A Quick Guide to LaTeX

David Lane

January 26, 2006

Contents

1 Basics **1**

2 Typesetting Text **2**

3 Typesetting Mathematics **6**

4 What's Missing **11**

---

1 Basics

A LaTeX document consists of a *preamble* and a *main body*. The first line of the preamble is always the `\documentclass` command, which takes arguments `letter`, `report`, `article`, etc. If you're writing a quiz or an exam then either the `report` or `article` classes will do, and you needn't worry about the differences. This command takes optional arguments which you can use to change the font size and the type of paper (The default is 10pt and $8\frac{1}{2} \times 11$in). After `\documentclass` you may invoke various style files with the `\usepackage` command. For example, you might want to change the default margins using the package `geometry.sty`, and you may want to call the $\mathcal{AMS}$ packages `amsmath.sty`, `amsfonts.sty` and `amssymb.sty`, which extend many of the commands, symbols, and font sets (In fact I'm assuming here that you will *always* always be calling the the three $\mathcal{AMS}$ packages, and I won't distinguish between LaTeX commands and $\mathcal{AMS}$ commands). There are many, many such packages, and we can't even begin to cover them all here.

```
\documentclass[11pt, a4paper]{article}
\usepackage{amsmath, amsfonts, amssymb}
\usepackage[margin=1.25in, top=1in, bottom=1in, nohead]{geometry}
```

More packages and user defined commands

```
\begin{document}
```

The body of the document

```
\end{document}
```

After the packages you may also place user-defined commands (using `\newcommand`, `\newenvironment` or `\DeclareMathOperator`). We'll cover this as an advanced topic in a follow-on lecture.

It is standard practice to collect all of your aliases into your own style file and to place this in a common directory which LaTeX knows how to find. This prevents having to re-type the same information over and over again in the preamble of every LaTeX file.

2 Typesetting Text

Font Control. TeX's Computer Modern font family contains three "sub"-families—roman, sans-serif and typewriter. These families are further modified by their "shape", which can be upright, slanted, italic or small caps. And there are two different "series"—boldface (`\bfseries` or `\textbf{}`) and normal weight (`\mdseries` or `\textmd{}`). This paragraph, for example, is typeset in the normal weight, upright shape, Computer Modern roman font.

There are two versions of each command. For example, the command `\bfseries` will change all subsequent text to bold-face. To limit the scope of the operation enclose the affected text *and the command* in curly braces. The other version of the command is `\textbf`, which takes the bold-faced text as a mandatory argument to the function. So the code `{\bfseries Hello World}` and `\textbf{Hello World}` both produce the text **Hello World**.

Family	Shape	Sample
`\rmfamily` or `\textrm{}`	`\upshape` or `\textup{}`	This is the upright shape.
	`\itshape` or `\textit{}`	*This is the italic shape.*
	`\slshape` or `\textsl{}`	*This is the slanted shape.*
	`\scshape` or `\textsc{}`	This is the Small Caps shape.
`\sffamily` or `\textsf{}`	`\upshape` or `\textup{}`	This is the upright shape.
	`\itshape` or `\textit{}`	*This is the italic shape.*
	`\slshape` or `\textsl{}`	*This is the slanted shape.*
	`\scshape` or `\textsc{}`	This is the Small Caps shape.
`\ttfamily` or `\texttt{}`	`\upshape` or `\textup{}`	`This is the upright shape.`
	`\itshape` or `\textit{}`	*`This is the italic shape.`*
	`\slshape` or `\textsl{}`	*`This is the slanted shape.`*
	`\scshape` or `\textsc{}`	`This is the Small Caps shape.`

Table 1: The Computer Modern Text Fonts

Font sizes are controlled by the commands `\tiny`, `\scriptsize`, `\footnotesize`, `\small`, `\normalsize`, `\large`, `\Large`, `\LARGE`, `\huge` and `\Huge`.

Text Control. In TeX the spacing between words, end of line breaks and hyphenation are controlled automatically, regardless of the source code. In other words:

```
The rains in Spain stay mainly in the plain.
```

and

```
The        rains
in
Spain stay mainly    in
the plain.
```

both produce the same output.

By default, text is aligned on the left and right-justified. Text can be centered or flushed right (why would you?) using the environments `\begin{center}` ... `\end{center}` and `\begin{flushright}` ... `\end{flushright}`. The right-justification can be turned off using `\raggedright` or `\begin{flushleft}` ... `\end{flushleft}`.[1]

A new paragraph is created whenever TEX senses a blank line. The new paragraph begins on the next line and is automatically indented. You can remove (*resp.* force) indentation with the commands `\noindent` (*resp.* `\indent`).

If you want a space between paragraphs, use the newline command `\\` followed by a blank line. Extra vertical or horizontal white space is inserted using the commands `\hspace` and `\vspace`. For example, the extra space at the beginning of this sentence was created using the command `\hspace{1cm}`. There are also a series of short commands for inserting small amounts of extra horizontal space, as shown below:[2]

Command	Example	Result
`\,`	`x\,x`	x x
`\:`	`x\:x`	x x
`\;`	`x\;x`	x x
`\ `	`x\ x`	x x
`\quad`	`x\quad x`	x x
`\qquad`	`x\qquad x`	x x

Table 2: Horizontal Spacing Commands

Fancy Accents. There are commands for *umlaut*'s, *cedille*'s, and anything else you can imagine. Here is a sampling:

Code	Result	Code	Result	Code	Result
`\‘{a}`	à	`\’{e}`	é	`\"{u}`	ü
`\c{c}`	ç	`\r{o}`	o̊	`\t{oo}`	o͡o
`\v{o}`	ǒ	`\^{o}`	ô	`\~{o}`	õ
`\AE`	Æ	`\ae`	æ	`\ss`	ß

Table 3: European Accents and Characters

Minipages and Makeboxes. A `\makebox` and its variants `\mbox` and `\text` ($\mathcal{AMS}$ math package) place text in an invisible box which is then typeset as though it were a single letter. You will use `\text` frequently to include text inside a mathematical expression (see below for

[1] The justification in TEX looks much better than in MS-Word, so I recommend you leave it on unless you have a compelling reason not to.

[2] These commands also work in math mode.

math typesetting). The code:

```
$\{ p\in \mathbb{Z} \mid p \text{\, is prime}\}$
```

produces $\{p \in \mathbb{Z} \mid p \text{ is prime}\}$

With the optional length variable, `\makebox` can be used to help line up text on the page, a sort of "poor man's" tabbing. If the main argument is empty, then it acts like a slug of white space in places where `\hspace` doesn't seem to work. For example, the code

```
\noindent
\makebox[3in][l]{\textbf{a.} $y=\sin x$} \textbf{b.} $y=\cos x$\\

\vspace{.25in}

\noindent
\makebox[3in][l]{\textbf{c.} $y=\csc x$} \textbf{d.} $y=\sec x$\\
```

produces

a. $y = \sin x$ **b.** $y = \cos x$

c. $y = \csc x$ **d.** $y = \sec x$

Here is another "fiddle": If you type a newline command `\\` on a line by itself, TEX will complain that "there's no line here to end". Try typing `\mbox{}\\` instead, so that the `mbox` acts like an empty letter.

The `\minipage` environment places an entire paragraph in a box and then treats it like a letter. In quizzes, I use side by side minipages to position text alongside graphics using

```
\begin{minipage}[t]{3in}
...
\end{minipage}
\hfill
\begin{minipage}[t]{3in}
...
\end{minipage}
```

This trick usually requires some experimenting to get the vertical alignment correct.

The Tabular Environment. This environment is for making tables like the ones in this document[3].

[3]Actually I've used an improved version called `booktabs.sty`, which I recommend.

```
\begin{tabular}{lc} \hline
Font &Sample \\ \hline
math bold &$\mathbf{a}$\\
math italic &$\mathit{a}$\\
math roman &$\mathrm{a}$\\ \hline
\end{tabular}
```

Font	Sample
math bold	$\mathbf{a}$
math italic	a
math roman	a

Don't confuse the *table* environment with the *tabular* environment. A *table* is a "float" used to position *tabular*'s in the document (in bizarre and unpredictable ways). The only advantage of using `\begin{table}` ... `\end{table}` is that it provides captioning. But you can work around this anyway with the package `ccaption.sty` (which is what I've done here).

List Environments. The basic `list` environment is highly customizable, but for most purposes all you will need are the environments `itemize` and `enumerate`.

```
\noindent
This is a bulleted list.
\begin{itemize}
\item
This is a bullet.
\item
This is another bullet.
\begin{itemize}
\item Bullet.
\item Bullet.
\end{itemize}
\item
Yet another bullet.
\end{itemize}
```

This is a bulleted list.

- This is a bullet.
- This is another bullet.
  - Bullet.
  - Bullet.
- Yet another bullet.

The same thing using `enumerate`:

```
\noindent
This is a numbered list.
\begin{enumerate}
\item
This is a numbered item.
\item
This is another item.
\begin{enumerate}
\item Item.
\item Item.
\end{enumerate}
\item
Yet another item.
\end{enumerate}
```

This is a numbered list.

1. This is a numbered item.
2. This is another item.
   (a) Item.
   (b) Item.
3. Yet another item.

3 Typesetting Mathematics

In-line vs Display Math. TEX has two math modes, *in-line* and *display*. Display math is centered with vertical space on either side. To shift in and out of in-line mode use `\(` ... `\)` or `$` ... `$`, and to shift in and out of display math use either `\[` ... `\]` or `$$` ... `$$`[4]. For example, this is the integral $\int_{x=1}^2 e^x\, dx$ set in in-line mode, and this is the same integral

$$\int_{x=1}^2 e^x\, dx$$

in display math. The last sentence was typeset as follows:

```
For example, this is the integral $\int_{x=1}^2 e^x\, dx$ set in
in-line mode, and this is the same integral
$$
\int_{x=1}^2 e^x\, dx
$$
in display math.
```

If you want to force display math while in-line (and you frequently will on exams), use `\displaystyle\int_{x=1}^2 e^x\, dx`. Here are some more commands which change their "look" in display math:

Code	In-line Math	Display Math
`\lim_{x \to x_0} f(x)`	$\lim_{x\to x_0} f(x)$	$\lim\limits_{x\to x_0} f(x)$
`\sum_{k=1}^n k^2`	$\sum_{k=1}^n k^2$	$\sum\limits_{k=1}^n k^2$
`\bigcap_{k=1}^n A_k`	$\bigcap_{k=1}^n A_k$	$\bigcap\limits_{k=1}^n A_k$
`\bigcup_{k=1}^n A_k`	$\bigcup_{k=1}^n A_k$	$\bigcup\limits_{k=1}^n A_k$
`\frac{1}{x+1}`	$\frac{1}{x+1}$	$\frac{1}{x+1}$
`\binom{n}{k}`	$\binom{n}{k}$	$\binom{n}{k}$

Table 4: Changing Appearance in Display Math

There are, of course, many more of these. The last two commands `\frac` and `\binom` have special display versions `\dfrac` and `\dbinom`.

[4] A TEX-nical purist would insist on using the braces rather than the dollar signs, yet I've used the dollar signs all my life without any problems. It's a bad habit I can't break!

Math Fonts. In LaTeX the fonts used for mathematical typesetting are a different from those used for text. The default font for a letter in math mode is called *math italic*.

Font	Command	Result
Roman	`\mathrm{R}`	R
Italic	`\mathit{R}`	R
Bold	`\mathbf{R}`	$\mathbf{R}$
Sans-Serif	`\mathsf{R}`	R
Typewriter	`\mathtt{R}`	$\mathtt{R}$
Calligraphic	`\mathcal{R}`	$\mathcal{R}$
Fraktur	`\mathfrak{R}`	$\mathfrak{R}$
Blackboard Bold	`\mathbb{R}`	$\mathbb{R}$

Table 5: Math Fonts

Operators. The commands `\sin`, `\log`, and so forth are called *math operators* in LaTeX. If you forget the backslash then TeX will think the letters are regular symbols and will typeset them accordingly; typing `$sin x$` will produce $sinx$ rather than $\sin x$.

Code	Result	Code	Result
`\sin x`	$\sin x$	`\sinh x`	$\sinh x$
`\cos x`	$\cos x$	`\cosh x`	$\cosh x$
`\tan x`	$\tan x$	`\tanh x`	$\tanh x$
`\log x`	$\log x$	`\ln x`	$\ln x$
`\arcsin x`	$\arcsin x$	`\arccos x`	$\arccos x$
etc ...			

Table 6: Math Operators

There are some surprising omissions from this list. For example, there is no command for `\arccot`. This is easily solved by placing the line

```
\DeclareMathOperator{\arccot}{arccot}
```

in the preamble of your document.

Delimiters. These are "parentheses"-like constructions. If your delimiter surrounds something tall, you'll want to stretch it vertically. Use `\left` and `\right`:

`$\left( \dfrac{1}{x-2} \right)$` produces $\left(\dfrac{1}{x-2} \right)$

There is also a series of absolute resizing commands: `\bigl`, `\Bigl`, `\biggl` and `\Biggl` for left delimiters, and `\bigr`, `\Bigr`, etc. for right delimiters.

Code	Result	Code	Result
`[`	$[$	`]`	$]$
`\{`	$\{$	`\}`	$\}$
`|`	$\|$	`\\|`	$\|\|$
`\langle`	$\langle$	`\rangle`	$\rangle$

Table 7: Delimiters

Binary Relations and Operations. The difference between these and the other *symbols*, *operators* and *delimiters* is the amount of white space TEX places on either side (A binary operation has a bit more space than a binary relation). This is normally not an issue since TEX makes all these decisions automatically (but you will sometimes need to insert small amounts of horizontal space using `\,`).

Avoid these two mistakes:

- The symbol `|` is typeset as either a regular symbol or a delimiter, depending on the context, but not as a binary relation. Hence the code `$\{ x | x\in\mathbb{Q}\}$` produces $\{x|x \in \mathbb{Q}\}$, which is wrong. You should use `\mid` instead:

  `$\{ x \mid x\in\mathbb{Q}\}$` produces $\{x \mid x \in \mathbb{Q}\}$

- On the other hand, the colon `:` *is* a binary relation, so the code `$f: \mathbb{R} \to \mathbb{R}$` produces $f : \mathbb{R} \to \mathbb{R}$. There's too much space between the f and the colon. Use the command `\colon` instead:

  `$f\colon \mathbb{R}\to\mathbb{R}$` produces $f\colon \mathbb{R} \to \mathbb{R}$

Here some commonly used binary operations and relations (There are lots more).

Code	Result	Code	Result
`=`	$=$	`\propto`	$\propto$
`\ne`	$\neq$	`\approx`	$\approx$
`>`	$>$	`\notin`	$\notin$
`\ge`	$\geq$	`\subset`	$\subset$
`\gg`	$\gg$	`\subseteq`	$\subseteq$
`<`	$<$	`\mid`	$\mid$
`\le`	$\leq$	`\parallel`	$\parallel$
`\ll`	$\ll$	`\perp`	$\perp$
`\sim`	$\sim$	`\simeq`	$\simeq$
`\in`	$\in$	`\ni`	$\ni$

Table 8: Binary Relations

Code	Result	Code	Result
`+`	$+$	`\div`	$\div$
`-`	$-$	`\ast`	$\ast$
`\pm`	$\pm$	`\cap`	$\cap$
`\cdot`	$\cdot$	`\cup`	$\cup$
`\times`	$\times$		

Table 9: Binary Operations

Other Symbols. These are partial lists. The usage is self-explanatory:

Code	Result	Code	Result
`\rightarrow`	$\rightarrow$	`\leftarrow`	$\leftarrow$
`\longrightarrow`	$\longrightarrow$	`\longleftarrow`	$\longleftarrow$
`\hookrightarrow`	$\hookrightarrow$	`\hookleftarrow`	$\hookleftarrow$
`\twoheadrightarrow`	$\twoheadrightarrow$	`\twoheadleftarrow`	$\twoheadleftarrow$
`\Rightarrow`	$\Rightarrow$	`\Leftarrow`	$\Leftarrow$
`\Longrightarrow`	$\Longrightarrow$	`\Longleftarrow`	$\Longleftarrow$
`\leftrightarrow`	$\leftrightarrow$	`\Leftrightarrow`	$\Leftrightarrow$
`\mapsto`	$\mapsto$	`\longmapsto`	$\longmapsto$

Table 10: Arrow Commands

Code	Result	Code	Result	Code	Result
`\alpha`	α	`\beta`	β	`\gamma`	γ
`\delta`	δ	`\epsilon`	ϵ	`\varepsilon`	ε
`\zeta`	ζ	`\eta`	η	`\theta`	θ
`\iota`	ι	`\kappa`	κ	`\lambda`	λ
`\mu`	μ	`\nu`	ν	`\xi`	ξ
`\pi`	π	`\rho`	ρ	`\sigma`	σ
`\tau`	τ	`\upsilon`	υ	`\phi`	ϕ
`\varphi`	φ	`\chi`	χ	`\psi`	ψ
`\omega`	ω	`\Gamma`	Γ	`\Delta`	Δ
`\Theta`	Θ	`\Lambda`	Λ	`\Xi`	Ξ
`\Pi`	Π	`\Sigma`	Σ	`\Upsilon`	Υ
`\Phi`	Φ	`\Psi`	Ψ	`\Omega`	Ω

Table 11: Greek Letters

Code	Result	Code	Result	Code	Result
`\aleph`	$\aleph$	`\ell`	ℓ	`\imath`	$\imath$
`\dag`	$\dagger$	`\ddag`	$\ddagger$	`\S`	$\S$
`\P`	$\P$	`\forall`	$\forall$	`\exists`	$\exists$
`\infty`	∞	`\emptyset`	$\emptyset$	`\angle`	$\angle$
`\partial`	∂	`\varnothing`	$\varnothing$	`\neg`	$\neg$

Table 12: Miscellany

Multi-line Display Math. There are several options, but the three I use most often are `\align*`, `\gather*` and `\alignat*`. Note that these automatically switch to display math mode, so there's no need to surround them with `\[` and `\]`. The code

```
\begin{align*}
\ln(xy)&=\ln\left( e^{\ln x} e^{\ln y} \right)\\
       &=\ln\left( e^{\ln x +\ln y} \right)\\
       &=\ln x + \ln y
\end{align*}
```

produces

$$\begin{aligned}\ln(xy) &= \ln\left(e^{\ln x}\cdot e^{\ln y}\right)\\ &= \ln\left(e^{\ln x+\ln y}\right)\\ &= \ln x + \ln y\end{aligned}$$

The `align*` environment supports multiple columns (To make two columns you actually need three ampersands; the middle one is interpreted as a column separator).

```
\begin{align*}
\ln(xy) &=\ln\left( e^{\ln x} \cdot e^{\ln y} \right)
    & &\text{definition of inverse functions}\\
&=\ln\left( e^{\ln x +\ln y} \right) & &\text{exponent rule}\\
&=\ln x + \ln y &&
\end{align*}
```

produces

$$\begin{aligned}\ln(xy) &= \ln\left(e^{\ln x}\cdot e^{\ln y}\right) && \text{definition of inverse functions}\\ &= \ln\left(e^{\ln x+\ln y}\right) && \text{exponent rule}\\ &= \ln x + \ln y\end{aligned}$$

In the above example there's too much space between the columns. To change this use the variant `alignat*`, which inserts no space between the columns. You can then control the

space directly with `\quad` or `\qquad`, like this:

```
\begin{alignat*}{2}
\ln(xy) &=\ln\left( e^{\ln x} \cdot e^{\ln y} \right)
    &&\qquad \text{definition of inverse functions}\\
&=\ln\left( e^{\ln x +\ln y} \right) &&\qquad \text{exponent rule}\\
&=\ln x + \ln y &&
\end{alignat*}
```

produces

$$\begin{aligned}
\ln(xy) &= \ln\left(e^{\ln x} \cdot e^{\ln y}\right) && \text{definition of inverse functions}\\
&= \ln\left(e^{\ln x+\ln y}\right) && \text{exponent rule}\\
&= \ln x + \ln y
\end{aligned}$$

The `\gather*` environment is used when you need successive lines gathered together and centered.

```
\begin{gather*}
3e^{x-1}=6\\
e^{x-1}=2\\
x-1=\ln 2\\
x=1+\ln 2
\end{gather*}
```

produces

$$\begin{gathered}
3e^{x-1} = 6\\
e^{x-1} = 2\\
x - 1 = \ln 2\\
x = 1 + \ln 2
\end{gathered}$$

4 What's Missing

This should be enough to get started. Some topics which have been omitted are:

- Graphics! There are two ways to go: LaTeX easily inserts encapsulated postscript files which you must create in Mathematica, Adobe Illustrator, etc.
- Automatic numbering of equations, theorems, figures, tables, sections and subsections.
- Labeling and referencing equations, theorems, etc. within the document.
- The differences between the document classes.

- Use of packages such as `ccaption.sty`, `multicol.sty`, `booktabs.sty`, etc.
- Generating a Table of Contents and Bibliography.
- User defined functions and environments.