Typesetting Mathematics with LaTeX

Christopher Hollitt

April 18, 2018

1 Introduction

One of the great strengths of Latex is the ease with which it manages complex mathematical formulæ. While WYSIWYG alternatives have improved over the last few years, Latex is still far more powerful and (arguably) produces superior results.

Latex has an enormous range of mathematical features. Most of the beginner's guides to Latex include a section of mathematical typesetting. The wikibooks Latex book (http://en.wikibooks.org/wiki/LaTeX) for example, includes an excellent introduction on its "Mathematics" and "Advanced Mathematics" pages. This document does not seek to replicate any of this introductory material – it is just intended to present a few helpful tips.

If you are typesetting mathematical material you will almost always want to use a couple of additional mathematical packages provided by the American Mathematical Society (AMS). While Latex alone is a powerful mathematical typesetting system, the addition of the AMS packages is extremely useful. To include the required packages, put the following at the start of your main .tex document.

```
\usepackage{amsmaths}
\usepackage{amssymb}
```

Anyone contemplating typesetting mathematical material using Latex should read through the amsmath package documentation, which is available at ftp://ftp.ams.org/pub/tex/doc/amsmath/amsldoc.pdf

The amssymb package is useful as it contains several commonly used mathematical characters such as $\therefore$. If you don't use those characters then

you obviously don't need to use the package. However, there is little cost to including it, so we suggest using the package by default.

There are many specialised packages available that might simplify various tasks. If you find a helpful package then feel free to use it.

You may find yourself searching for the latex codes for special characters. At such times the comprehensive Latex symbols list (http://ctan.unsw.edu.au/info/symbols/comprehensive/symbols-a4.pdf) may be of use. The list is rather daunting as it contains nearly 6000 symbols! Alternatively you can try the detexify tool (http://detexify.kirelabs.org/classify.html) that attempts to guess at a Latex character from you hand-drawn approximation.

There are many other guides available on the web that describe use of Latex's mathematical typesetting facilities. One example is at ftp://ftp.ams.org/ams/doc/amsmath/short-math-guide.pdf
Be careful when reading such guides, as some are out of date and suggest the use of constructs that are no longer the best option. If you begin by reading a general introduction and then the ams documentation then you should be in a good position to extract the useful material from other guides.

1.1 General advice

- Avoid using `\eqnarray`, as `\align` is much better behaved.
- Unless you really must, don't use the `array` environment to typeset matrices - the `pmatrix` environment and its ilk from the amsmath package are far superior.
- Make liberal use of the `\align` environment to keep series of equations neat and tidy. You can use `\intertext` if you want to interrupt a long series of equations to make a brief comment. This ensures that the equations above and below the comment remain aligned.
- Use `\left(` and `\right)` and their brethren to ensure that delimiters automatically scale to contain their contents.
- Define macros for mathematical sequences that you use often. For example, you could use

```
\newcommand{\lapint}[1]{\int_{0^-}^{\infty} #1
                                \mathrm{e{^{-st}\, \mathrm{d}t}
```

to define a command for the integral that defines the Laplace transform. (The above should be written in one line - I have inserted the line break to fit it onto the page.) With the command definition in place, you could write `$F(s)=\ilap{f(x)}$` to get the equation $F(s) = \int_{0^-}^{\infty} f(x)\mathrm{e}^{-st}\mathrm{d}t$.
If you have only a small number of macros then you could put them in your main latex file. Otherwise develop a style file containing your macros.

- The default `\vec` command used to specify vector quantities results in a variable with an arrow above it: `\vec{x}` results in $\vec{x}$. In engineering it is more typical to use bold face italics to indicate a vector quantity, so you should redefine the `vec` command using
`\renewcommand{\vec}[1]{\boldsymbol{#1}}`.
`\vec{x}` would then result in $\boldsymbol{x}$. If you are dealing with matrices then you could similarly define a `\mat` command (or some other similar name) to format matrices:
`\renewcommand{\mat}[1]{\boldsymbol{#1}}`.

- If you are typesetting significant amounts of mathematical material, then ask your supervisor if they have a style file or other appropriate set of macros that can make your life easier. Such style files are often idiosyncratic, so use of such style files may or may not save you time...

2 The Finer Points

This section will describe some stylistic conventions that you should follow and some common errors to be avoided.

2.1 Equation Numbering

Unless your supervisor tells you otherwise, you should number all of your equations. While *you* may not need to refer to each of your equations, your

proofreader or examiners may wish to. Consequently avoid the "starred" version of the mathematical environments (eg use `align`, not `align*`). Similarly you should use,

```
\begin{equation}
  a=3
\end{equation}
```

rather than `\[ a=3 \]` to ensure that you get an equation number.

Use labels to give an equation a descriptive name that you can then use later when referring to the equation. (A common practice is to begin equation labels with "eqn:" to distinguish them from other types of labels.) *Never* explicitly type something like "In Equation 6.10 we see that", because you are guaranteed to lose track of the equation number as you edit your text. Instead, write

```
\begin{equation}
  x=3 \label{eqn:x_is_three}
\end{equation}
Equation \eqref{eqn:x_is_three} tells us that $x$ is always
equal to three.
```

to get:

$$x = 3 \tag{1}$$

Equation (1) tells us that x is always equal to three.
Notice the use of `\eqref` rather than `\ref` to reference an equation. The `\eqref` form simply inserts parentheses around the equation number for you.

2.2 Font

Latex switches to a set of special mathematical fonts whenever you enter a mathematical environment. If you have redefined the `\vec` and `\mat` commands as described above, then this will help ensure that your typesetting will comply with the international standard for the typesetting of mathematical material, ISO 80000-2 (which superseded ISO31-11 in 2009). Latex has some minor deviations from the standard, but in general is pretty good. Unfortunately the standard itself is not freely available, but there is a good

accessible article on ISO31-11 at
http://www.tug.org/TUGboat/tb18-1/tb54becc.pdf, which describes the important features.

In particular you should notice that mathematical variables inside a mathematical environment are typeset in italics. It is important to maintain the mathematical typesetting whenever you refer to that variable in the body of your text. For example, it we have an equation

$$x = 2\hbar$$

and we wish to discuss the variable x, then we should write
"where x is the length of the plank" (by using `$x$` in your .tex file).
Not
"where x is the length of the plank" (by just using `x` in your .tex file).

Conversely, you should ensure that *only* variables are typeset in italics when using mathematical mode. This can sometimes go wrong when you are using functions, the names of which should always be typeset in an upright roman font. Latex knows about many common functions, so can ensure that they are typeset correctly. For example, if you typeset sin using `\sin(\theta)` then you correctly get $\sin(\theta)$. If you forget the backslash character before the sin (`sin(\theta)`) then you will get $sin(\theta)$. Notice that sin is incorrectly typeset in italics, because latex thinks that "s","i" and "n" are variables.

When Latex doesn't know about a mathematical function then you can rectify the problem using the `DeclareMathOperator` command:
For example we could define "trace" to be a function using
`\DeclareMathOperator{\trace}{Tr}`.
Then `$\trace(\mat{A})$` will produce $\mathrm{Tr}(\boldsymbol{A})$, with the trace function correctly typeset in upright roman font. If you want to avoid declaring a new mathematical operator for your whole document, then you can instead use the `\operatorname` construct:
`$\operatorname{adj}(\mat{A})$` results in $\operatorname{adj}(\boldsymbol{A})$, where the adjugate function is correctly upright. Of course, if you use this approach you would need to include all of the operatorname syntax every time you used the adjugate function.

Variables that have subscripts (or superscripts) that are English words or abbreviations of English words should be typeset in roman font, not italics. So for example, you should write `$G_{\mathrm{robot}}$` so that you get

G_{robot}, *not* `$G_{robot}$` which results in G_{robot}. Similarly you might write K_{p} for proportional gain (the "p" in this case being short for proportional).

A subscript that refers to another variable should be typeset in italics. For example the specific heat of a material for constant pressure would be written C_p, not C_{p}. This is because volume can take a certain value (it is a variable).

Units are not variables, so should never be typeset in italics. They should *always* be in upright font and separated from the expression/number by a space. As an example, `$5d mm$` results in $5dmm$ (which is bad), but `$5d$mm` results in $5d\,\mathrm{mm}$ (which is good). While the `$5d$mm` syntax automatically leaves a space between the equation and its associated units, it is a good idea to use a non-breaking space character `~` instead. This ensures that matlab doesn't embarrassingly insert a line of page break between an expression and its associated units: `$5d$~mm`. Some people prefer to use a thin space instead `$5d$\,mm` ($5d\,\mathrm{mm}$), but then you lose the non-breaking functionality.

Strictly speaking, mathematical constants such as e, j, π and d (in derivatives and integrals) should be typeset in an upright font. However, this rule is commonly ignored in practice. If you feel so inclined you can use `$\mathrm{e}$` (e), `$\mathrm{j}$` (j) and `$\mathrm{d}$` (d). (These are excellent candidates for a `\DeclareMathOperator`). Unfortunately, it is difficult to typeset π in the striclty correct font, as Latex does not have good support for upright lower-case Greek letters.

2.3 Extraneous Paragraph Breaks

A very common error is to insert a blank line between an equation and subsequent text. This causes Latex to begin a new paragraph and mess up the indentation of the following text. Contrast the indentation of the text following the equation in the following two examples. The second version is the desired layout.

$$\begin{aligned} & x = 2d \\ \implies & x = 7 \end{aligned}$$

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

$$\begin{aligned} & x = 2d \\ \implies & x = 7 \end{aligned}$$

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

To avoid this problem you need to make sure that there is no blank line between the end of your equation block and the subsequent text. That is, to ge the correct version above you would use

```
  \begin{align*}
  x&=2d \\
  \implies x&=7
\end{align*}
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.
```

2.4 Punctuating Equations

Equations are treated just like words for the purposes of punctuation. For example, if a sentence ends with an equation then there should be a full stop after the equation. In the case of inline equations this is fairly natural. For example "`Solving the quadratic equation results in $x=3\pm4\mathrm{i}$.`" gives:
Solving the quadratic equation results in $x = 3 \pm 4\mathrm{i}$.

For display like environments things get a bit trickier, as you need to include the punctuation inside the display to prevent it being pushed to the following line. For example

```
The quadratic equation can be written as
\begin{equation}
  x=\frac{b\pm\sqrt{b^2-4ac}}{2a} \text{\quad .}
\end{equation}
```

results in:
The quadratic equation can be written as

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad .$$

Notice that the full stop has been included inside the equation environment and we have used a `\quad` command to add some horizontal space between the equation and the full stop.

Some authors feel the need to insert a colon into the middle of a sentence when introducing an equation. In the example above they would put a colon (or maybe a semicolon) after "as". This is not required, so fight the urge!