Notes on LaTeX

Adrian Campbell–Smith

October 1999

Abstract

I present some (hopefully) useful notes which serve as an introduction, ready-reference, and indication of some of my personal preferences for using the LaTeX typesetting program. These notes are not supposed to be totally comprehensive; they are designed to introduce some of the most useful commands and options in LaTeX roughly in the order in which they appear in a document.

1 The Source File

The source file (which should have a `.tex` tag) comprises two parts: the *preamble* and the *body*. The preamble contains general commands which affect the overall appearance of your final document, and the body contains the text of your document, with formulæ, figures, and typesetting commands. By running the program `latex` on your source, you can generate printable output: see section 9.

Sources at your disposal for learning LaTeX are: the LaTeX source of papers from archives on the web (hep–th etc.), your colleagues, and the many LaTeX Introductions and Guides; combined searches for "latex" and specific package or command names on yahoo.com and alltheweb.com can be very successful.

Before starting to create documents, there are some points to notice:

LaTeX 2_ε and LaTeX 2.09

The most up-to-date version of LaTeX, and that run in this department, is 2_ε. This is backwards compatible with the older version 2.09. There are a few differences between the two, principally that 2_ε compiles files more quickly, and that the American Physical Society document style, REVTeX, is unavailable (as yet) in 2_ε. Unless you wish to submit to the APS, you should certainly choose LaTeX 2_ε.

emacs

`emacs` is a wonderful text editor which is wholeheartedly recommended for the creation of source files. It takes some effort to get to know how to use it well, but it is well worth doing so. It has its own latex mode, which enables it to recognize a lot of LaTeX syntax an run useful macros as a result; you can also run `latex` and view resulting `dvi` and postscript files within the program itself. In the long run, studying the help pages inside `emacs` (type `Ctrl-h i`) will save you lots of time and effort when creating LaTeX documents. Note that `emacs` also has C and Fortran modes, and a whole host of other stuff (including a feature which computes the sunrise time at your longitude/latitude!).

Comments

Anything which follows the symbol % on a given line in your source file is ignored by LaTeX. Use % for comments to yourself, or for temporary removal of a part of your source file.

2 The Preamble

All source files start with the command

```
\documentclass{...}
```

where the ellipsis is replaced with one of

```
letter   article   report   book
```

These are the different classes of document available. The `letter` class will not be discussed further here. The principal difference between the other three classes is in the section numbering, and the existence of chapters and parts in the `report` and `book` classes. Papers are in general written as `article` class documents, theses as `reports`. A crucial exception to this is for authors who wish to use the American Physical Society's own document style, REVTeX. As mentioned above, this is only available in the old version 2.09, in which case one has to open the preamble with (indeed the differing command here tells LaTeX 2ε to run in backward-compatible mode)

```
\documentstyle{revtex}.
```

When syntactical differences between 2ε and 2.09 occur as above, I shall do my best to point them out, especially since REVTeX is a useful and attractive document style, and is almost essential for submission to APS journals.

2.1 Style Options

The different style options and external packages which can be loaded by LaTeX are included with the command

```
\usepackage{...}
```

Multiple options/packages can either be loaded with several `\usepackage` commands, or by separating the options with commas (and no spaces) in one command. The ellipsis is replaced with the name of the required package. I will mention some of the most useful:

```
a4   a4wide   amsmath   amssymb   axodraw   cite   graphics   fancyhdr
```

The first two of these tell LaTeX to fit your document nicely onto an A4 page, the latter with a slightly wider text width. This document is written with the `a4wide` option. The two packages starting with `ams` are for mathematical typesetting (see section 4): `amsmath` loads some extra equation-building environments, and `amssymb` loads extra mathematical symbols, e.g. $\dotplus, \lll$. The `axodraw` package must be downloaded from the web (or copied from a colleague), and is the best package for drawing Feynman graphs and flowcharts. It is very easy to use, and comes with its own excellent (and pedagogical) manual, so will not be introduced in any detail here. The `cite` package collates your citations in a nice way, such that references like [1,2,3,4,9] are changed automatically to [1–4,9]. The `graphics` package enables a bunch of nice commands for handling postscript figures, to which I shall return briefly in section 5. Finally `fancyhdr` is a package which generates attractive headers for pages, useful for theses.

In LaTeX 2.09, these packages are loaded as "options" to the `\documentstyle` command:

```
\documentstyle[amsmath,cite]{revtex}
```

It is a general feature of LaTeX that optional arguments for commands appear in square brackets, while essential arguments are enclosed in curly braces.

The REVTeX document style has its own set of useful options/packages. Principal among these are

```
aps   12pt   preprint   tighten   twocolumn   eqsecnum
```

The `aps` option tells LaTeX to fit your document nicely onto the page (the text width is larger than the `a4` style), and conforms to the APS' requirements for section headings (centred, in capitals) and section numbering (roman), etc. One option from the set {`aps,osa,seg`} must appear, and `aps` is recommended. `12pt` and `preprint` both force REVTeX into a "preprint" style (unlike the 10-point style which appears in APS journals) with double-spacing and 12-point type. They differ in the presentation of the abstract and the exact format of the printed page: which you choose is a matter of personal taste. To close up the double-spacing for either of these styles, choose `tighten`. To have a look at your document in the two-column style which appears in the journal use `twocolumn`, although mysterious things happen to the abstract, and long equations are not broken. This option should only really be used to get an idea of the number of pages the paper will occupy in the journal; in general it is best to leave the editors to change your paper to a two-column format, and stick to a `tighten`ed "preprint" style. `eqsecnum` numbers your equations by section, which is useful. This can be achieved outside REVTeX with a little effort, see section 4.8.

2.2 Title and Author

You can declare who you are and what you are writing about in the preamble with the commands

```
\title{Notes on \LaTeX}
\author{Adrian Campbell--Smith}
\address{Nonsense Land}
\date{October 1999}
```

Note that the `\address` command is only available in REVTeX; the address can be put into the author field (after a carriage return `\\`) in other document classes:

```
\title{Notes on \LaTeX}
\author{Adrian Campbell--Smith\\Nonsense Land}
\date{October 1999}
```

LaTeX will build the title out of these declarations once you ask it to do so in the body of your document (see section 3). Each of the declarations is optional; in particular, you may not wish to mention a date. For the record, today's date can be generated with `\today`. References to e.mail addresses and the like can be made with the `\thanks{...}` command, which if put after an author's name will generate a labelled footnote containing the argument of the `\thanks`.

2.3 Declarations

The rest of the preamble comprises any declarations you may wish to make: new commands, the page style to use, etc.

2.3.1 New Commands

A new command is defined by (replacing the old TeX `\def`):

```
\newcommand{\boldsigma}{\mbox{\boldmath$\sigma$}}
```

means that any time you type `\boldsigma` in the body of your document, LaTeX reads it as `\mbox{\boldmath$\sigma$}` which, for the record, is how to generate bold Greek characters (without `amsmath`). I strongly discourage you from using this to generate abbreviations for things which you type commonly in your documents. You are much better off creating a global abbreviation inside `emacs`. In particular, you can get into problems if you happen to choose a newcommand name which already exists somewhere in LaTeX or one of its loaded packages (`amssymb` can cause particular problems here on account of the enormous number of extra symbols).

2.3.2 The Page Style

The page style is set by the declaration

```
\pagestyle{...}
```

where the ellipsis is to be replaced with one of the following:

`empty` Nothing on the page except your text (i.e. no page number).

`plain` Generates a page number at the bottom (used in this document).

`headings` Page number at the bottom, and header determined by the document class.

`myheadings` Page number at the bottom, your own decision about the header.

See your manual for a detailed description of how to use the `(my)headings` styles. Sometimes you may wish that a particular page has no number at the foot, while the rest of your document has the `plain` page style. This can be achieved with the `\thispagestyle{empty}` command at the appropriate place in the body.

If you are unsatisfied with the size of the printed page generated then you can set the text width and height yourself in the preamble:

```
\setlength{\textwidth}{13.7cm}
\setlength{\textheight}{23cm}
```

If you wish for LaTeX to be relaxed rather than rigorous about adhering to the text length (which is often a good idea for articles), then use the command `\raggedbottom` in the preamble.

3 The Body

The preamble is ended and the body begun with the command

```
\begin{document}
```

Immediately after this a typical series of commands might be

```
\maketitle

\begin{abstract}
In this article our objective is to waffle endlessly about some problems
which we could not solve except in the most ludicrous approximations.
\end{abstract}

\tableofcontents
```

The first of these creates the title according to the declarations (`\author`, `\title` etc.) you have made in the preamble, and the document class. `\tableofcontents` builds a table of contents: it will take LaTeX several runs until it has it all correct. Similar commands will generate lists of figures and tables, though these are principally of use only for theses. If you do not like the way LaTeX creates your title and abstract, you can simply create it yourself with LaTeX's formatting commands (see below) and forget about the declarations in the preamble. To create a separate page on which your title etc. appear, use the environment

```
\begin{titlepage}
...
\end{titlepage}
```

in which you are free to do whatever you like: use commands like

```
\small   \large   \Large   \LARGE   \huge   \Huge
```

to change the type size, and things like

```
\textbf{...} \textit{...} \textsl{...} \textsc{...} \textsf{...} \texttt{...}
```

to generate **bold**, *italic*, *slanted*, SMALL CAPS, sans serif and `type`. To use these in any of the mathematics modes (see section 4), replace `\text` with `\math`. Note the existence of the command `\emph{...}` which generates *emphasized text* with the correct spacing around it. For information on spacing and formatting, see sections 3.2 and 3.3 respectively.

3.1 Logical Units

After generating your title etc. as above you are ready to start typing the main body of your document. Logical units are automatically generated and numbered with the commands

```
\part  \chapter   \section   \subsection   \subsubsection
```

The first is only available in the `book` document class, the second in the `book` and `report` classes, and the remainder in all the classes. There are actually units below `\subsubsection`s should you need them, unfortunately called `\paragraph` and `\subparagraph`. Each of these commands takes an optional and an essential argument:

```
\section[short_title]{Logical Units}
```

If you include the optional short title, it will appear in the page header and table of contents. The longer title appears in your document, as with "Logical Units" above.

Appendices are generated with the `\appendix` command, after which you use `\section` commands to actually generate the appendices, which will then be numbered A, B, and so on, and given the titles which you put as the argument of the `\section` command as normal.

If you do not wish your ((sub)sub)section to be numbered, then simply place a `*` before the argument. I used this to generate the unnumbered section on Emacs at the beginning, and this construction is useful for such things as acknowledgements (note that REVTEX has its own `\acknowledgments` command (and note the American spelling) which has the same result). Logical units with asterisks will not appear in the table of contents, and the next logical unit is numbered correctly (i.e. as though your asterisked section were not present).

New paragraphs within sections are obtained by leaving a blank line in your source file:

```
...end of paragraph on this.

Now we start a new paragraph on that...
```

LATEX will indent each new paragraph as it goes along. To suppress this indentation for any reason, use `\noindent`. In general carriage returns can be forced with the command `\\`.

3.2 Spacing and Line(Page)-Breaking

LATEX almost always gets the spacing in your document right. When you wish to add extra space for whatever reason, use the commands

```
\hspace{...}   \vspace{...}
```

for horizontal and vertical space respectively: the argument is the space, with units: 2pt, 3cm, 1.5in, etc. Both can take negative arguments if necessary. Note also that the carriage return command `\\` takes an optional argument `\\[...]` for an extra amount of space (in some units) to leave after the carriage return (good for leaving a blank gap between lines in an `align` environment see section 4).

Sometimes LATEX will apparently fail to pay any attention to your horizontal and vertical spacing commands, if it thinks the page looks better without. Therefore there are two "I am angry with you, I absolutely insist you put this space in" commands:

```
\hspace*{...}   \vspace*{...}
```

which LaTeX will perform no matter what it thinks.

Page breaking, should you wish to force LaTeX to do so at any point, is done with the command `\newpage`. It is not worth messing around with such things until you have the final draft of your document. The addition of a few words here and there can make an enormous difference to where LaTeX makes its page breaks.

You can change the line spacing in your document with ease: the simplest way is to use the package `setspace` (with `\usepackage{setspace}` in the preamble) which then enables the commands

```
\singlespacing
\onehalfspacing     (this is equivalent to (1.5)\singlespacing)
\doublespacing
```

which affect the spacing accordingly for text following the command. This is useful if you want a double-spaced thesis, but single-spaced title pages, contents table, acknowledgements, etc.

3.3 Formatting Text

There are several environments which can be used for formatting text in different ways (and note that they can contain mathematics as well). To centre some text, use the following environment, with different centred lines separated by `\\` commands:

```
\begin{center}
...\\
...
\end{center}
```

Note the American spelling of `center`.

> This text is generated in a `quotation` environment, which is useful for highlighting particular bits of text or for quoting long passages from other sources. The text is centered and the lines shorter.

There are a couple of environments for making text `flushleft` and `flushright` e.g.:

This text is generated with <code>\begin{flushright}</code>,
(Separate lines with \\)
and ended with <code>\end{flushright}</code>

There are a number of useful list-making environments, principal among them being `itemize` and `description`. `itemize` produces a bulletted list as follows: the input

```
\begin{itemize}
\item  Item one.
\item  Item two.
\item  And so on.
\end{itemize}
```

generates the list:

- Item one.
- Item two.
- And so on.

The `description` environment is used when you want something other than a bullet at the beginning: the source which produced the list of page styles in section 2.3.2 is as follows:

```
\begin{description}
\item[\texttt{empty}]  Nothing on the page except your text (i.e. no
page number).
\item[\texttt{plain}] Generates a page number at the bottom (used in
this document).
\item[\texttt{headings}] Page number at the bottom, and header
determined by the document class.
\item[\texttt{myheadings}]  Page number at the bottom, your own
decision about the header.
\end{description}
```

There is also the `enumerate` environment which generates a numbered list with the syntax as above.

3.4 Labelling and Cross-referencing

Any time you wish, you can give a part of your text a label with the command `\label{the_label}`, and then cross-reference to this using `\ref{the_label}`. For example, if you put the `\label` command after a `\section{...}`, then a subsequent `\ref` will call the section number; a `\label` appearing inside an equation (see section 4) will cause `\ref` to generate the equation number. Figures, tables, subsections, chapters, etc. can all be cross-referenced this way. When calling `\ref` it is worth knowing of the existence of `~` which generates an unbreakable space: LaTeX will never make a line-break between "section" and its number in the following example

```
...refer to section~\ref{sec:whatever} for further information.
```

This is also useful for ensuring that names and their associated initials are not separated.

3.5 Citations

Citations are made using the `\cite{...}` command. What appears as the argument is the keyword of the citation you wish to call. Later, LaTeX will collect your citations in a bibliography at the end of the document. Again, it is worth attaching the citation to the preceding text with an unbreakable space as in

```
...refer to reference~\cite{witten99} for the truth, stated without proof.
```

For LaTeX to call this citation, there must be a bibliography item which is called `witten99`: bibliographies are discussed further in section 7.

3.6 ShowKeys

There is a fabulous package called `showkeys` which can be called in the preamble with the usual `\usepackage{showkeys}` command. It makes LaTeX print out the actual `label`s, `ref`erences and `cite`ation keywords which you have used. This is very useful for keeping track of what label you gave to something to which you wish to subsequently refer. The labels thus printed will show up in both the `dvi` and postscript files, so they can be printed. They do not affect the typesetting at all, and will all disappear as soon as you comment out or delete the `\usepackage` command.

3.7 Footnotes

Footnotes are very simple: they are generated by the `\footnote{...}` command, which takes the text of your footnote as its argument[1].

[1] A footnote

4 Mathematics

The real power of LaTeX is in typesetting mathematics. The more you use the program, the more you will find a feel for creating good-looking formulæ. I cannot recommend the package `amsmath` (and its symbols set `amssymb`) strongly enough: it enables a set of environments for equations which are hugely superior to the basic ones available in LaTeX. Basic REVTeX is also quite good, essentially upgrading the basic environments to the `amsmath` standard. I will therefore mention the standard environments quickly before giving details of the superior `amsmath` commands.

4.1 Basic Mathematics in LaTeX

Mathematics embedded in text is delimited with `\(...\)` or (completely equivalently) `$...$`. A displayed (centred) formula is obtained with `\[...\]`. For example, the source

```
A simple U(1) gauge field theory is defined by the action functional
\[ S= \int d^4x\; \left[ -\frac{1}{4} F_{\mu\nu}F^{\mu\nu} +
\overline{\Psi} (i\gamma^\mu D_\mu) \Psi \right],\]
where $F_{\mu\nu}$ and \(D_\mu = \partial_\mu + ieA_\mu\) are
respectively the U(1) curvature tensor and covariant derivative.
```

generates the LaTeX output:

A simple U(1) gauge field theory is defined by the action functional

$$S= \int d^4x\; \left[-\frac{1}{4} F_{\mu\nu}F^{\mu\nu} + \overline{\Psi} (i\gamma^\mu D_\mu) \Psi \right],$$

where $F_{\mu\nu}$ and $D_\mu = \partial_\mu + ieA_\mu$ are respectively the U(1) curvature tensor and covariant derivative.

Numbered equations are generated by the environments `equation` and `eqnarray`. For example, by replacing the `\[` and `\]` above with `\begin{equation}` and `\end{equation}` we obtain:

A simple U(1) gauge field theory is defined by the action functional

$$S= \int d^4x\; \left[-\frac{1}{4} F_{\mu\nu}F^{\mu\nu} + \overline{\Psi} (i\gamma^\mu D_\mu) \Psi \right], \tag{1}$$

where $F_{\mu\nu}$ and $D_\mu = \partial_\mu + ieA_\mu$ are respectively the U(1) curvature tensor and covariant derivative.

The `eqnarray` environment is used for equations broken across several lines. The environment asks you to split each line of your equation into three logical bits separated by `&` symbols, and separate each line by `\\`. Corresponding logical bits appear vertically aligned in the output. The input

```
\begin{eqnarray}\label{Fdef}
F_{\mu\nu} &\doteq& \partial_{[\mu} A_{\nu]}\\
&=& \partial_\mu A_\nu - \partial_\nu A_\mu \nonumber\\
\Rightarrow \partial^\mu F_{\mu\nu} &=& 0 \label{dF=0}.
\end{eqnarray}
```

generates the output

$$\begin{aligned} F_{\mu\nu} &\doteq \partial_{[\mu} A_{\nu]} && (2)\\ &= \partial_\mu A_\nu - \partial_\nu A_\mu \\ \Rightarrow \partial^\mu F_{\mu\nu} &= 0. && (3) \end{aligned}$$

Note the use of `\nonumber` to leave the second line unnumbered. This environment is often used for typesetting single equations too large to fit onto one line; there are several better ways of doing this with `amsmath`, and I will mention the most flexible in the following. Note in particular that the spacing around the = signs above is too large: this is an artifact of the array generation, and looks ugly when near to `equations`, such as equation (1), where the spacing is correct. All the `amsmath` environments correct this.

4.2 Labelling

To cross-reference an equation, you put in a `\label` as above: note the position of the `\labels` in the references to equations (2) (`\ref{Fdef}`) and (3) (`\ref{dF=0}`). The labels can appear anywhere between the start of each line and the `\\` symbol. This is because the equation counter is only incremented upon the appearance of an `\end{eq...}` or a `\\` command, and LaTeX knows that the `\ref` command refers to the counter of whatever environment is currently being type-set.

As with sections, `equation` and `eqnarray` have numberless versions, namely `equation*` and `eqnarray*`. The starred `equation` does not differ from a basic `\[...\]`, but the starred `eqnarray` can be useful.

4.3 Mathematics with amsmath

The `amsmath` package improves greatly on the basic `equation` and `eqnarray` environments. They are respectively replaced with `gather` and `align`, which are much more robust, get spacing correct, never write equation numbers over your text (which happens in `eqnarray` and `equation` when the equations are very long), and are simpler to use.

The `gather` environment allows multiple lines (separated by `\\` as usual) and centres each line. For one-line formulæ this behaves like an improved `equation` environment. For example, the input:

```
\begin{gather}
(a+b)^2  = a^2 + 2 ab + b^2 \\
(a+b)\cdot(a-b) = a^2 - b^2
\end{gather}
```

generates the output (you could use `\nonumber` to suppress an equation number):

$$(a+b)^2 = a^2 + 2ab + b^2 \tag{4}$$

$$(a+b)\cdot(a-b) = a^2 - b^2 \tag{5}$$

To obtain aligned multi-line output, the simplest approach is to use an `align` environment. Unlike `eqnarray` this does not create an array, but simply aligns whatever `&` symbols you put into the formula: the input

```
\begin{align}
x^2 +y^2 &= 1\\
x&= \pm \sqrt{1-y^2}
\end{align}
```

generates the output

$$x^2 + y^2 = 1 \tag{6}$$

$$x = \pm\sqrt{1-y^2} \tag{7}$$

More complicated alignments with proper spacing can also be obtained effortlessly:

```
\begin{align}
(a+b)^2 &= a^2+2ab+b^2 &x^2+y^2 &= 1\\
(a-b)^2 &= a^2-2ab+b^2 &x &=\pm\sqrt{1-y^2}
\end{align}
```

generates

$$(a+b)^2 = a^2 + 2ab + b^2 \qquad x^2 + y^2 = 1 \tag{8}$$

$$(a-b)^2 = a^2 - 2ab + b^2 \qquad x = \pm\sqrt{1-y^2} \tag{9}$$

If you thought the above would look better with the x symbols aligned rather than the = signs, then you only have to move the `&` symbols appropriately. The `align` environment can be used to split long equations using `\nonumber` to prevent the generation of an equation number. Note that both `gather` and `align` have starred versions, which suppress the generation of equation numbers.

There is also the so-called `multline` environment, which is another attractive way to typeset long mathematical relations. It left justifies the first line, centres any middle parts and then right justifies the last line:

$$\begin{multline}\int \bar{d}^4 q\; d^4\theta\; \frac{1}{2} 4e^2 G_4\, \Phi_+(-p,\theta,\bar\theta)\, \Delta^{VV}(q)\, \Phi_+^\dagger(p,\theta,\bar\theta) \\ = \int \bar{d}^4 q\; 2e^2 G_4\, \frac{2}{q^4}(1-\xi) \int d^4\theta\; \Phi_+^\dagger(p,\theta,\bar\theta)\, \Phi_+(-p,\theta,\bar\theta). \tag{10}\end{multline}$$

Finally, you can insert text into an `align` environment while preserving the alignment:

```
\begin{align*}
\Psi &\longrightarrow {\frak g}\Psi\\
\overline{\Psi} &\longrightarrow \overline{\Psi} {\frak g}^{-1}\\
\intertext{and the gauge field transforms as}
A &\longrightarrow {\frak g} \left( A+\partial\right) {\frak g}^{-1}
\end{align*}
```

generates:

$$\begin{aligned}\Psi &\longrightarrow \mathfrak{g}\Psi \\ \overline{\Psi} &\longrightarrow \overline{\Psi}\mathfrak{g}^{-1}\end{aligned}$$

and the gauge field transforms as

$$A \longrightarrow \mathfrak{g}\left(A+\partial\right)\mathfrak{g}^{-1}$$

Of course, the editor and referee will not notice any of this beautiful typesetting, but at least you will know that your document looks correct (unless of course you let your supervisor get hold of the source file, in which case literally anything can happen to your wonderful paper).

4.4 Spacing and Breaking

Most of the time LaTeX gets the spacing in your equations just right, but occasionally it does not understand what you mean, or you just prefer something else. An excellent example of this is integration, where LaTeX does not appreciate that the measure needs to be separated from the integrand. Compare:

```
\begin{gather*}
\int d^4 x C(x) \delta^{(4)}(x) = C(0)\\
\int d^4 x\; C(x)\, \delta^{(4)}(x) = C(0)
\end{gather*}
```

which generates

$$\int d^4 x C(x) \delta^{(4)}(x) = C(0)$$

$$\int d^4 x \; C(x) \, \delta^{(4)}(x) = C(0)$$

There are several different sizes of space in LaTeX which you can use to move things around:

```
\,    \;    \:    \      \quad    \qquad    \!
```

These are respectively: thin space, medium space, thick space, inter-word-space (that is backslash-space), a "quad" a double "quad," and a negative thin space. If you are desperate, you can always use `\hspace{...}` (which can take a negative argument), see section 3.2.

Sometimes LaTeX chooses to change page in the middle of an `eqnarray` or `align`, at one of your line breaks `\\`. You can prevent this by using `\\*` which says "break line but do not break page." Of course at this stage LaTeX may choose to move your entire equation onto the next page: it is not worth playing with any of this until you have a *final* draft of your paper. Remember that `\\[...]` takes a useful optional argument, good for extra space in `align`ed formulæ.

4.5 Text in Mathematics

Aside from text generated by the `\intertext` command, you may wish to put other text into your mathematics. While in any of the mathematics-generating environments, LaTeX treats everything as mathematics until you tell it to do something else. The commands

```
\mathrm{...}   \mathit{...}
```

are used to generate correctly spaced roman and italic type inside mathematics. Note that since LaTeX still thinks it is producing mathematics, it will ignore any spaces between words. If you wish to include more than one word (i.e. with spaces), use the `\text{...}` command:

```
\begin{gather*}
{\frak L}_{eff} \quad\text{is to be compared with}\quad {\frak
L}_{\mathit{eff}}
\end{gather*}
```

generates

$$\mathfrak{L}_{eff} \quad\text{is to be compared with}\quad \mathfrak{L}_{\mathit{eff}}$$

In the first effective Lagrangian, LaTeX treats the subscript "eff" as the product $e \times f \times f$, whereas in the second it spaces it correctly for text. The command `\mathrm` generates non-italic text with the correct spacing inside mathematics: $\mathfrak{L}_{\mathrm{eff}}$.

4.6 Delimiters

Delimiters (, [, {, | etc. are produced with the obvious direct input, save that { is produced with `\{`. LaTeX will expand a delimiter to make it as big as the symbols between if you prepend it with a `\left` and prepend a `\right` to its partner closing delimiter. In the following example several different uses of `\left` and `\right` appear: note the use of `\left.` which produces no output, but keeps LaTeX from complaining that there is an unmatched delimiter (in this case the matching delimiter is the "to be evaluated at" sign):

```
\begin{gather*}
S[\phi] = \int d^4x\; \left.\left( -\frac{1}{4} F^{\mu\nu}F_{\mu\nu} +
\frac{1}{2} \left| D_\mu \phi \right|^2 \right)\right|_{|\phi|^2=1}
\end{gather*}
```

which produces

$$S[\phi] = \int d^4x \; \left(-\frac{1}{4} F^{\mu\nu} F_{\mu\nu} + \frac{1}{2} \left| D_\mu \phi \right|^2 \right) \Bigg|_{|\phi|^2=1}$$

If for any reason LaTeX does not produce a delimiter as large as you would wish, you can try `\big(`, `\Big(` etc., which produce larger delimiters (the commands can be used with delimiters other than `( )`) automatically.

```
\begin{gather*}
\text{The integral} \quad\int db\; \big( A(b)+B(b) \big) \quad\text{looks better
than}\quad \int db\; \left( A(b)+B(b) \right)
\end{gather*}
```

$$\text{The integral} \quad \int db\; \big(A(b)+B(b) \big) \quad \text{looks better than} \quad \int db\; \left(A(b)+B(b) \right)$$

since LaTeX does not understand the distinction between the delimiters of the arguments and the delimiters of the integrand. It is normally worth seeing what `\left` and `\right` produce before you start to play. In the example with ϕ above, the modulus signs around $D_\mu \phi$ are no bigger than they would have been if I had left out the `\left` and `\right`: as you get used to LaTeX you will get a feel for when these commands are unnecessary like the case above.

Note that for correlation functions there exist `\langle` and `\rangle` (which can of course be used with `\left` and `\right`) which are much better than `<` and `>` as exemplified in the following:

```
\begin{gather*}
<\phi(x) \phi(0) > \neq \langle \phi(x) \phi(0) \rangle.
\end{gather*}
```

$$< \phi(x) \phi(0) > \neq \langle \phi(x) \phi(0) \rangle.$$

If you have used `\left` in an equation which is broken across several lines, you need to match all the `\left` commands with `\right` commands *before* you put in a line break. A silly example:

```
\begin{align*}
S[\Psi_1,\Psi_2] = \int d^4x\;& \left[ \bar{\Psi}_1 (i\gamma^\mu
D_\mu ) \Psi_1 +\right.\\ &\qquad+\left.\overline{\Psi}_2 (i\gamma^\mu
D_\mu ) \Psi_2 \right]
\end{align*}
```

which generates

$$\begin{aligned} S[\Psi_1,\Psi_2] = \int d^4x\;& \left[\bar{\Psi}_1 (i\gamma^\mu D_\mu) \Psi_1 +\right.\\ &\qquad+\left.\overline{\Psi}_2 (i\gamma^\mu D_\mu) \Psi_2 \right] \end{aligned}$$

By the way, note the difference between `\overline` and `\bar` in the example above.

4.7 Matrices

There are five matrix generating environments in `amsmath`: `matrix`, `pmatrix`, `bmatrix`, `vmatrix` and `Vmatrix`; these are used as follows (and can generate matrices as large as 10×10)

```
\begin{gather*}
\begin{matrix} 0 & 1 \\ 1 & 0\end{matrix}\quad
\begin{pmatrix} 0 &-i \\ i &0 \end{pmatrix}\quad
\begin{bmatrix} 1 &0\\0 &-1\end{bmatrix}\quad
\begin{vmatrix} a&b\\c&d\end{vmatrix}\quad
\begin{Vmatrix} 1&0\\0&1\end{Vmatrix}
\end{gather*}
```

and generate

$$\begin{matrix}0 & 1\\1 & 0\end{matrix} \quad \begin{pmatrix}0 & -i\\i & 0\end{pmatrix} \quad \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix} \quad \begin{vmatrix}a & b\\c & d\end{vmatrix} \quad \begin{Vmatrix}1 & 0\\0 & 1\end{Vmatrix}$$

Occasionally you may be disappointed with the alignment in a matrix: compare

$$\begin{pmatrix}0 & -1\\2 & 1\end{pmatrix} \qquad \text{with} \qquad \begin{pmatrix}0 & -1\\2 & 1\end{pmatrix}.$$

The right-hand matrix was generated with the following source containing the devilishly cunning `\phantom{...}` command:

```
\begin{pmatrix} 0&-1\\2&\phantom{-}1\end{pmatrix}
```

4.8 Equation Numbering

You may wish to number your equations by section, by chapter, or whatever. Simply create a file called `seceq.sty` (or something similar) and include the following in it:

```
\renewcommand{\theequation}{\thesection.\arabic{equation}}
\@addtoreset{equation}{section}
```

Then load this style file with `\usepackage{seceq}`. In `report` class documents, this automatically numbers your equations as chapter.section.equation.

4.9 Final Remarks

There are a host of different (in)equality signs, and some look better than others, for example:

- `\approx` $\approx$ cf. `\thickapprox` $\thickapprox$ cf. `\simeq` $\simeq$ cf. `\approxeq` $\approxeq$
- `\leq` $\leq$ cf. `\leqslant` $\leqslant$ cf. `\eqslantless` $\eqslantless$ cf. `\leqq` $\leqq$
- `<<` $<<$ cf. `\ll` $\ll$

There are some extra fonts which are useful:

- `\mathbb{...}` for sets etc.: $\mathbb{R}, \mathbb{C}$
- `\boldsymbol{...}` for bold symbols, including Greek: $\boldsymbol{\gamma} \cdot \boldsymbol{\partial}$
- `\mathcal{...}` for caligraphic upper case letters: $\mathcal{L} \sim \mathcal{O}(g^2)$
- `\frak{...}` for Euler Fraktur letters (upper and lower case): $\mathfrak{L}$, $\mathfrak{D}$, $\mathfrak{g}$
- `\pmb{...}` "Poor man's bold" for otherwise unavailable (=random) boldface characters: $\pmb{\oint}$
- I suggest `\mathtt{R\!e}` for $\mathtt{R\!e}\{z\} = x$.

The Fraktur $\mathfrak{L}$ and $\mathfrak{D}$ are good for Lagrangian densities and functional measures; at least I think they look better than the caligraphic ones $\mathcal{L}$ and $\mathcal{D}$.

When you first start LaTeXing, you may end up with lots of spurious indentations around equations, for if you leave a blank line before the `\begin{...}` and/or a blank line after `\end{...}` then LaTeX will assume you wish to begin a new paragraph. The same is true around figure environments and the like.

5 Figures

Pictures can be generated inside LaTeX, for example using such packages as `axodraw`, and appear inside `picture` environments. Alternatively, you can import (encapsulated) postscript into your document. All these appear inside a `figure` environment as follows (remember the American spelling of `center`!)

```
\begin{figure}
\begin{center}
...
\caption{Here is the figure caption.\label{fig_caption}}
\end{center}
\end{figure}
```

Note the position of the `label`: it is tied to the caption. To include postscript files use the `graphics` package, and the command `\includegraphics`:

```
\begin{figure}
\begin{center}
\includegraphics{file.ps}
\end{center}
\end{figure}
```

Should you wish to take only a little part of the file `file.ps`, delimited by the box with lower left-hand coordinates (x, y) and upper right-hand coordinates (x', y'), `\includegraphics` takes a pair of optional arguments:

```
\includegraphics[x,y][x',y']{file.ps}
```

where you must specify the units (which can all be different) for $\{x, y, x', y'\}$.

To rescale postscript files, use the command `\resizebox` which takes three arguments, the width, the height and the object to be rescaled:

```
\resizebox{width}{height}{\includegraphics{file.ps}}
```

Replacing either `width` or `height` with `!` retains the aspect ratio of the figure while scaling to meet your required height/width. The width and height must have units, but note the existence of `\textwidth` which can be used (with or without a multiplicative factor) in the width slot.

Programs such as `Mathematica` often produce really nasty postscript with lots of ugly extra space etc. which can be the reason why you cannot seem to get anything else on the page which contains your graph. Type the following in a command shell (not in your source file)

```
ps2epsi file.ps file.eps
```

which will generate a properly bounded encapsulated postscript file (which can be included with the `\includegraphics` command as above).

5.1 Subfigures

The package `subfigure` allows you to put several different figures into one, and have them individually numbered N(a), N(b), etc., labelled and captioned, and allows an overall caption for the group of figures. This is useful for putting lots of graphs for similar results together. The syntax is as follows:

```
\begin{figure}
\begin{center}
\subfigure[First sub-caption]{\label{first_label}\includegraphics{file1.ps}}
\subfigure[Second sub-caption]{\label{second_label}\includegraphics{file2.ps}}
```

```
...
\caption{The overall caption.\label{overall_label}}
\end{center}
\end{figure}
```

Then the call

```
figure~\ref{first_label}
```

generates "figure N(a)", while `\ref{overall_label}` generates "N" and so on.

6 Tabular Environments and Tables

Tabular environments are for aligning text. The `\begin{tabular}` command takes an argument, which is the alignment of the columns. Columns themselves are separated by the usual `&` symbols, and rows by `\\` carriage returns. For example:

```
\begin{tabular}{|c|l|c|}\hline
Gauge Group & Charge & Gauge Field\\
\hline\hline
SU(3)& Colour & 8 gluons\\
SU(2)& Weak isospin & W$^\pm$, Z\\
U(1) & Electric charge & $\gamma$\\\hline
\end{tabular}
```

generates

Gauge Group	Charge	Gauge Field
SU(3)	Colour	8 gluons
SU(2)	Weak isospin	W$^\pm$, Z
U(1)	Electric charge	γ

Notice how the double `\hline` separates the top from the rest: if you only put one, this will not happen. There is a great amount of flexibility in choosing the layout and presence/absence of lines when building your tablular environment: if you really need a complex table, look up `\cline` and `\multicolumn` in a LaTeX manual. To put the above tabular environment in an article it is best to put it into a properly spaced `table` environment with caption etc. as follows:

```
\begin{table}
\begin{center}
The tabular environment above
\caption{The gauge groups of the standard model \label{tab:ggofsm}
\end{center}
\end{table}
```

Gauge Group	Charge	Gauge Field
SU(3)	Colour	8 gluons
SU(2)	Weak isospin	W$^\pm$, Z
U(1)	Electric charge	γ

Table 1: The gauge groups of the standard model

7 Bibliography

Basic bibliographies are generated with the `\begin{thebibliography}` environment (do not forget to `end` it again before ending your document): this puts a section heading "Bibliography" and then lets you list your references in whatever order you wish them to appear. The command `\cite{cornwall74}` in the body of your document will refer to the `\bibitem` as below:

```
...
\bibitem{cornwall74}
J.~Cornwall, R.~Jackiw, and E.~Tomboulis,
Physical Review \textbf{D 10}, 2428 (1974).
...
```

In this way your document will conclude with a list of `\bibitem`s like this which will form your bibliography. LaTeX will need to be run twice on your document to cite everything properly.

7.1 BIBTeX

A very good way to get LaTeX to create bibliographies for you is by using BIBTeX. Running LaTeX once on your document creates a file `file.aux` in which a record is kept of which citations have been made. Running BIBTeX (with the command `bibtex file` or from within emacs) on your file scans the `.aux` file and looks up the citations from your citations master-file in which you keep records of all the citations you may ever wish to make. BIBTeX then creates a file `file.bbl` which is essentially the bibliography section for your article, ordered correctly and formatted according to the style file which you declare. Running LaTeX again on your file will actually put the contents of `file.bbl` into the generated `dvi` as though it had been part of your source file. When you submit the LaTeX form of an article (e.g. to a journal), you can simply import the `.bbl` file into your document and remove the two commands below which invoke BIBTeX.

This may sound complicated, but it is actually very easy: you never have to worry about the order of your citations, you never have to worry about formatting your bibliography, and most importantly, you only ever have to look up citations once. Once they appear in your references master file, you can cite them whenever you want.

To invoke this bibliography creation you must declare the style file you wish to use for the bibliography; this affects things like the order of volume, page, (date), the use of boldface, the use of italics, the use of *et al* etc. There are a number of style files available over the web or from your office mates, simulating Physical Review, Nuclear Physics, etc. bibliography styles. When you wish your bibliography to appear (normally the last thing in the document), you simply use the commands

```
\bibliographystyle{sty_file}
\bibliography{references_file}
```

which respectively declare the style file you wish to use and the name of your citations master-file (which should have a `.bib` tag). If you have the style file in your home directory somewhere you obviously only want to keep one copy, so refer to `sty_file` with its full path (dropping the final `.sty`), starting from `/home/wytham...`. Or if you are clever you can change your paths so that LaTeX knows where to find it. To create your citations master-file requires a little effort; here is where `emacs` really comes into its own, for it understands the syntax in your master-file as well, and has all sorts of nice features which help with the construction of your bibliography. Particle-people will be pleased to know that the SPIRES database has a BIBTeX function which will print out the entry which you should put into your master references file for any paper since 1975, and for many from before. After a while it is pretty easy to figure them out for yourself. For instance, the entry which appears in my `refs.bib` file which generates the Cornwall 1974 reference above is:

```
@Article{cornwall74,
     author    = "John M. Cornwall and R. Jackiw and E. Tomboulis",
     title     = "EFFECTIVE ACTION FOR COMPOSITE OPERATORS",
     journal   = "Phys. Rev.",
     volume    = "D10",
     year      = 1974,
     pages     = 2428,
     SLACcitation  = "%%CITATION = PHRVA,D10,2428;%%"
}
```

There are also `@Book` types, `eprint` fields and lots more: it is worth playing with BIBTEX and working out how to get it to do what you want. In the end, it is much simpler and quicker to have all your references in one place (you can include your own notes on each paper which BIBTEX will ignore) and to let BIBTEX do the sorting.

8 Errors

DO NOT PANIC.

It is essentially inevitable that LaTeX will generate errors when you first try to compile a document. There are many common errors which occur, and as you get used to things, you will start to spot them more easily.

`emacs` can help a lot, for it can tell you when you do not have matching brackets, dollar signs, etc., and can `end` environments automatically after you `begin` them.

For other errors, you just have to go back and look at your document: no doubt you will have done something strange/silly. Sometimes it is worth letting LaTeX continue if it will, for often the output which it generates makes it much easier to trace your error. However, this can go wrong when one error leads to another, e.g. when you have missed a curly brace delimiting an argument of a command, such as when you type

```
\frac{1}{a lot of old rubbish here
```

and LaTeX tries in vain to put the entirety of the remainder of your document in the denominator of your fraction. Who can blame it for complaining?

It sometimes happens that you make a catastrophic error (which may actually seem quite harmless to you) and after you have corrected it LaTeX will apparently refuse to compile your document ever again, complaining that it does not even understand the command `\begin{document}`. Do not worry: simply remove the `.aux` file which accompanies your source and run LaTeX again.

One last thing: LaTeX regularly complains about lines being too long, i.e. when it complains about an `Overfull \hbox`. This can be an equation that needs wrapping, or simply a bunch of words it does not know how to hyphenate. Although LaTeX will happily compile such documents, they look terrible, so do try to sort out the problem. LaTeX does not know how to hyphenate some scientific words, and you can teach it to do so as and when required with the "you may put a hyphen here if you want" command `\-`, as in `renormal\-ization`. You only need to do this as and when problems arise: LaTeX copes quite happily most of the time. For equations, sometimes you simply have to accept that your wonderful mathematics has to be broken over two lines, and include some carriage returns. After all: do you want to do it yourself, or are you willing to let the EDITOR do it?
DO NOT PANIC.

9 Processing and Viewing Your Document

To compile your document, `source.tex`, run the following command (the commands in this section are to be run in an ordinary command shell):

```
latex source
```

(The `.tex` tag is implied: if you are perverse and have not used this then you will have to tell LaTeX explicitly what you are talking about.) This will generate all sorts of other files, `.log` files, `.aux` files, etc.: do not worry about them. If you have citations, references etc., then LaTeX will initially complain that you have all sorts of undefined whatnots. Run LaTeX again and only then will it stop complaining and cross-reference everything properly (assuming you have made no mistakes). If you are using BIBTeX then run

```
bibtex source
latex source
latex source
```

after the original LaTeX run. Do not worry: you do not need to run LaTeX three times every time you compile, but if you change your citations, it will take the above sequence to get it right. In the course of making a document, you will run LaTeX many many many times, and the citations, references, etc. will catch up with themselves; there is no point in running `bibtex` unless you have changed your citations.

All this LaTeXing will generate a `source.dvi` which can be viewed with

```
xdvi source &
```

(the ampersand runs `xdvi` in the background so that you can still use your window). If you wish to convert to postscript then use

```
dvips -o source.ps source.dvi
```

and you can then view the postscript with `ghostview` or print with `lpr`. Note that `dvips source` (i.e. without the option `-o`) will print without cluttering up your home directory with postscript. Note further that there are some things (`axodraw` figures, for instance) which only appear in the final postscript (which you can view or print).

To print on one side of the paper only in this department, use

```
lpr -Z simplex source.ps
```

Presumably the `Z` stands for "Zen" or something similar; nevertheless, it works.

10 Conclusion

Finally I should just like to say that LaTeX provides many different solutions to the same typesetting problems: it is fun to try to find the right way, the most elegant way, the way which impresses your office-mates the most, and so on. It is really worth scanning a LaTeX guide, or better, a *Companion*, and finding the right way to do things. Alternatively, an excellent local web-site:

`www-thphys.physics.ox.ac.uk/users/JennyWilliams/home.html`

in which there is much about LaTeX, and BIBTeX in particular.

11 The End

Do not forget to end your document:

```
\end{document}
```