Writing a Thesis in LaTeX: hints, tips and advice

Nicola Talbot
http://theoval.cmp.uea.ac.uk/~nlct/

Friday 20th October, 2006

1 Structuring Your Document

Structure your document before you start.

- Front Matter
  - Use lowercase Roman numeral page numbering:

    ```
    \pagenumbering{roman}
    ```
  - Title Page (see below)
  - Table of contents, list of figures/tables

    ```
    \tableofcontents
    \listoffigures
    \listoftables
    ```
  - Abstract

    ```
    \begin{abstract}
    \end{abstract}
    ```
  - Acknowledgements

    ```
    \chapter*{Acknowledgements}
    ```
- Main Matter
  - Use Arabic numbers

    ```
    \pagenumbering{arabic}
    ```
  - Chapters/sections
- Back Matter
  - glossary: important to define symbols (e.g. is x' the derivative of x or a new value of x?)
  - bibliography
  - If you have written computer code, don't include all the code you have ever written!

2 Formatting

2.1 Title Page

- Either use `\author{<text>}`, `\title{<text>}` and `\date{<text>}` with `\maketitle`

```
\author{A.N. Other}
\title{A Sample Thesis}
\date{October 2006}
\maketitle
```

- or use titlepage environment

```
\begin{titlepage}
\null\vfill
\begin{center}\Large
A Thesis submitted for the degree of
Doctor of Philosophy\par\vskip1cm
School of Mathematics\par
University of Somewhere\par
\vskip1cm
\large A Sample Thesis \par
\vskip1cm
A.N. Other\par
October 2006
\end{center}\vfill
```

2.2 Double Spacing

Many universities insist on double spacing to provide examiners room for annotations. Use setspace package:

- `\singlespacing`
- `\onehalfspacing`
- `\doublespacing`

2.3 Theorems and Algorithms

`\newtheorem{`*<type>*`}{`*<title>*`}[`*<in-counter>*`]`

- Creates an environment called *<type>*
- Start of environment will have *<title>* and associated number in bold
- If *<in-counter>* is present, associated number will depend on *<in-counter>*
- Body of environment will be in italic
- New environment *<type>* has optional argument to provide a sub-title.

2.4 Verbatim Text

- verbatim or verbatim* environments
- `\verb`*<c><text><c>* or `\verb*`*<c><text><c>*
- verbatim package:
  - `\verbatiminput{`*<filename>*`}`
- moreverb package:
  - verbatimtab environment
  - `\verbatimtabinput{`*<filename>*`}`

  - listing environment
  - `\listinginput{<filename>}`

- Verbatim text can not be included in command arguments!

2.5 Symbols

- LaTeX provides many common symbols
- For additional maths symbols try one of the following packages:
  - amsfonts/amssymb
  - stmaryrd
  - wasysym
  - mathabx
  - txfonts/pxfonts

  Maths symbols need to be in a maths environment.

- For a comprehensive list of all symbols, see "The Comprehensive Symbol List" available on CTAN.
- To negate a symbol use `\not`

  `$\not<$` $\not<$

- For a degree symbol use `^\circ`

  `$45^\circ$` 45°

- For calligraphic fonts use `\mathcal{<text>}`

  `$\mathcal{S}$` $\mathcal{S}$

3 Results Chapter

- Results chapters often cause problems where there are a large number of figures and tables
- All figures and tables must have explanatory text
- Always given LaTeX some choice as to where to position the floats

  ✗ `\begin{figure}[h]`

  ✓ `\begin{figure}[htbp]`

- If you absolutely and emphatically want a float to go "right here" it's not a float!
- Use `\clearpage` if you get the `Too many unprocessed floats` error.
- Captions are produced with `\caption{<text>}`
- Labels should go *after* the caption
- Caption styles can be changed using:
  - caption package
  - ccaption package
  - float package
  - KOMA-Script classes
  - memoir class

3.1 Tables

- Less than a page use table environment
- More than a page use
  - longtable environment (longtable package)
  - supertabular environment (supertab package)
- Captions should go at the top of the table
- Table contents (arranging material in rows and columns):
  - Use tabular environment inside table environment `\begin{tabular}{<`*format*`>}`
  - Argument specifies format of each column: `l` (left justified), `c` (centred), `r` (right justified) and `p{<`*width*`>}` (formatted paragraph of given width.)
  - Within tabular environment use `&` to move to next column, use `\\` to move to next row.
  - Horizontal lines: `\hline` (over all columns) and `\cline{<`*n*`>-<`*m*`>}` (over columns <*n*> to <*m*> inclusive.)
  - Vertical lines added using `|` (bar character) in <*format*>
  - Spanning columns: `\multicolumn{<`*n*`>}{<`*align*`>}{<`*text*`>}`
  - Spanning Rows (multirow package): `\multirow{<`*n*`>}{<`*col-width*`>}{<`*text*`>}`

3.2 Figures

- Use figure environment
- Caption should go at the bottom
- Figure contents can either be created internally or externally.
- Internally Created Images
  - Construct image in the document using commands and environments.
    - picture environment (primitive but portable)
    - pstricks package (very powerful—uses PostScript)
    - pdftricks generates PDF files from pstricks (fiddly)
    - pgf package works with LaTeX+dvips and PDFLaTeX. Has user interface tikz to make it easier to use.
    - Search CTAN!
- Externally Created Images
  - Use external application to create image.
  - Save image as:
    - LaTeX code (use `\input{<`*filename*`>}`)
    - an image format known to LaTeX, e.g. EPS, PDF, PNG (use `\includegraphics{<`*filename*`>}` in graphicx package)

3.2.1 LaTeX code v Image Formats

- LaTeX code:
  - Text in images will use same font as document.
  - Images can include well formatted equations.
  - The LaTeX code can be edited to fine-tune image.
  - LaTeX code can only produce vector graphics.
  - You may need a particular driver to understand the code
- Image format:
  - Text in images may not match document font.
  - Image files can either be vector or raster graphics:
    - * If possible save as vector graphics (e.g. EPS, PDF).
    - * Raster images don't scale well.
  - Driver needs to understand image format, e.g.:
    - * EPS : `latex` + `dvips`
    - * PDF : `pdflatex`

3.2.2 Transformations (graphicx package)

File	`myImg.tex`	`myImg.eps`
No transform	`\input{myImg}`	`\includegraphics{myImg}`
Magnify ×2	`\scalebox{2}{\input{myImg}}`	`\includegraphics[scale=2]{myImg}`
Scale so that width=3in	`\resizebox{3in}{!}{\input{myImg}}`	`\includegraphics[width=3in]{myImg}`
Rotate by 45°	`\rotatebox{45}{\input{myImg}}`	`\includegraphics[angle=45]{myImg}`

3.3 External Datafiles

- You may have data stored in external files, e.g. results from experiments
- Data can be included in your thesis:
  - Directly using, e.g., csvtools package (ASCII)
  - Using an external application:
    - * exceltex : package combined with Perl script
    - * Excel-to-LaTeX : converts Excel to LaTeX tables
    - * xl2latex : converts Excel to LaTeX tabulars
    - * Calc2LaTeX : converts OpenOffice to LaTeX tables
    - * PstChart : generates various charts (pstricks code)
- Common ASCII formats:
  - Comma Separated Variables (`.csv`)

    ```
    Name,Quantity
    "Apples",20
    ```

  - Tab Separated Variables (`.txt`)

    ```
    Name    Quantity
    "Apples"        20
    ```

3.3.1 Using the csvtools Package

- Assumes comma separated variable files. If you are using tab separated files use `\setcsvseparator{^^I}`
- Header row must be on line 1
- Commands provided by csvtools iterate through each line of data (not including header row.) To access entry in a given column of the current row use:
  - `\field{<n>}`
  - `\insertbyname{<header>}`
  - `\insert<header>`

  Where <*n*> is the column number and <*header*> is the header text for that column.
- Example: given header row:
  `Name,Quantity`
  To access elements in 1st column: `\field{1}` or `\insertName` or `\insertbyname{Name}`
- To convert data to tabular environment:
  `\CSVtotabular{<file>}{<align>}{<header>}{<all but last>}{<last>}`
- To convert data to longtable environment:
  `\CSVtolongtable{<file>}{<align>}{<header>}{<all but last>}{<last>}`
- Where:

<*file*>	: name of data file (e.g. `sample.csv`)
<*align*>	: column specifiers (e.g. `|l|r|`)
<*header*>	: code for header row (data not accessed in this row)
<*all but last*>	: code for all but last row of data
<*last*>	: code for last row of data

- Example:

```
\CSVtotabular{sample.csv}{|l|r|}
{\hline\bfseries Name & \bfseries Quantity\\\hline\hline}
{\insertName & \insertQuantity\\}{\insertName & \insertQuantity\\\hline}
```

- Applying same code to each row of data: `\applyCSVfile{<filename>}{<code>}`
- Example:
  - You have a CSV file (`results.csv`) containing the name of an image displaying the result of a given experiment:

```
Experiment,File
abc,abcResults.eps
xyz,xyzResults.eps
```

    (Lots more lines omitted!)

  - You want to include each image file in a separate figure:

```
\applyCSVfile{results.csv}{%
\begin{figure}[htbp]
\begin{center}
\includegraphics{\insertFile}
\end{center}
\caption{Results from Experiment \insertExperiment}
\label{fig:exp\insertExperiment}
\end{figure}}
```

3.3.2 Using the csvpie Package

- `\csvpiechart[`*<options>*`]{`*<variable>*`}{`*<filename>*`}`
- Creates a simple circular pie chart
- Segments can be separated from the chart
- "Inner" and "Outer" labelling
- Labelling format can be customised
- Segment colours can be customised
- Can read in decimal numbers from CSV file, but rounding will occur (TEX only performs integer arithmetic.)
- Uses tikz package
- *<options>* must be a comma-separated list of *<key>*=*<value>* pairs. Common keys:

Key	Value	Default	Description
`start`	*<number>*	0	The start angle of the first segment
`total`	*<number>*	100	The sum of all the segment values
`radius`	*<length>*	2cm	The radius of the pie chart
`cutway`	*<list>*		List or range of segments to separate from the pie chart

- Example: `\csvpiechart[cutaway={1-2}]{\field{2}}{sample.csv}`

4 Creating Glossaries

- gloss (Glossaries - uses BibTeX)
- Packages that use Makeindex:
  - glossary (Glossaries, Acronyms)
  - glosstex (Glossaries, Acronyms, General sorted lists)
  - nomencl (List of symbols)

4.1 The glossary package

- In preamble:
  - `\makeglossary`
  - `\storeglosentry{`*<label>*`}{`*<entry>*`}`
    <entry> is a *<key>*=*<value>* list

Key	Value
`name`	the entry name/term/symbol
`description`	a description of the entry
`sort`	how to sort the entry
`format`	how to format the entry page number

- In document:
  - `\gls{`*<label>*`}`
  - `\useGlosentry{`*<label>*`}{`*<text>*`}`
- Where you want the glossary to appear: `\printglossary`
- Save your document (say, `myDoc.tex`)

- Either

  ```
  latex myDoc
  makeindex -s myDoc.ist -o myDoc.gls myDoc.glo
  latex myDoc
  ```

- Or:

  ```
  latex myDoc
  makeglos myDoc
  latex myDoc
  ```

- Example:
  - Defining the entry:

    ```
    \storeglosentry{deriv}{name={$f'(x)$},
    description={The derivative of $f$},sort={f'}}
    ```

  - Using the entry:

    ```
    An entry about \gls{deriv}.
    ```

- Caveat: the characters `|"!@` are `makeindex` special characters. Example:
  - Defining the entry:

    ```
    \storeglosentry{mod}{name={$"|x"|$},
    description={modulus of $x$},sort={modulus}}
    ```

  - Using the entry:

    ```
    An entry about \useGlosentry{mod}{$|x|$}.
    ```

- Acronyms:
  - `\usepackage[acronym]{glossary}`
  - Preamble: `\makeacronym`
  - Define acronym: `\newacronym{`<*acronym*>`}{`<*long*>`}{`<*glos-entry*>`}`
  - Where you want the list of acronyms: `\printacronym`
  - Either:

    ```
    makeindex -s myDoc.ist -o myDoc.acn myDoc.acr
    ```

  - Or:

    ```
    makeglos myDoc
    ```

  - Example:
    - ∗ Defining an acronym:

      ```
      \newacronym{svm}{support vector machine}{%
      description={Statistical pattern recognition technique}}
      ```

    - ∗ Using the acronym:

      ```
      This method uses a \svm.
      ```

    - ∗ alternatively:

      ```
      This method uses a \useacronym{svm}.
      ```

FAQs are available for the glossary and csvtools packages at http://theoval.cmp.uea.ac.uk/~nlct/latex/packages/faq/