LaTeX Level 4
Further document preparation

Susan Hutchinson

Department of Statistics, University of Oxford.

June 2009

Course outline and structure

1. Revision
2. Planning and managing longer documents
   - Exercise: creating, compiling and viewing simple documents
   - Exercise: managing longer documents
3. Customising LaTeX: creating and changing commands
4. Creating slides using the Beamer class
   - Exercise: creating and customising commands
   - Exercise: make your own slide show
5. Exploring packages
6. Going further: finding answers, asking good questions
   - Exercises: exploring packages
   - Exercise: tips and tricks to solve problems yourself
7. Conclusion

Our LaTeX environment

- We will use OSS Watch Ubuntu Linux. Ubuntu Linux runs from CD without touching the hard disk.
- This version of Ubuntu comes from OSS Watch, the Open Source advisory service based at OUCS.
- A good version of LaTeX is included on the CD.
- Familiarity with a Linux user interface is assumed.

How does LaTeX work?

In order to use LaTeX two components are needed.

1. A LaTeX engine or distribution
   In Windows the most commonly used distribution is MiKTeX. It provides all the infrastructure for creating documents such as fonts, style files, compilation and previewing commands and much else. Another popular distribution is TeXLive which is widely used in Linux; Mac users generally use MacTeX.

How does LaTeX work?

In order to use LaTeX two components are needed.

1. A LaTeX engine or distribution
   In Windows the most commonly used distribution is MiKTeX. It provides all the infrastructure for creating documents such as fonts, style files, compilation and previewing commands and much else. Another popular distribution is TeXLive which is widely used in Linux; Mac users generally use MacTeX.
2. An editor or IDE (Integrated Development Environment)
   This is used to edit, compile and preview LaTeX documents. There are many editors and IDEs available. Emacs is a popular editor which is available for both Windows and Linux; Lyx is available for both too. Several other platform-dependent editors are available such as Kile in Linux, TeXNicCenter, WinEDT and Winshell in Windows and Aquamacs for Macs. In Linux a simple text editor such as gedit can be a good option.

Developing LaTeX documents

- LaTeX works on an edit → compile → view cycle.
- Use gedit (or some other editor) to create, save and edit your `.tex` file.
- Use a terminal window to execute commands to compile and view the file.
  - Use `pdflatex file.tex` command to compile.
  - Use `xpdf file.pdf &` to view. Don't forget the `&` at the end of this command.
- Image files — plots, graphs, pictures — can cause confusion.
  - Images in PostScript format (`.eps` or `.ps`) are compiled with `latex` to produce a `.dvi` file, previewed with `xdvi` and printed with `dvips`.
  - Images in PDF, PNG, or JPG format are compiled with `pdflatex` producing a `.pdf` file which can be previewed and printed. This is what we'll be using today.

A simple LaTeX file

```
\documentclass[a4paper,12pt]{report}
\begin{document}
\chapter{Introduction}
Hello there. This is the first paragraph.

Goodbye now. That's it.
\end{document}
```

A simple LaTeX file explained

The text of the document is surrounded by commands.

```
\documentclass[a4paper,12pt{report}
```

. . . This bit is called the preamble . . .

```
\begin{document}
```

. . . This bit is called the body . . .

```
\end{document}
```

Planning a longer document

When writing a longer document such as thesis there are many advantages to breaking your document into separate files.

Planning a longer document

When writing a longer document such as thesis there are many advantages to breaking your document into separate files.

- Imposes a structure on the document as a whole.

Planning a longer document

When writing a longer document such as thesis there are many advantages to breaking your document into separate files.

- Imposes a structure on the document as a whole.
- Allows you to focus on each part separately.

Planning a longer document

When writing a longer document such as thesis there are many advantages to breaking your document into separate files.

- Imposes a structure on the document as a whole.
- Allows you to focus on each part separately.
- Conditional compilation can speed up the compile $\longrightarrow$ edit $\longrightarrow$ preview cycle.

Planning a longer document

When writing a longer document such as thesis there are many advantages to breaking your document into separate files.

- Imposes a structure on the document as a whole.
- Allows you to focus on each part separately.
- Conditional compilation can speed up the compile $\longrightarrow$ edit $\longrightarrow$ preview cycle.

It is also possible to create a standard preamble which can be used whenever you start a new LaTeX document.

Structuring a longer document

When dividing up a longer document into separate files you need to do the following.

- Create a root file which is the file you use when compiling. This file contains markup which points to the files you want to include.
- When including files the `.tex` suffix is not needed.

So if your `main.tex` is the root file which contains the `\include` commands then you should run `pdflatex main.tex`

How to break up a long document

```
\documentclass[a4paper,12pt]{report}
\begin{document}
\title{My long document}
\author{My Name}
\date
\maketitle
\include{intro}
\include{theory}
\include{research}
\include{results}
\include{conclusion}
\end{document}
```

In this case the document is organised into five separate files called `intro.tex`, `theory.tex` and so on, each of which will contain a single chapter.

Conditional compilation

There are many reasons why you may want to compile only one or two chapters. It is particularly useful if one chapter contains a lot of images and is slow to compile.

- Make sure that one compilation is done with all the chapters.
- Add the line

  `\includeonly{intro,theory}`

  in the preamble and recompile.
- Only the first two chapters will appear, but cross-referencing will be preserved, even to the chapters that have been left out.

Alternatively the `\excludeonly` command can be used to exclude one or more files.

Revision: labels and cross-references

LaTeX makes it easy to create cross-references to other parts of your document. You can create references to equations, figures, chapters, sections, pages, tables and so on. To create a reference to a chapter and page use the following:

- Add the following markup

  ```
  \label{ch:intro}
  ```

  at the beginning of the chapter. Labels are case-sensitive so INTRO is different from intro.
- Then when you want to make reference to the chapter add

  ```
  In Chapter \ref{ch:intro} on \pageref{ch:intro}
  ```

  when to reference the chapter and the page number.

Remember that you may need to compile your LaTeX file twice (at least) to get references right.

Exercises

Now do the following exercises.

- Creating, compiling and viewing a simple document using gedit
- Managing longer documents

Customising LaTeX

It is possible to define new commands and redefine existing ones. You may want to do this for several reasons.

- To save typing. If you keep typing the same markup it makes sense to define a command in the preamble and use the shortened version in the document.
- To keep a consistent style.

It is possible to create new commands that are shortcuts to existing commands.

```
\newcommand{\bi}{\begin{itemize}}
\newcommand{\ei}{\end{itemize}}
```

Creating your own commands

For example instead of writing

```
{\textrm {\slshape{Department of Statistics}}}
```

every time I want to write *Department of Statistics* I can define a new command using

```
\newcommand{\DOS}{\textrm {\slshape Department
of Statistics}}
```

and use \DOS instead.

It is easy to change your mind later – the change would only need to be made in one place.

In slides the default font is sans–serif so in order to make my new command clearer I use a Roman font.

Defining a new command with an argument

The syntax of the command to define a new command is

```
\newcommand{cmd}[n-args][default]{text}
```

where

- cmd is the name you decide to use for the new command.
- n-args [optional] is the number of arguments your new command will have.
- default [optional] what to use as a default argument.
- text what LaTeX should do with when this command is used.

To define a command called *keyword* which typesets a word or phrase in bold using a typewriter font and coloured blue add to the preamble

```
\newcommand{\keyword}[1]{\texttt{\slshape\color{blue}#1}}
```

So to typeset the word "section" as a keyword use `\keyword{section}` which would appear as ***section***.

Customising an existing command

Using \renewcommand the behaviour of existing commands can be changed. For example to change the page with your table of contents to say "Table of Contents" rather than "Contents" you would use

```
\renewcommand*{\contentsname}{Table of Contents}
```

There are many predefined names which can be changed.

```
\contentsname
\listfigurename
\listtablename
\figurename
\chaptername
```

Note that some names will only apply to some documentclasses. For example there is no Chapter command when using the *article* class. The use of an asterisk (star) in a \renewcommand indicates that the command is *short* and unlikely to include a paragraph break.

Advice on customising commands

Customisation needs to be done with care. LaTeX imposes good typesetting rules on your document so you need to make sure your changes are necessary.

- LaTeX does not allow you to create new commands with the same name as existing commands.
- Check that there isn't a package which will achieve the effect you are looking for, before trying to make extensive customisations to an existing command.
- If you do end up creating many customisations your preamble may become rather long; it is relatively simple to bundle them all up into your own package which can then be invoked with `\usepackage{mypackage}`.

Why use a LaTeX slide-making environment?

. . . when there's Microsoft Powerpoint?

- It is straightforward to include mathematics.
- It is possible to link to bibliographies.
- Customisation is possible, so that the environment can be tailored to suit your needs.

So the same considerations apply. If you have chosen to use LaTeX to write your thesis then you will probably need to be able to create presentations in LaTeX too.

Creating slideshows

There are several slide-making document classes. Originally there were several classes that were suited to creating transparencies and foils.

SliTeX A separate program, written by Leslie Lamport for creating transparencies.

slides Early LaTeX slide-making document class. It is not widely used now as it lacks many more complex features. It was good for creating transparencies; less so for online presentations.

seminar and foils Originally developed to produce acetate foils, but can produce output suitable for an overhead projector. Not much used.

Powerpoint changed everything. The expectation was that a presentation, as well as containing useful information would also be colourful, include dynamic effects such as animations and pausing between bullet points.

Current slide-making classes

Several new slide-making environments were produced in response.

prosper Based on seminar and includes the ability to produce dynamic effects. Now superceded largely by powerdot.

beamer Relatively powerful and easy to learn; creating dynamic effects is relatively straightforward.

talk Again, easy to learn. It doesn't impose a particular slide-style on you.

We will be using Beamer today. Early distributions of LaTeX did not always include Beamer; but if you are using an up-to-date version then it should be available. A couple of additional packages such as *pgfpages* are also needed.

Beamer will produce output in PDF format which makes it very portable. PDF format presentations will "just work" on most systems.

What Beamer provides

Beamer extensions

There are several additional environments and commands that are specific to Beamer.

- New environments include *block*, *column* and *animate*.
- There are also several mathematical environments such as *theorem*, *proof* and *definitions*.
- Transitions, pauses and overlays are easily managed.

What Beamer provides

Beamer extensions

There are several additional environments and commands that are specific to Beamer.

- New environments include *block*, *column* and *animate*.
- There are also several mathematical environments such as *theorem*, *proof* and *definitions*.
- Transitions, pauses and overlays are easily managed.
- That was a pause.

What Beamer provides

Beamer extensions

There are several additional environments and commands that are specific to Beamer.

- New environments include *block*, *column* and *animate*.
- There are also several mathematical environments such as *theorem*, *proof* and *definitions*.
- Transitions, pauses and overlays are easily managed.
- That was a pause.

beamercolorbox allows you to change both the background and foreground colour of a part of a slide.

A simple beamer file

```
\documentclass[pdf]{beamer}
\usetheme{Warsaw}
\begin{document}
\title{My first slideshow}
\subtitle{I hope you like it}
\author{Susan Hutchinson}
\institute{University of Oxford}
\date
\maketitle
\tableofcontents
\section{Introduction}
\frame{My first slide}
\end{document}
```

The components of a slide

1. a headline and a footline
2. a left and right sidebar
3. navigation bars
4. navigation symbols
5. a logo
6. a frametitle
7. a background
8. some frame contents

Not all slides have all these components. The first three are usually set up by the theme you choose. The contents are your problem!

Beamer themes and colours

The appearance of your slides depends on the theme you have chosen. There is a smaller range than with powerpoint but there is much scope for customisation. To change the theme and colour use

```
\usetheme{PaloAlto}
\usecolortheme{albatross}
```

Some popular themes are

AnnArbor	Berkeley	Berlin	Boadilla
CambridgeUS	Copenhagen	Darmstadt	Frankfurt
Hanover	Luebeck	Marburg	PaloAlto
Pittsburgh	Rochester	Singapore	Warsaw

Some popular colours are

albatross	beaver	beetle	crane	dolphin	dove
fly	orchid	rose	seagull	seahorse	wolverine

The layout of a slide

Each slide has the format

```
\begin{frame}
\frametitle{}

The contents of the slide go here.

\end{frame}
```

The contents of the slide can include LaTeX commands, pictures, tables and so on.

Beamer restrictions

- The depths of *itemize*,*enumerate* and *description* environments are limited.
- Pictures and figures need careful handling.
- Using *bibtex* is rather fiddly.

The title page

Add the following to the preamble

```
\title[Short title]{My long title}
\subtitle[Short subtitle]{My long subtitle}
\author{My Name}
\date{November 2007}
\institute{My University}
```

and then include

```
\begin{frame}
\frametitle{Outline}
\maketitle
\end{frame}
```

after \begin{document}. A short version of the text has been included between [and] which will appear at the foot of each slide.

Effects can be included such a pauses and overlays. For example to pause between items like this:

- Mount Everest grows by 1cm a year

Effects can be included such a pauses and overlays. For example to pause between items like this:

- Mount Everest grows by 1cm a year
- A new planet is discovered every day.

Effects can be included such a pauses and overlays. For example to pause between items like this:

- Mount Everest grows by 1cm a year
- A new planet is discovered every day.

use

```
\begin{itemize}
\item Mount Everest grows by 1cm a year
\pause
\item A new planet is discovered every day.
\end{itemize}
\pause
```

Creating overlays

Overlays allow you to determine in what order items appear. For example

Theorem

There is no largest prime number.

Proof.

1. Suppose p where the largest prime number.

4. Thus $q + 1$ is also prime and greater than p. □

Creating overlays

Overlays allow you to determine in what order items appear. For example

Theorem

There is no largest prime number.

Proof.

1. Suppose p where the largest prime number.
2. Let q be the product of the first p numbers.

4. Thus $q + 1$ is also prime and greater than p. □

Creating overlays

Overlays allow you to determine in what order items appear. For example

Theorem

There is no largest prime number.

Proof.

1. Suppose p where the largest prime number.
2. Let q be the product of the first p numbers.
3. Then $q + 1$ is not divisible by any of them.
4. Thus $q + 1$ is also prime and greater than p. □

Creating overlays

Overlays allow you to determine in what order items appear. For example

Theorem

There is no largest prime number.

Proof.

1. Suppose p where the largest prime number.
2. Let q be the product of the first p numbers.
3. Then $q + 1$ is not divisible by any of them.
4. Thus $q + 1$ is also prime and greater than p. □

Proved using *reduction ad absurdum*.

The markup for overlays

```
\begin{theorem}
There is no largest prime number.
\end{theorem}
\begin{proof}
\begin{enumerate}
\item<1-> Suppose $p$ where the largest prime number.
\item<2-> Let $q$ be the product of the first $p$ numbers.
\item<3-> Then $q + 1$ is not divisible by any of them.
\item<1-> Thus $q + 1$ is also prime and greater than
 $p$. \qedhere
\end{enumerate}
\end{proof}
\uncover<4->{Proved using \textit{reduction ad absurdum}.}
```

Note the use of <1->, <2-> to determine the order in which information is revealed.

Adding some structure

The \section and \subsection commands are used to add structure to the slides. These are used outside frames. They can contain a long and short version. The long version appears in the table of contents, the short version in the header line.

```
\section[Slide creation]{Creating slides using Beamer}
```

Add a slide that contains

```
\begin{frame}
\frametitle{Outline}
\tableofcontents
\end{frame}
```

and a slide which includes all the section and subsections headings will be generated.
Finally navigation symbols can be added which allow you to find your way around the presentation when it is being given.

Exercises

Now do the following exercises.

- Creating and customising commands
- Make your own slide show

What are packages

Packages are used to alter or add to basic LaTeX behaviour. For example to change the way paragraphs are separated add \usepackage{parskip} to the preamble.

```
\documentclass{article}
\usepackage{parskip}
\begin{document}
\section{Introduction}
Hello there. This is the first paragraph.

Goodbye now. That's it.
\end{document}
```

More about packages

Finding and configuring packages to do what you want is a key skill for LaTeX users.

- Packages extend existing functions.
- Packages add extra functions.
- There are hundreds of packages, many of which will be installed with your LaTeX distribution.

Some commonly-used packages

amsmath,amssymb	Additional mathematical characters
fancyhdr	Extends headers and footers on the page
graphicx	More configuratable picture environment
longtable	Allows tables to extend over more than one page
lscape	Change the orientation of a page
natbib	Add a bibliography
tocloft	Changes Table of Contents format

Some more commonly used packages

babel	Allows you to choose the language of key words.
color, xcolor	Change the colour of text and other features.
fancyvrb	Enhances the verbatim environment. Useful if you need to make text appear exactly as it is typed.
glossaries	Create a glossary of key words.
hyperref	Make your references, tables of contents, lists of figures and tables links.

Learning a new package

Once you have found the package you want, you will need to understand how it works.

- Read the documentation. Most packages have a guide which can be read with the `texdoc` command if you are using a Unix/Linux system.
- Use Google.
- Create a small working example to help you become familiar with the features.

Solving your own problems

These slides and exercises are designed to get you started. At some point you will hit a problem (I do all the time) which you can't solve immediately.

- Move `\end{document}` further up the file. This may isolate the problem.
- If the error reports a line number make sure you look for errors before that.
- Use Google to search for the error. Or look at a guide.
- Create a minimal example. This helps you narrow the problem down.
- Either subscribe to `comp.text.tex` via a news reader like Thunderbird or use Google groups.

Exercises

Now do the following exercises.

- Exploring packages.
- Tips and tricks to solve problems yourself.

Conclusion

Good luck!

I hope you will all now write beautiful documents.