

L^AT_EX Tips and Tricks

UWS CompSci Postgraduates

January 30, 2006

1 Notes

It's useful to be able to annotate your work. [Note: Like this](#) I define two commands for this. This first is for me, and the second one is for Chen to use when I send him drafts [Note: This paragraph is all wrong - Chen](#). The two commands are defined using this section:

```
\newcommand{\note}[1]{\color{blue}\underline{Note: #1}}
\newcommand{\chennote}[1]{\color{red}\underline{Note: #1 - Chen}}
%For final version, uncomment the following lines:
%\renewcommand{\note}[1]{}
%\renewcommand{\chennote}[1]{}

```

The last two lines, when uncommented, instruct L^AT_EX to ignore all of the text highlighted this way, allowing all notes to be quickly removed for a final (non-draft) version. The underlining and colouring of the notes makes it easy to identify them in the typeset output.

2 Listings

A lot of people use the verbatim package for code listings, but this lacks a number of useful features such as automatic line wrapping and syntax highlighting.

The listings package defines a number of environments for typesetting listings. It is also relatively easy to add new languages. The top of this file contains definitions of a slightly nicer way of highlighting XML than the default and a basic set of keywords for typesetting a sigma algebra (for all of the theory people).

The listings package includes a number of features. Snippets of code can be inserted in a paragraph, blocks of code can be inserted such as the one below, and entire files can be included. The last feature is particularly useful

for large code listings - you can include an entire source file (or selected lines from a file) and not have to worry about keeping the original in sync with the document.

Listing 1: An example of lstlistings

```
Algebra Bool is
  sort Bool
  ops TT FF : → Bool
    op and : Bool Bool → Bool
    ...
  var A, B : Bool
  eq and(TT,TT) → TT
  eq and(B,FF) → TT
  eq and(A,B) → and(B,A)
  ...
end Algebra
```

Listing 1 shows an example algebra (a simplified version of the booleans) typeset using the listings package. Note the use of the `literate` option to replace `->` with a right arrow.

3 Makefiles

As with any large project, a \LaTeX document often has many dependencies. Make is an invaluable tool for keeping everything up to date.

Listing 2: An example Makefile

```
Paper.pdf: Paper.tex pdfgraphs figures/Figure1.pdf Paper.bbl
  pdflatex Paper.tex

Paper.bbl: Paper.aux
  bibtex Paper

Paper.aux: Paper.tex
  pdflatex Paper.tex

pdfgraphs:
  cd plots && make

view: Paper.pdf
  open Paper.pdf || xpdf Paper.pdf

clean:
  rm *.aux *.bbl *.dvi *.blg *.log *.out *.pdf > /dev/null 2>&1
  cd plots && make clean

vimclean:
  rm *~ *.bak > /dev/null 2>&1
  cd plots && make vimclean

check:
  aspell -t check Paper.tex
```

Listing 2 shows a simple Makefile that could be used for a paper. It assumes that gnuplot is being used to generate plots using a separate Make-

file in the plots/ subdirectory. Running `make view` will open the output in the default viewer (on OpenStep / Mac OS X) or `xpdf` on *nix. Running `make check` will spellcheck the document using `aspell`, which ignores \LaTeX markup.

4 I want to be a tree!

Drawing trees can be done simply using the `synttree` package. This takes a nested square-bracketed expression and converts it into a tree. There are other packages, such as `xyfig`, that give a more flexible way of generating directed graphs, but for simple trees the `synttree` package is very convenient. It is particularly useful for trees generated programmatically; the syntax is simple and so it is very easy to automatically generate.

Incidentally, the floating figure to the right of this paragraph was generated by the `floatflt` package, which provides analogues of the standard floating environments with text wrapped around them.

The tree in Figure 1 was generated using this line of code:

```
\synttree[a [b [[c] [d]] [e]] [f] [g] [{h i j}]]
```

Each sub-tree is represented by a node name followed by a list of sub-trees in square brackets. Note that node names which contain spaces must be wrapped in braces.

5 Declaring Macros

As seen in the Notes section, declaring a macro is very easy using `\newcommand`. If there is a word or phrase you use often, then it can be convenient to declare a simple macro to use instead. This is particularly useful when you are unsure of the correct capitalisation of a phrase, or when it includes some \LaTeX markup.

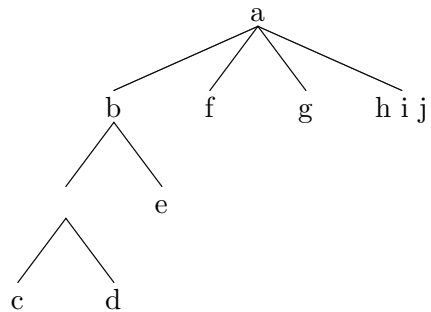


Figure 1: An example tree generated with the `synttree` package

6 Graphics in \LaTeX and pdf\LaTeX

\LaTeX allows the inclusion of EPS images. pdf\LaTeX allows the inclusion of more or less anything except EPS images. The `epstopdf` package allows you to incorporate EPS images into pdf\LaTeX documents. Another option is to use the `\DeclareGraphicsRule` command to define external programs to be used to convert images between formats.

The `\includegraphics` command has some other nice features, such as the ability to crop an image. The `trim` attribute is used in Figure 2 to trim 2cm off each side of the university logo. This can be particularly useful when you have an image that has a large border. `GNUplot` is particularly bad about cropping its output, and trimming 15mm or so from each side can save you a lot of space in a document.



Figure 2: An EPS figure in a PDF document.

7 Small Capitals

If you have a lot of text then from a distance, the page should look a constant shade of grey. If you write WORDS IN CAPITALS, THEN THEY LOOK MORE BLACK. To counter this, you can use the `\textsc` command TO MAKE SMALL CAPS, WHICH HAVE THE SAME SIZE (AND HENCE THE SAME GREY VALUE) AS LOWER CASE LETTERS.

8 Hyperlinks and References

The `hyperref` package allows you to insert hyperlinks in a document using the `\href` and `\url` commands. The former allows you to associate a link with some text; this text links to the departmental web site, for example. If you just wish to insert a URL, then the latter allows you to do, like this:<http://cs.swan.ac.uk>

Other advantages of the `hyperref` package is that it automatically turns references (including those pointing to the bibliography) into links inside the document when generating PDF or HTML output. This lets you easily check that all of your references point at the correct things simply by clicking on them, as long as you are using a PDF viewer that supports hyperlinks.

9 Friendly References

Generic references (`\ref`) in \LaTeX are useful for cross-referencing. They give a figure or a section number, allowing you to write things like Figure 2, and have it update depending on which number this figure actually has.

Sometimes it is nicer to be able to put a slightly more human-friendly reference, something like the 'on the last page,' or 'above.' Unfortunately, such things can become inaccurate when the document is modified and typeset again. The solution to this is the `varioref` package, which adds a little extra information. It can generate the forms of output shown in Table 1

Table 1: Output from the `varioref` package

\LaTeX	Output
<code>the figure on page \pageref{fig:eps}</code>	the figure on page 4
<code>Figure \ref{fig:eps}</code>	Figure 2
<code>Figure \vref{fig:eps}</code>	Figure 2 on the page before
<code>the figure \vpageref{fig:eps}</code>	the figure on the preceding page
<code>Sections \vrefrange{sec:makefiles}{sec:hyperref}</code>	Sections 3 to 8 on pages 2-4
<code>\vpagerefrange{sec:makefiles}{sec:hyperref}</code>	Sections on pages 2-4

10 HTML Output

The `tex4ht` program is a good way of generating HTML output. It provides a large number of options, and supports a significant number of \LaTeX packages. This is not always useful for papers, but for some other contexts it can be useful to make a version of your work available for browsing on the web as well as in printed form. If this is the case then many users find browsing a set of HTML pages nicer than reading a PDF document.

During the conversion process, a large document is usually broken up into smaller HTML pages, allowing downloads to take place much more quickly.