Course A, Part 2
More Basic Formatting in LaTeX

Now that you know a thing or two about LaTeX, let's see how to typeset more complicated mathematics. Again, this will not be all that you'll ever need to know, but it should get you pretty far. Today's lesson will concentrate mainly on mathematics, not so much on text.

Operators
In the language of LaTeX, an *operator* is a symbol that usually takes some decoration, such as

$$\int_a^b, \quad \sum_{k=1}^{\infty}, \bigoplus_{a\in A}, \text{ or } \bigcup_{i=1}^{n}.$$

The operators above are given by `\int`, `\sum`, `\bigoplus`, and `\bigcup` respectively. These always use an underscore for the lower decoration and a caret (`^`) for the upper. For example, to set

$$\int_a^b f(x)\ dx$$

I coded

```
\[
\int_a^b f(x) \ dx
\]
```

It would work the same way in-line, except that the integral symbol would be typeset very small so that it fits in-line. For example, this $\int_a^b f(x)\ dx$ is what the same integral would look like in-line. If you want to set it in-line in display-size, you have to start your math string with the `\displaystyle` command. WARNING: this usually looks ugly.

More complicated decorations work exactly the same way, being careful with your curly braces. So to get

$$\int_{\sqrt{t+1}}^{t^2} f(x)\ dx$$

I just set[1]

```
\[
\int_{\sqrt{t+1}}^{t^2}  \ f(x) \ dx
\]
```

Setting Matrices
This is really easy. There are tons of ways to typeset matrices, all having slightly different effects. WinEdt users may notice the "insert → matrix" option above the toolbars; in my opinion, this is *not* the best way to go. The easiest option giving the nicest-looking matrices is the `pmatrix` environment. Entries are separated by ampersands (`&`) and rows are broken with a double backslash (just like a linebreak). This is a math command (unlike `\begin{center}`) so it must occur inside a math environment. For example, coding

[1] Note the use of buffers in these integrals.

```
\[
\begin{pmatrix}
-2 & x^2 & x\\
a+b & c & 4
\end{pmatrix}
\]
```
produces

$$\begin{pmatrix} -2 & x^2 & x \\ a+b & c & 4 \end{pmatrix}.$$

Other methods of typesetting matrices may not automatically center the entries, but may still be worthwhile for other reasons. Refer to a manual for more assistance.

NOTE: In the above, I did not use a linebreak on the last row. This is fairly common in many LaTeX subsidiary environments like `pmatrix`. In fact, many commands will give an error if you use a linebreak on the last row! It is best to get in the habit of *not* breaking final rows—if the command actually wants you to, it will tell you during compilation.

Piecewise Functions

If you ever teach algebra or calculus, this will be indispensable. When you're ready to define the "pieces" of your function, start with `\begin{cases}`. The rest of the coding acts just like `pmatrix`. As an example,

$$g(x) = \begin{cases} \sin x, & x > 0 \\ x^3, & x \leq 0 \end{cases}$$

is the result of
```
\[
g(x) = \begin{cases} \sin x, & x > 0 \\ x^3, & x \leq 0 \end{cases}
\]
```

Aligning Equations, Tagless

This section describes how to align equations *without* auto-generated tags (that will be described next). This is the way to go for small documents or equations that aren't used elsewhere in your document.

The alignment process starts with `\begin{align*}`. The asterisk is very important—it indicates that you do *not* want LaTeX to number your equations automatically (you can still tag them manually, but we won't get into this). Enter your equations, and place an ampersand (`&`) to the left of the symbol that should serve as the alignment point. Break lines in the usual way, except for the last row. To align a couple of equations, as in

$$\begin{aligned} x^2 + y^2 &= z^2 + 1 \\ u^2 - v^2 &= 2 \end{aligned}$$

just do

```
\begin{align*}
x^2 + y^2 & = z^2 + 1\\
u^2 - v^2 & = 2
\end{align*}
```

WARNING: LaTeX knows that the `align*` environment is for setting mathematical expressions, so you should *not* enter this inside a math environment (i.e., with `$`'s or `\[`'s). Otherwise, an error may result.

Because I put the ampersand to the left of the equal signs, it aligns the equations by the equal signs. To center a multi-line group of symbols that are *not* equations, use the `gather*` environment instead (deleting the asterisk if you want to gather with auto-generated tags).

Aligning an entire vertical chain of equations follows the same format. Just do an ordinary alignment, but the first entry on subsequent rows should be left blank. For example, to get

$$\begin{aligned} f(x) &= (x+3)^2 \\ &= x^2 + 6x + 9 \end{aligned}$$

I coded

```
\begin{align*}
f(x) & = (x+3)^2\\
& = x^2 + 6x + 9
\end{align*}
```

Note that there is nothing preceding the ampersand in the second equation. Moreover, there is nothing sacred about equal signs: you can align along anything.

Aligning Equations, With Tags

Sometimes you'll want to refer to previous equations or derivations in your document. It's simpler (and more professional) to refer to a numbered equation than to regurgitate the whole thing in front of the reader. This is called "tagging." I'm only going to explain auto-tagging, where LaTeX does all the work and you have no control over the tags. If you want to customize your tags, look this up in a decent LaTeX reference book (it's not hard at all).

Let's look back at the first example in the tagless alignment section. If you want LaTeX to label those equations automatically, just drop the asterisks from the `align*` declaration: so

$$x^2 + y^2 = z^2 + 1 \tag{1}$$
$$u^2 - v^2 = 2 \tag{2}$$

was produced by

```
\begin{align}
x^2 + y^2 & = z^2 + 1\\
u^2 - v^2 & = 2
\end{align}
```

Subsequent equations in your paper will be automatically numbered in sequence by LaTeX! If in the editing process you insert or remove some equations, LaTeX will automatically adjust the tags so that everything is still in sequence. Absolutely amazing. Wanna see? Look:

$$f(x) = (x+3)^2 \tag{3}$$
$$= x^2 + 6x + 9. \tag{4}$$

Sometimes you don't want every single line to be tagged. This is simple to fix: just place `\notag` right before you end the line. For example, to get

$$\begin{aligned} g(x) &= (x-4)^2 \\ &= x^2 - 8x + 16 \end{aligned} \tag{5}$$

I coded

```
\begin{align}
g(x) & = (x-4)^2 \notag\\
& = x^2 - 8x + 16
\end{align}
```

Now, quite honestly, all of this is useless if you do not have an efficient way of referring back to the equation. You could do this "manually," just typing lines in text like

By referring to equation (5) above....

This is a horrible idea: do you see why? What would happen if you inserted or deleted some equations? What if your document was a 150-page Ph.D. thesis?

This is why the brilliant people behind LaTeX created the notion of a *label*. If you attach a label to a line, you may refer back to that label, and LaTeX will automatically insert the correct tag. Even when you re-edit, LaTeX will re-compile the tags and references to match up correctly. Amazing.

For example, suppose I want to refer back to the equations in

$$X \oplus Y = A \cap B \tag{6}$$

$$X^* \oplus Z = W \times Z. \tag{7}$$

You create a label with `\label{your label name}`, placing it right before the end of the line you wish to label. I obtained the above by

```
\begin{align}
X \oplus Y & = A \cap B \label{eqn1}\\
X^* \oplus Z & = W \times Z. \label{eqn2}
\end{align}
```

General Tip: Your label names should be descriptive, just for the sake of convenience. (OK, so my label names above aren't so descriptive, but this is just an example.)

To refer to a label in the text environment, just refer to it as `\eqref{your label name}` and LaTeX will take care of the rest. For example, to get

By a careful inspection of (6) above, we see that (7) follows.

I simply wrote

```
By a careful inspection of \eqref{eqn1} above, we see that \eqref{eqn2} follows.
```

Now if you insert or delete some equations, LaTeX will re-number them in sequence and it will also replace your references with the updated tags!

Note: There are other kinds of tags and labels, and these would *not* be called up with `\eqref`. Consult a good LaTeX book for more information.

Bonus Tip!

Ever get tired of typing the same long, laborious command string again and again? Me too. Well, you can make your own customized shortcuts!

I teach Multivariable Calculus or Linear Algebra nearly every semester, so I am constantly typesetting vectors, like $\mathbf{u} = \langle 4, -6, 8 \rangle$. The way LaTeX processes that line is with the code

```
\mathbf{u} = \langle 4, -6, 8 \rangle.
```

The `\mathbf` command is for bold-faced fonting of math symbols, while `\langle` is short for "left angled bracket." (I'm betting you can guess what `\rangle` is short for.) I'm too lazy to type all that. Here's what I actually code to produce the same line:

```
\vu = \la 4, -6, 8 \ra.
```

Much simpler. But how did I do that? I have defined a shortcut called `\vu` which replaces the string `\mathbf{u}`. I use `\vu` because I read it as "vector u." To do this, I simply placed

```
\newcommand{\vu}{\mathbf{u}}
```

in the header of my document. Similarly, I defined `\la` to be a shortcut for `\langle` with

```
\newcommand{\la}{\langle}.
```

You get the idea now: the syntax for a shortcut is

```
\newcommand{\shortcut_name}{\actual_code_string}.
```

This comes with the obvious rule: you can't use a shortcut name that is already defined in LaTeX, for this would create a contradiction.