Writing a Thesis in LaTeX: hints, tips and advice

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

School of Computing Sciences
University of East Anglia

Overview

Introductory Notes

- There is generally more than one way of doing things
- I will describe the method I know best
- I will also mention alternatives, but will not describe them
- Look up the documentation on CTAN (`http://www.tex.ac.uk/`)

Before You Start

- Decide on an appropriate class file.
  - Ask your supervisor if one is provided
  - If not, try the `report` or `scrreprt` class file
- Structure your document:
  - Front Matter
  - Main Matter
  - Back matter

Front Matter

- Use lowercase Roman numeral page numbering

  `\pagenumbering{roman}`

Front Matter

- Use lowercase Roman numeral page numbering

  `\pagenumbering{roman}`
- Title page

Front Matter

- Use lowercase Roman numeral page numbering

  `\pagenumbering{roman}`
- Title page
- Table of contents, list of figures/tables

  `\tableofcontents \listoffigures \listoftables`

Front Matter

- Use lowercase Roman numeral page numbering

  `\pagenumbering{roman}`
- Title page
- Table of contents, list of figures/tables

  `\tableofcontents \listoffigures \listoftables`
- Abstract should go in `abstract` environment

  `\begin{abstract} \end{abstract}`

Front Matter

- Use lowercase Roman numeral page numbering

  `\pagenumbering{roman}`
- Title page
- Table of contents, list of figures/tables

  `\tableofcontents \listoffigures \listoftables`
- Abstract should go in `abstract` environment

  `\begin{abstract} \end{abstract}`
- Acknowledgements (funding body etc)

  `\chapter*{Acknowledgements}`

  (You may be told to put acknowledgements in back matter)

Main Matter

- Use Arabic numbers

  `\pagenumbering{arabic}`

Main Matter

- Use Arabic numbers

```
\pagenumbering{arabic}
```

- Chapters, sections etc. (check with your supervisor)

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

\chapter{Technical Introduction} \label{ch:techintro}

\chapter{Method} \label{ch:method}

\chapter{Results} \label{ch:results}

\chapter{Conclusions} \label{ch:conc}
```

Back Matter

- Glossary of terms or notation. (You may be told to put this in the front matter)
  - Important to define symbols (e.g. is x' the derivative of x or a new value of x?)
  - Include a list of acronyms, especially newly defined acronyms.

Back Matter

- Glossary of terms or notation. (You may be told to put this in the front matter)
  - Important to define symbols (e.g. is x' the derivative of x or a new value of x?)
  - Include a list of acronyms, especially newly defined acronyms.
- Bibliography

Back Matter

- Glossary of terms or notation. (You may be told to put this in the front matter)
  - Important to define symbols (e.g. is x' the derivative of x or a new value of x?)
  - Include a list of acronyms, especially newly defined acronyms.
- Bibliography
- If you have written computer code, don't include all the code you've ever written!
  - Examiners will view it as padding
  - Don't annoy your examiners!

Formatting

- Title Page
- Double Spacing
- Theorems and Algorithms
- Verbatim Text
- Symbols

Creating a Title Page

- Simplest method is to provide title, author and date information with `\maketitle`:

```
\title{A Sample Thesis}
\author{My Name}
\date{October 2006}
\maketitle
```

- Some class files and packages provide additional commands:
  - `scrreprt` Class File
  - `titling` Package
- Alternatively use the `titlepage` environment

The titlepage Environment

Example:

```
\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
Me \par
October 2006
\end{center}\vfill
\end{titlepage}
```

Double Spacing

- Many universities insist on double spacing to provide examiners room for annotations
- Use `setspace` package:
  - `\singlespacing`
  - `\onehalfspacing`
  - `\doublespacing`

Theorems and Algorithms

- Use `\newtheorem`
- To modify the default style:
  - `amsthm` (`amsmath`)
  - `empheq` (Extension to `amsmath`)
  - `ntheorem`
  - `nccthm`
  - `algorithmicx`
- If you want theorems/algorithms as a float:
  - `alg`
  - `algorithm2e`
  - `algorithms`
  - `float`

Defining Theorem-Type Structures

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

Defining Theorem-Type Structures

- `\newtheorem{`<*type*>`}{`<*title*>`}[`<*in-counter*>`]`
- Creates an environment called <*type*>

Defining Theorem-Type Structures

- `\newtheorem{`<*type*>`}{`<*title*>`}[`<*in-counter*>`]`
- Creates an environment called <*type*>
- The start of the environment will have <*title*> and the associated number in bold.

Defining Theorem-Type Structures

- `\newtheorem{`<*type*>`}{`<*title*>`}[`<*in-counter*>`]`
- Creates an environment called <*type*>
- The start of the environment will have <*title*> and the associated number in bold.
- The counter can be associated with another counter using <*in-counter*>

Defining Theorem-Type Structures

- `\newtheorem{`<*type*>`}{`<*title*>`}[`<*in-counter*>`]`
- Creates an environment called <*type*>
- The start of the environment will have <*title*> and the associated number in bold.
- The counter can be associated with another counter using <*in-counter*>
- The body of the environment will be in italic font

Defining Theorem-Type Structures

- `\newtheorem{`<*type*>`}{`<*title*>`}[`<*in-counter*>`]`
- Creates an environment called <*type*>
- The start of the environment will have <*title*> and the associated number in bold.
- The counter can be associated with another counter using <*in-counter*>
- The body of the environment will be in italic font
- The new environment <*type*> has an optional argument to provide a sub-title for the theorem

Examples

```
\newtheorem{theorem}{Theorem}
```

Examples

```
\newtheorem{theorem}{Theorem}
```

1. `\begin{theorem} If a real sequence is bounded and monotone, it converges.\end{theorem}`

   Theorem 1 *If a real sequence is bounded and monotone, it converges.*

Examples

```
\newtheorem{theorem}{Theorem}
```

1. `\begin{theorem} If a real sequence is bounded and monotone, it converges.\end{theorem}`

   Theorem 1 *If a real sequence is bounded and monotone, it converges.*

2. `\begin{theorem}[Cayley's Theorem] Every group is isomorphic to a group of permutations. \end{theorem}`

   Theorem 2 (Cayley's Theorem) *Every group is isomorphic to a group of permutations.*

Verbatim Text

- `\verb`<*c*><*text*><*c*> command (in line verbatim)
- `\verb*`<*c*><*text*><*c*> command (in line verbatim)

Verbatim Text

- `\verb`<*c*><*text*><*c*> command (in line verbatim)
- `\verb*`<*c*><*text*><*c*> command (in line verbatim)
- `verbatim` environment (displayed verbatim)
- `verbatim*` environment (displayed verbatim)

Verbatim Text

- `\verb<c><text><c>` command (in line verbatim)
- `\verb*<c><text><c>` command (in line verbatim)
- `verbatim` environment (displayed verbatim)
- `verbatim*` environment (displayed verbatim)
- `verbatim` package:
  - `\verbatiminput{<filename>}`

Verbatim Text

- `\verb<`*c*`><`*text*`><`*c*`>` command (in line verbatim)
- `\verb*<`*c*`><`*text*`><`*c*`>` command (in line verbatim)
- `verbatim` environment (displayed verbatim)
- `verbatim*` environment (displayed verbatim)
- `verbatim` package:
  - `\verbatiminput{<`*filename*`>}`
- `moreverb` package:
  - `verbatimtab` environment
  - `\verbatimtabinput{<`*filename*`>}`
  - `listing` environment
  - `\listinginput{<`*filename*`>}`
- Verbatim text can not be include in command arguments!

Examples (verbatim environment)

1.

```
\begin{verbatim}
Some  %$& \large odd
text.
\end{verbatim}
```

```
Some  %$& \large odd
text.
```

2.

```
\begin{verbatim*}
Some  %$& \large odd
text.
\end{verbatim*}
```

```
Some␣␣%$&␣\large␣odd
text.
```

Examples (\verb command)

1. `\verb"some %$& text"` → `some %$& text`
2. `\verb+some %$& text+` → `some %$& text`
3. `\verb|some %$& text|` → `some %$& text`
4. `\verb*|some %$& text|` → `some␣%$&␣text`

Symbols

- LaTeX provides many common symbols
- Packages:
  - `amsfonts/amssymb`
  - `stmaryrd`
  - `wasysym`
  - `mathabx`
  - `txfonts/pxfonts`
  - Many more! See "The Comprehensive Symbol List" available on CTAN.
- Most maths symbols can only be used in a maths environment.

Commonly Used Maths Symbols

`\le`	$\le$	`\ge`	$\ge$	`\ll`	$\ll$	`\gg`	$\gg$
`\neq`	$\neq$	`\equiv`	$\equiv$	`\sim`	$\sim$	`\approx`	$\approx$
`\in`	$\in$	`\notin`	$\notin$	`\ni`	$\ni$	`\empyset`	$\emptyset$
`\forall`	$\forall$	`\exists`	$\exists$	`\partial`	∂	`\\|`	$\|$

Commonly Used Maths Symbols

`\le`	$\le$	`\ge`	$\ge$	`\ll`	$\ll$	`\gg`	$\gg$
`\neq`	$\neq$	`\equiv`	$\equiv$	`\sim`	$\sim$	`\approx`	$\approx$
`\in`	$\in$	`\notin`	$\notin$	`\ni`	$\ni$	`\empyset`	$\emptyset$
`\forall`	$\forall$	`\exists`	$\exists$	`\partial`	∂	`\\|`	$\|$

- To negate a symbol use `\not`, e.g.:

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

Commonly Used Maths Symbols

`\le`	$\le$	`\ge`	$\ge$	`\ll`	$\ll$	`\gg`	$\gg$
`\neq`	$\neq$	`\equiv`	$\equiv$	`\sim`	$\sim$	`\approx`	$\approx$
`\in`	$\in$	`\notin`	$\notin$	`\ni`	$\ni$	`\empyset`	$\emptyset$
`\forall`	$\forall$	`\exists`	$\exists$	`\partial`	∂	`\\|`	$\|$

- To negate a symbol use `\not`, e.g.:

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

- For a degree symbol use `^\circ`, e.g.:

  `$45^\circ$` 45°

Commonly Used Maths Symbols

`\le`	$\le$	`\ge`	$\ge$	`\ll`	$\ll$	`\gg`	$\gg$
`\neq`	$\neq$	`\equiv`	$\equiv$	`\sim`	$\sim$	`\approx`	$\approx$
`\in`	$\in$	`\notin`	$\notin$	`\ni`	$\ni$	`\empyset`	$\emptyset$
`\forall`	$\forall$	`\exists`	$\exists$	`\partial`	∂	`\\|`	$\|$

- To negate a symbol use `\not`, e.g.:

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

- For a degree symbol use `^\circ`, e.g.:

  `$45^\circ$` 45°

- For calligraphic fonts use `\mathcal{`*text*`}`, e.g.:

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

Results Chapter

- Results chapters often cause problems because of the large number of figures and tables

Results Chapter

- Results chapters often cause problems because of the large number of figures and tables
- All figures and tables must have explanatory text

Results Chapter

- Results chapters often cause problems because of the large number of figures and tables
- All figures and tables must have explanatory text
- Always give LaTeX some choice as to where to position the floats
  - ✘ `\begin{figure}[h]`
  - ✔ `\begin{figure}[htbp]`

Results Chapter

- Results chapters often cause problems because of the large number of figures and tables
- All figures and tables must have explanatory text
- Always give 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!

Results Chapter

- Results chapters often cause problems because of the large number of figures and tables
- All figures and tables must have explanatory text
- Always give 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!
- As a last resort use `\clearpage` if you get the error:
  `Too many unprocessed floats`

Captions

- Captions are produced with:
  `\caption[`<*lof caption*>`]{`<*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

Tables

- Less than a page
  - `table` environment

Tables

- Less than a page
  - `table` environment
- More than a page
  - `longtable` environment (`longtable` package)
  - `supertabular` environment (`supertab` package)

Tables

- Less than a page
  - `table` environment
- More than a page
  - `longtable` environment (`longtable` package)
  - `supertabular` environment (`supertab` package)
- Captions should go at the top of the table

```
\begin{table}[htbp]
\caption{A Sample Table}\label{tab:sample}
\begin{center}
% table contents
\end{center}
\end{table}
```

>>

Table Contents : The `tabular` environment

- Use `tabular` environment to arrange material in rows and columns.

  `\begin{tabular}{`<*format*>`}`
- Argument specifies the format of each column:
  - `l` : left justified
  - `c` : centred
  - `r` : right justified
  - `p{`<*width*>`}` : formatted paragraph of given width
- Within `tabular` environment:
  - Use & to move to next column
  - Use \\ to move to next row

Table Example

```
\begin{table}[htbp]
\caption{A Sample Table}\label{tab:sample}
\begin{center}
\begin{tabular}{clrlr}
Dataset & MSE1 & Time1 (s) & MSE2 & Time2 (s)\\
Benchmark1 & 0.001 & 5 & 0.02 & 8\\
Benchmark2 & 0.035 & 10 & 0.0005 & 15
\end{tabular}
\end{center}
\end{table}
```

Table Example

Table 1: A Sample Table

Dataset	MSE1	Time1 (s)	MSE2	Time2 (s)
Benchmark1	0.001	5	0.02	8
Benchmark2	0.035	10	0.0005	15

Tabular Entries

- Each entry in a `tabular` environment is in an implicit group
- Declarations are localised
- Example:

  ```
  \bfseries Dataset & MSE1 & Time1 (s)
  ```

  Only `Dataset` will be in bold

Table Example

```
\begin{table}[htbp]
\caption{A Sample Table}\label{tab:sample}
\begin{center}
\begin{tabular}{clrlr}
\bfseries Dataset &
\bfseries MSE1 & \bfseries Time1 (s) &
\bfseries MSE2 & \bfseries Time2 (s)\\
Benchmark1 & 0.001 & 5 & 0.02 & 8\\
Benchmark2 & 0.035 & 10 & 0.0005 & 15
\end{tabular}
\end{center}
\end{table}
```

Table Example

Table 2: A Sample Table

Dataset	**MSE1**	**Time1 (s)**	**MSE2**	**Time2 (s)**
Benchmark1	0.001	5	0.02	8
Benchmark2	0.035	10	0.0005	15

Adding Lines

- Vertical lines added using | in placement specifier

  `\begin{tabular}{|c|lr|lr|}`
- Horizontal lines added at the start of the row using:
  - `\hline` : span all columns
  - `\cline{`<*n*>`-`<*m*>`}` : span columns <*n*> to <*m*>

  `\hline Benchmark1 & 0.001 & 5 & 0.02 & 8\\`
- Double lines added using `\hline\hline`:

  `\hline\hline Benchmark1 & 0.001 & 5 & 0.02 & 8\\`

Table Example

```
\begin{table}[htbp]
\caption{A Sample Table}\label{tab:sample}
\begin{center}
\begin{tabular}{|c|lr|lr|}
\hline \bfseries Dataset &
\bfseries MSE1 & \bfseries Time1 (s) &
\bfseries MSE2 & \bfseries Time2 (s)\\
\hline\hline Benchmark1 & 0.001 & 5 & 0.02 & 8\\
Benchmark2 & 0.035 & 10 & 0.0005 & 15\\
\hline
\end{tabular}
\end{center}
\end{table}
```

Table Example

Table 3: A Sample Table

Dataset	**MSE1**	**Time1 (s)**	**MSE2**	**Time2 (s)**
Benchmark1	0.001	5	0.02	8
Benchmark2	0.035	10	0.0005	15

Spanning Multiple Columns/Rows

- Spanning Columns:

  \multicolumn{<*n*>}{<*align*>}{<*text*>}
  - <*n*> number of columns to span
  - <*align*> alignment
  - <*text*> entry text

Spanning Multiple Columns/Rows

- Spanning Columns:

  `\multicolumn{`<*n*>`}{`<*align*>`}{`<*text*>`}`
  - <*n*> number of columns to span
  - <*align*> alignment
  - <*text*> entry text
- Spanning Rows (`multirow` package):

  `\multirow{`<*n*>`}{`<*width*>`}{`<*text*>`}`
  - <*n*> number of rows to span
  - <*width*> column width
  - <*text*> entry text

Table Example

```
\begin{table}[htbp]
\caption{A Sample Table}\label{tab:sample}
\begin{center}
\begin{tabular}{|l|lr|lr|}\hline
\multirow{2}{0.8in}{Dataset} &
\multicolumn{2}{c|}{Method 1} &
\multicolumn{2}{c|}{Method 2}\\
& MSE & Time (s) & MSE & Time (s)\\\hline\hline
Benchmark1 & 0.001 & 5 & 0.02 & 8\\
Benchmark2 & 0.035 & 10 & 0.0005 & 15\\\hline
\end{tabular}
\end{center}
\end{table}
```

Table Example

Table 4: A Sample Table

Dataset	Method 1		Method 2	
	MSE	Time (s)	MSE	Time (s)
Benchmark1	0.001	5	0.02	8
Benchmark2	0.035	10	0.0005	15

Fine Tuning

- Use `\newlength` and `\settowidth` to calculate widest entry:

```
\newlength{\maxwidth}
\settowidth{\maxwidth}{Benchmark2}
\multirow{2}{\maxwidth}{Dataset}
```

Fine Tuning

- Use `\newlength` and `\settowidth` to calculate widest entry:

```
\newlength{\maxwidth}
\settowidth{\maxwidth}{Benchmark2}
\multirow{2}{\maxwidth}{Dataset}
```

- Use `\hfil` or `\hfill` to shift text over:
  - Centred:

    ```
    \multirow{2}{\maxwidth}{\hfil Dataset}
    ```

  - Right Justified:

    ```
    \multirow{2}{\maxwidth}{\hfill Dataset}
    ```

Table Example

```
\newlength{\maxwidth}\settowidth{\maxwidth}{Benchmark2}
\begin{table}[htbp]
\caption{A Sample Table}\label{tab:sample}
\begin{center}
\begin{tabular}{|l|lr|lr|}\hline
\multirow{2}{\maxwidth}{\hfil Dataset} &
\multicolumn{2}{c|}{Method 1} &
\multicolumn{2}{c|}{Method 2}\\
& MSE & Time (s) & MSE & Time (s)\\\hline\hline
Benchmark1 & 0.001 & 5 & 0.02 & 8\\
Benchmark2 & 0.035 & 10 & 0.0005 & 15\\\hline
\end{tabular}
\end{center}
\end{table}
```

Table Example

Table 5: A Sample Table

Dataset	Method 1		Method 2	
	MSE	Time (s)	MSE	Time (s)
Benchmark1	0.001	5	0.02	8
Benchmark2	0.035	10	0.0005	15

Figures

- Use `figure` environment
- Caption should go at the bottom
- Example:

```
\begin{figure}[htbp]
\begin{center}
% contents of figure go here
\end{center}
\caption{A Sample Figure}
\label{fig:sample}
\end{figure}
```

- Figure contents can either be created internally (in the document) or externally (via another application)

Internally Created Images

- Construct image in the document using commands and environments

Internally Created Images

- Construct image in the document using commands and environments
  - `picture` environment (primitive but portable)

Internally Created Images

- Construct image in the document using commands and environments
  - `picture` environment (primitive but portable)
  - `pstricks` package (very powerful—uses PostScript)

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)

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` works with LaTeX+dvips and PDFLaTeX. Has user interface `tikz` to make it easier to use

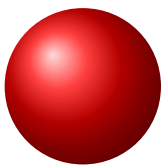
Internally Created Images

- 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` works with LaTeX+dvips and PDFLaTeX. Has user interface `tikz` to make it easier to use
  - ... (search CTAN!)

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` works with LaTeX+dvips and PDFLaTeX. Has user interface `tikz` to make it easier to use
  - ... (search CTAN!)
- Example (using `tikz` package):

```
\tikz \shade[ball color=red] (0,0) circle (5mm);
```



Externally Created Images

- Use external application to create image. Examples:

Application	**Platform**	**Output Format**
Matlab[1]	various	various inc. EPS and PDF
Gnuplot	various	various inc. LaTeX, EPS, PDF
Xfig[2]	Unix	various inc. LaTeX, EPS, PDF
TeXCAD[3]	PC	LaTeX code
JpgfDraw[4]	JVM[5]	LaTeX code

  Many more that create EPS, PDF, PNG etc—search the web!

- Include Image in Document:
  - `\input` (LaTeX code)
  - `\includegraphics` (Image format)

---

[1] Commercial Software

[2] There is a Java based clone of xfig called jfig

[3] There is also a Unix port called xtexcad

[4] Beta version

[5] Java Virtual Machine

Externally Created Images: LaTeX code v Image Formats

- If you use an application that creates 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

Externally Created Images: LaTeX code v Image Formats

- If you use an application that creates 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
- If you use an application that creates an image file:
  - 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`

Including LaTeX Files

- Use `\input{`*<filename>*`}`.

Including LaTeX Files

- Use `\input{`<*filename*>`}`.
- Examples (image in file `mypicture.tex`):
  - Include image "as is":

    `\input{mypicture}`

Including LaTeX Files

- Use `\input{`*<filename>*`}`.
- Examples (image in file `mypicture.tex`):
  - Include image "as is":

    `\input{mypicture}`
  - Magnify image by factor of 2:

    `\scalebox{2}{\input{mypicture}}`

Including LaTeX Files

- Use `\input{`*<filename>*`}`.
- Examples (image in file `mypicture.tex`):
  - Include image "as is":

    `\input{mypicture}`
  - Magnify image by factor of 2:

    `\scalebox{2}{\input{mypicture}}`
  - Scale image so that its width is 3 inches:

    `\resizebox{3in}{!}{\input{mypicture}}`

Including LaTeX Files

- Use `\input{`*<filename>*`}`.
- Examples (image in file `mypicture.tex`):
  - Include image "as is":

    `\input{mypicture}`
  - Magnify image by factor of 2:

    `\scalebox{2}{\input{mypicture}}`
  - Scale image so that its width is 3 inches:

    `\resizebox{3in}{!}{\input{mypicture}}`
  - Rotate image by 45°:

    `\rotatebox{45}{\input{mypicture}}`

Including LaTeX Files

- Use `\input{`*<filename>*`}`.
- Examples (image in file `mypicture.tex`):
  - Include image "as is":

    `\input{mypicture}`
  - Magnify image by factor of 2:

    `\scalebox{2}{\input{mypicture}}`
  - Scale image so that its width is 3 inches:

    `\resizebox{3in}{!}{\input{mypicture}}`
  - Rotate image by 45°:

    `\rotatebox{45}{\input{mypicture}}`
  - Combination (scale then rotate):

    `\rotatebox{45}{\resizebox{3in}{!}{\input{mypicture}}}`

Including LaTeX Files

- Use `\input{`<*filename*>`}`.
- Examples (image in file `mypicture.tex`):
  - Include image "as is":

    `\input{mypicture}`
  - Magnify image by factor of 2:

    `\scalebox{2}{\input{mypicture}}`
  - Scale image so that its width is 3 inches:

    `\resizebox{3in}{!}{\input{mypicture}}`
  - Rotate image by 45°:

    `\rotatebox{45}{\input{mypicture}}`
  - Combination (scale then rotate):

    `\rotatebox{45}{\resizebox{3in}{!}{\input{mypicture}}}`
- Need `graphicx` package to transform image.

Including Image Format Files

- Use `\includegraphics{`<*filename*>`}` (`graphicx`).

Including Image Format Files

- Use `\includegraphics{`<*filename*>`}` (`graphicx`).
- Examples (image in file `mypicture.eps`):
  - Include image "as is":

    ```
    \includegraphics{mypicture.eps}
    ```

Including Image Format Files

- Use `\includegraphics{`<*filename*>`}` (`graphicx`).
- Examples (image in file `mypicture.eps`):
  - Include image "as is":

    `\includegraphics{mypicture.eps}`
  - Magnify image by factor of 2:

    `\includegraphics[scale=2]{mypicture.pes}`

Including Image Format Files

- Use `\includegraphics{`<*filename*>`}` (`graphicx`).
- Examples (image in file `mypicture.eps`):
  - Include image "as is":

    `\includegraphics{mypicture.eps}`
  - Magnify image by factor of 2:

    `\includegraphics[scale=2]{mypicture.pes}`
  - Scale image so that its width is 3 inches:

    `\includegraphics[width=3in]{mypicture.eps}`

Including Image Format Files

- Use `\includegraphics{`<*filename*>`}` (`graphicx`).
- Examples (image in file `mypicture.eps`):
  - Include image "as is":

    `\includegraphics{mypicture.eps}`
  - Magnify image by factor of 2:

    `\includegraphics[scale=2]{mypicture.pes}`
  - Scale image so that its width is 3 inches:

    `\includegraphics[width=3in]{mypicture.eps}`
  - Rotate image by 45°:

    `\includegraphics[angle=45]{mypicture.eps}`

Including Image Format Files

- Use `\includegraphics{<filename>}` (`graphicx`).
- Examples (image in file `mypicture.eps`):
  - Include image "as is":

    `\includegraphics{mypicture.eps}`
  - Magnify image by factor of 2:

    `\includegraphics[scale=2]{mypicture.pes}`
  - Scale image so that its width is 3 inches:

    `\includegraphics[width=3in]{mypicture.eps}`
  - Rotate image by 45°:

    `\includegraphics[angle=45]{mypicture.eps}`
  - Combination (scale then rotate):

    `\includegraphics[width=3in,angle=45]{mypicture.eps}`

Portable Graphics

People often require both a PS and PDF version of the same document.

Portable Graphics

People often require both a PS and PDF version of the same document.

- Have both EPS and PDF versions of same image (e.g. `mypicture.eps` and `mypicture.pdf`)
- Don't include extension in `\includegraphics`
- LaTeX will include EPS file
- PDFLaTeX will include PDF file
- Examples:
  - `\includegraphics{mypicture}`
  - `\includegraphics[width=3in]{mypicture}`
  - `\includegraphics[width=3in,angle=45]{mypicture}`

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:
  - Tab separated (`.txt`)
  - Comma separated (`.csv`)

Example Data

- Comma Separated Variable (sample.csv):

```
Name,Quantity
"Apples",20
"Pears",15
"lemons,limes",30
"Peaches",25
"Cherries",10
```

- Tab Separated Variable (sample.txt):

```
Name    Quantity
"Apples"        20
"Pears" 15
"lemons,limes"  30
"Peaches"       25
"Cherries"      10
```

Using the csvtools Package (v1.2)

- csvtools assumes a comma separated variable file
- If you are using tab separated files, use

  `\setcsvseparator{^^I}`
- Header row must be on line 1
- To access entry in given column of current row use:
  - `\field{<`*n*`>}`
  - `\insertbyname{<`*header*`>}`
  - `\insert<`*header*`>`

  Where
  - <*n*> is the column number
  - <*header*> is the header text for that column

Using the csvtools Package

- Example file has header row:

  `Name,Quantity`
- To access elements in 1st column:
  - `\field{1}`
  - `\insertName`
  - `\insertbyname{Name}`
- To access elements in 2nd column:
  - `\field{2}`
  - `\insertQuantity`
  - `\insertbyname{Quantity}`

csvtools : Creating Tables from Data Files

- 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)
<*all but last*>	:	code for all but last row of data
<*last*>	:	code for last row of data

\CSVtotabular Example

Using earlier sample.csv data:

```
\begin{table}[htbp]
\caption{My Results}\label{tab:results}
\begin{center}
\CSVtotabular{sample.csv}{|l|r|}
{\hline\bfseries Name & \bfseries Quantity\\\hline\hline}
{\insertName & \insertQuantity\\}
{\insertName & \insertQuantity\\\hline}
\end{center}
\end{table}
```

\CSVtotabular Example

Table 6: My Results

Name	**Quantity**
Apples	20
Pears	15
lemons,limes	30
Peaches	25
Cherries	10

\CSVtotabular Example

```
\newcounter{total}
\begin{table}[htbp]
\caption{My Results}\label{tab:results}
\begin{center}
\CSVtotabular{sample.csv}{|l|r|}
{\hline\bfseries Name & \bfseries Quantity\\\hline\hline}
{\insertName & \insertQuantity
\addtocounter{total}{\insertQuantity}\\}
{\insertName & \insertQuantity
\addtocounter{total}{\insertQuantity}\\\hline\hline
\bfseries Total & \thetotal\\\hline}
\end{center}
\end{table}
```

\CSVtotabular Example

Table 7: My Results

Name	**Quantity**
Apples	20
Pears	15
lemons,limes	30
Peaches	25
Cherries	10
Total	100

Applying Same Code for Each Data Row

- Example:
  - You have a CSV file containing the name of an image file displaying the result of a given experiment.
  - You want to include each image file in a separate figure
  - CSV file looks like:

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

    (Imagine there are a lot more lines!)
- Use `\applyCSVfile{`<*data file*>`}{`<*code*>`}`

\applyCSVfile Example

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

`\applyCSVfile` Example — Determining First and Last References

- Each of the figures has a label constructed from the experiment name: `fig:exp`<*name*>

`\applyCSVfile` Example — Determining First and Last References

- Each of the figures has a label constructed from the experiment name: `fig:exp`<*name*>
- How can you determine the first and last labels so that you can do, e.g.: the results are shown in figures 4.2–4.22?

`\applyCSVfile` Example — Determining First and Last References

- Each of the figures has a label constructed from the experiment name: `fig:exp`<*name*>
- How can you determine the first and last labels so that you can do, e.g.: the results are shown in figures 4.2–4.22?
  1. Look in the CSV file and determine the names of the first and last experiment, and work out the corresponding labels.
     - What happens if you add in a new experiment?
     - What happens if you redo the experiments in a different order?

`\applyCSVfile` Example — Determining First and Last References

- Each of the figures has a label constructed from the experiment name: `fig:exp`<*name*>
- How can you determine the first and last labels so that you can do, e.g.: the results are shown in figures 4.2–4.22?
  1. Look in the CSV file and determine the names of the first and last experiment, and work out the corresponding labels.
     - What happens if you add in a new experiment?
     - What happens if you redo the experiments in a different order?
  2. Get LaTeX to work out the first and last references

\applyCSVfile Example

```
\newcommand{\firstref}{??}
\newcommand{\lastref}{??}
\applyCSVfile{results.csv}{%
\begin{figure}[htbp]
\begin{center}
\includegraphics{\insertFile}
\end{center}
\caption{Results from Experiment \insertExperiment}
\label{fig:exp\insertExperiment}
\end{figure}
\ifthenelse{\value{csvrownumber}=1}
{\xdef\firstref{\ref{fig:exp\insertExperiment}}}{}
\xdef\lastref{\ref{fig:exp\insertExperiment}}}
Results are shown in figures~\firstref--\lastref.
```

Creating Pie Charts with `csvpie`

- `\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

\csvpiechart Options

- Optional argument is a comma-separated list of <*key*>=<*value*> pairs

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

For other options see documentation.

Pie Chart Example

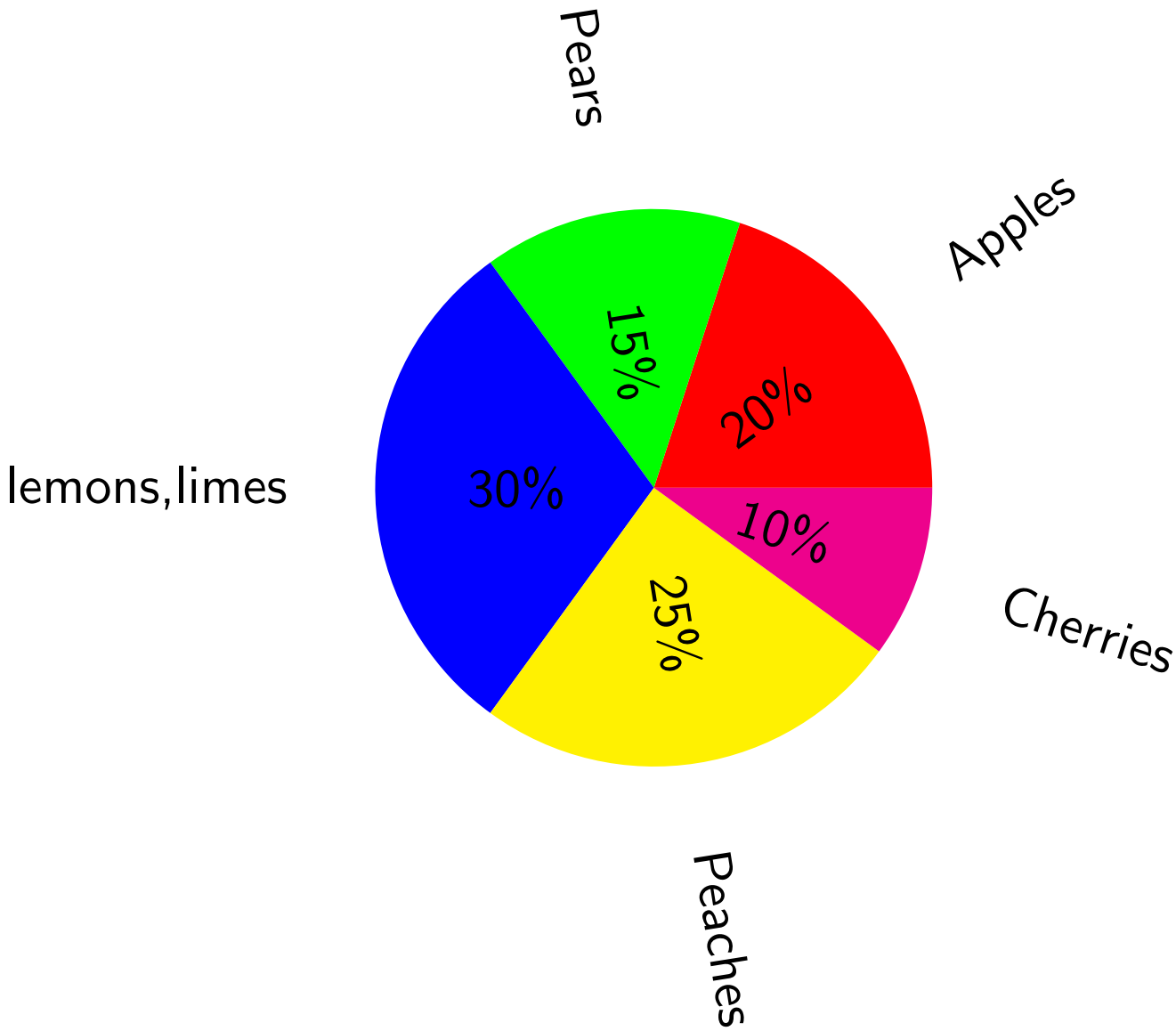
- Use earlier CSV data (`sample.csv`):

```
Name,Quantity
"Apples",20
"Pears",15
"lemons,limes",30
"Peaches",25
"Cherries",10
```

- Using data in second column so <*variable*> is `\field{2}` or `\insertQuantity`
- Second column sums to 100, so don't need `total` option.

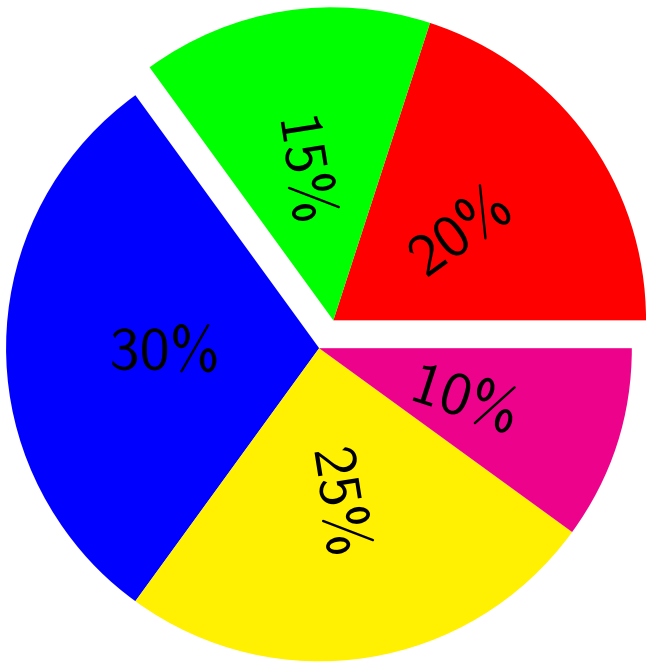
```
\csvpiechart{\field{2}}{sample.csv}
```

Pie Chart Example

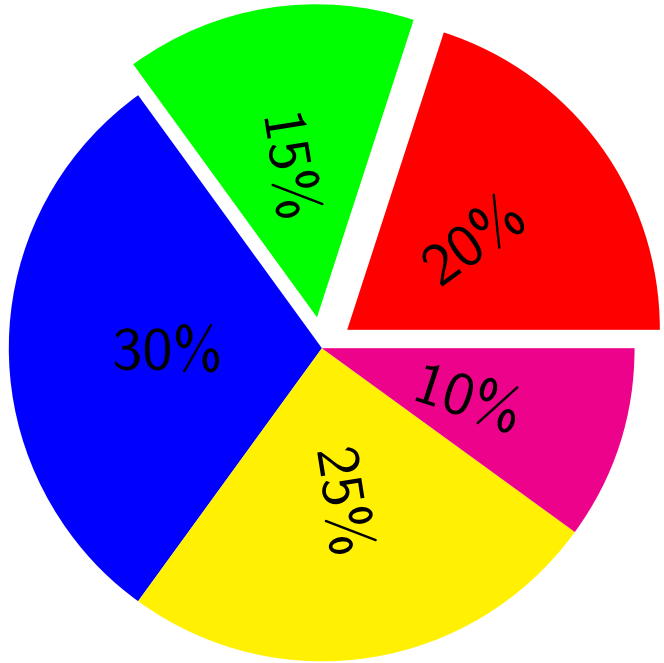


Pie Chart Example

```
\renewcommand{\csvpieouterlabel}{}% remove outer labels
```




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




```
\csvpiechart[cutaway={1,2}]
  {\field{2}}{sample.csv}
```

Creating Glossaries

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

The glossary Package

- In preamble:
  - \makeglossary

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

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*>}

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

Creating a Glossary

- 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
```

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

Examples

1. Use the `sort` key if `name` contains special characters
   - Defining the entry:

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

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

Examples

1. Use the `sort` key if `name` contains special characters
   - Defining the entry:

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

     ```
     An entry about \gls{deriv}.
     ```
2. Dealing with a makeindex special character:
   - 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}.
```

- Make first letter uppercase:

```
\svm* research.
```

  - First use: Support vector machine (svm) research.
  - subsequently: Svm research.

Example

Acronyms with non-alphabetical characters:

- Defining the acronym:

```
\newacronym[ksvm]{k-svm}{kernel support vector
machine}{description={Statistical pattern
recognition technique}}
```

- Using the acronym:

```
This method uses a \ksvm.
```

- alternatively:

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

Recommended Reading

- "A Guide to LaTeX." Helmut Kopka and Patrick W. Daly.
- "The LaTeX Companion" Michel Goossens, Frank Mittelbach and Alexander Samarin
- CTAN's FAQ includes a list of tutorials: `http://www.tex.ac.uk/cgi-bin/texfaq2html?label=tutorials*`