can't see

4 Using emacs

Now that we know how to write a very basic document we're going to learn how to use emacs.

Let's use emacs to create a blank file called first.tex in your LaTeXlab directory. To do this move to your LaTeXlab directory and type

```
emacs first.tex &
```

(NOTE: The & is important since it tells shell to run this in the background. If we didn't do this then we would not be able to type anything further into shell until we closed emacs.)

If first.tex exists then emacs will open it. If it does not exist then emacs will create a empty file called first.tex. When emacs opens you'll see a empty file and if you look at the bottom left you'll see a little message saying first.tex. The white space below the place where the message appears is called the minibuffer. The minibuffer prints out useful messages to tell you what emacs is doing.

We can now begin to type a LaTeX document in this empty file.

Exercise

Create a LaTeX article containing only text.

To save this file go to the files menu and then to save. If you now type `ls` in shell you will see that the file exists.

5 Compiling

Now you have your document we need to learn how to compile it. In shell type

```
latex first.tex
```

and press return. Some messages will appear from your LaTeX compiler and you will either see `Output written on first.dvi. Transcript written on first.log.` or some error messages.

5.1 Finding and fixing errors

When you try to compile a file and it produces an error message LaTeX will reproduce the part of the document where the error occurred and attempt a fix (so it can recover, ignore the error and continue compiling). If you can't spot the error type H and press return for a further hint. If you can spot the error then press X to exit. If you accidentally press return instead of X you will sometimes get a $*$ prompt instead of a `?` prompt (or it may move onto the next error depending on whether its attempted fix worked). If you get the $*$ prompt then hold Ctrl and press C to get back to the `?` prompt so that you can type X.

5.2 Viewing your file

If your file has compiled without errors then view the file using the command

```
xdvi first.dvi &
```

If you have errors, try to work out what you've done wrong. If you can't then call either me or Stephen over.

6 Commands in LaTeX

There are three main types of commands in LaTeX

1. The characters `& $ ^_ { } ~ # %` all tell LaTeX to do something.
2. Backslash \ followed by a single non-letter character. For example, to produce the character & I must type `\&`.

3. Backslash followed by one or more letters.

The third type of command generally has the structure
`\commandname[optionalarguments]{mandatoryarguments}`.

6.1 Environments

Environments have the following structure

```
\begin{environment}
   body of environment
\end{environment}
```

The text inside the environment is treated differently from normal text. For example

```
\begin{center}
some centred text
\end{center}
```

gives

some centred text

6.2 Declarations

A declaration is a special type of command that changes the values or meanings of certain parameters or commands without immediately printing any text. Its effect is immediate and ends only when another declaration of the same type is encountered. However, if a declaration occurs within an environment or a {...} pair, its scope only extends to the corresponding `\end` command or closing brace}.

For example `{\footnotesize This is footnotesize.}` gives This is footnotesize.
and

```
\begin{center}
\footnotesize This is footnotesize.
\end{center}
```

produces

This is footnotesize.

The size of the text outside the environment or $\{\ldots\}$ pair does not change.

7 Font sizes in LaTeX

To change the size of the font in a small section of the text, the following declarations may be used:

`\tiny` (tiny)	`\scriptsize` (scriptsize)
`\footnotesize` (footnotesize)	`\small` (small)
`\normalsize` (normalsize)	`\large` (large)
`\Large` (Large)	`\LARGE` (LARGE)
`\huge` (huge)	`\Huge` (Huge)

It is important to note that each of these is relative to the font size specified as an option in the `\documentclass` command.

7.1 Exercise

Modify your document so it contains at least two paragraphs, some centred text and some text of a different size. Compile and view your modified document.

8 Structuring a Document

Structure may be added to your document by using sectioning commands. The following commands are available for producing automatic sequential sectioning:

```
\part      \chapter      \subsection       \paragraph
           \section      \subsubsection    \subparagraph
```

With the exception of `\part`, these commands form a sectioning hierarchy. In document classes `book` and `report`, the highest sectioning level is `\chapter`. The chapters are divided into sections using `\section` and further subdivided using `\subsection` and `\subsubsection` and so on. In document class `article`, the hierarchy begins with `\section` since chapter is not available.

For each sectioning command there is an internal counter that is incremented by one every time that command is used, and reset to zero on every use of the next highest sectioning command. The syntax for these commands is either of these

```
\section_command[short_title]{title}
\section_command*{title}
```

In the first case, the section is given the next number in the sequence, which is then printed together with a heading using the text `title`. The text of `short_title` becomes the entry in the table of contents (and on the page head if the page style `headings` has been selected). If the option `short_title` is omitted, it is set equal to `title`. In the second case, the section counter is not incremented, no section number is printed and no entry is made in the table of contents.

8.1 Labelling

At some point you may wish to refer to a section of your document. You could just type in the section number but if you were to enter extra material then that section number could change. The LaTeX cross-referencing system helps us out here. Simply put the command \`label`{`name`}`, where name is a simple label of your choice, just after the sectioning command. Then you may refer to that section by using the command \`ref`{`name`}`. For example, typing

```
\section{A First Approximation}\label{first}
In Section \ref{first}
```

will produce the appropriate section number in place of the command \`ref`{`first`}`.

8.2 Exercise

Add sections to your document and refer to them using the \ref command. Compile your document. Note: LaTeX may ask you to run it again so that it can get some references right. It does this when you number equations or cite references. We will get onto why this happens later in the course. Assuming you don't have any error messages and you have run LaTeX as many times as you need, you can now view your LaTeX output by typing

```
xdvi first.dvi &
```

at the prompt. (If you kept xdvi open from the last time you viewed your file then its contents will have updated automatically so you need not open it again.)

9 Printing

Once you have checked for typographical errors etc. you are ready to convert your file to a form from which it can be printed.

- To convert your file to PostScript type `dvips myfile.dvi -o myfile.ps` at the prompt. This will create a file called `myfile.ps` that you can view by typing `gv myfile.ps` at the prompt.
- To convert your file to a `.pdf` it is best to create the file `myfile.ps` as above and then type `ps2pdf myfile.ps myfile.pdf` at the prompt. This creates a file called `myfile.pdf` that can be viewed by typing `acroread myfile.pdf` at the prompt.

9.1 Printing from Unix on the BUCS network

You can only print to the library's printers from this network. Should you wish to do so use the commands below but remember you'll have to pay to retrieve your printing.

- Printing `myfile.ps` in gv: Print by selecting file, then print and use the print command `lpr -Pll` to print on the library laser printer.
- Printing `myfile.pdf` in acroread: Use the same printing commands as in gv.

9.2 Printing from a Windows PC

To print to a printer other than those in the library you'll need to open your files in Windows. Most machines have Adobe Reader, some have GSview.

- Printing `myfile.ps`: Open the file with GSview and print from there.
- Printing `myfile.pdf`: Open the file with Adobe Reader and print from there.

10 Making a title

The `\maketitle` command is a quick and easy way to make a basic title for any document. Simply give the information

```
\title{Your Title}
\author{Your name}
\date{The date}
```

in the preamble and then type `\maketitle` where you want the title to appear.

If you don't like the layout this gives then you can make your own title page using the `titlepage` environment:

```
\begin{titlepage}
Title page text
\end{titlepage}
```

Exercise

Add a title to your document using the \maketitle command.

11 A Simple Document

Here's a simple document. I'll explain all the new commands as we go through the course. For now, use this as a reference point if you want to create a new article with an abstract and a table of contents.

```
\documentclass[12pt]{article}
\title{Our First Article}
\author{Petra J. Harwin\\University of Bath}
\date{\today}
\begin{document}
\maketitle
\begin{abstract}\addcontentsline{toc}{section}{Abstract}
Here is my abstract.
\end{abstract}
\tableofcontents
\section{This is a Section} \label{first sect}
The body of the Section \ref{first sect} goes here.
\subsection{This is a Subsection}
```

```
\label{first subsect}
The body of the Subsection \ref{first subsect} goes here.
\end{document}
```