Hardcore LaTeX Math

RSI 2011 Staff

June 18, 2011

Part I: basic features

- Review
- Theorems and proofs such

Entering math mode

Use `$...$` or `\(...\)` for text math mode and `\[...\]` or `$$...$$` for display math mode:

```
This sentence refers to integers $n$ and \(m\) and
reminds you that
\[ \int_0^\infty e^{-t} dt = 1. \]
```

This sentence refers to integers n and m and reminds you that

$$\int_0^\infty e^{-t} dt = 1.$$

It's probably better to use `$...$` than `\(...\)`, because dollar signs are more traditional, and thus easier to read. But for displayed equations, `\[...\]` is preferable.

Punctuation

Inline equations should not contain punctuation belonging to the sentence.

```
Because the ideal $I$ contains $x$, ...
```

Displayed equations that end sentences should contain a period.

```
Therefore, the value of the integral is
\[ \int_0^\infty e^{-t} dt = 1. \]
```

Numbered equations

Use the `equation` environment to create a numbered equation. Put a label inside it using `\label{foo}` and then use `\ref{foo}` to typeset the equation number. *Preferred:* You can type `\eqref{foo}` to add parentheses automatically.

```
It follows that
\begin{equation}
\label{verywrong}
6 \times 9 = 42.
\end{equation}
From \eqref{verywrong} we can deduce we screwed up somewhere.
```

It follows that

$$6 \times 9 = 42. \tag{1}$$

From (1) we can deduce we screwed up somewhere.

Variables

For most variables with one-character names, just type the character in math mode: `$x + y$` for $x + y$.

For Greek letters, backslash their names: `$\alpha, \beta, \Gamma$` for α, β, Γ. (There is no `$\Alpha$`; simply use `$A$`.)

For long names, there are two tasteful options:

```
\[ \mathit{long\_variable\_name} + \text{roman name} = 240 \]
```

$$\mathit{long\_variable\_name} + \text{roman name} = 240$$

Don't just type a long name into math mode or you'll get

$$long_variable_name.$$

Variables in other fonts

$\mathbf{A}$	`\mathbf{A}`	
$\mathcal{F}$	`\mathcal{F}`	(capitals only)
$\mathbb{Z}$	`\mathbb{Z}`	(capitals only)
$\mathfrak{ab}$	`\mathfrak{ab}`	
$\mathscr{L}$	`\mathscr{L}`	(capitals only; requires `mathrsfs` package)

Put only the variable inside the font-changing command:

```
\[ \mathfrak{c}(\mathfrak{a} + \mathfrak{b}) \]
```

$$\mathfrak{c}(\mathfrak{a} + \mathfrak{b})$$

Roman operators

Many notations like sin(x) and log(x) are typeset in upright characters. Say `\sin(x)`, `\log(x)`, etc. For comparison, `sin(x)` produces $sin(x)$, which is italicized.

For the condition under `\lim`, `\max`, or the like, use a subscript:

```
\[ \lim_{x\to \infty} 1/x = 0 \]
```

$$\lim_{x\to\infty} 1/x = 0$$

(Note that subscripts and superscripts which are more than one character need brackets. For instance, `10^10` appears as 10^10.)

Defining your own operators

To define your own, say

```
\DeclareMathOperator{\spec}{Spec}
```

in `preamble.tex` and then use `$\spec A[x]$` for $\operatorname{Spec} A[x]$. You can also type `$\operatorname{Spec} A[x]$` to get the same result without defining a command.

Big operators

For big operators such as sums (`\sum`), products (`\prod`), and integrals (`\int`), use subscripts and superscripts to get the limits:

```
\[ \int_{1}^{\infty} \frac{dx}{x^2} = 1. \]
```

$$\int_{1}^{\infty} \frac{dx}{x^2} = 1.$$

```
\[ \sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}. \]
```

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}.$$

Theorems and such

After a couple of declarations, you can do this:

```
\begin{theorem}
Herding cats is hard.
\end{theorem}
```

Theorem 1. *Herding cats is hard.*

```
\begin{lemma}[Smith, 1972]
Herding alligators is harder,
and much more dangerous besides.
\end{lemma}
```

Lemma 1 (Smith, 1972)**.** *Herding alligators is harder, and much more dangerous besides.*

Creating theorem types

Before using `theorem` and `lemma`, I had to do this:

```
\newtheorem{theorem}{Theorem}
\newtheorem{lemma}{Lemma}
```

The first argument is the internal name, so the word you put in the `\begin{}` and `\end{}` statements. The second argument is the external name, so it appears in the document before the number.

Each theorem type has its own counter (hence Theorem 1 and Lemma 1 in the previous example).

Put these declarations in `preamble.tex`.

Several types, one counter

```
\newtheorem{prop}{Proposition}
\newtheorem{propdef}[prop]{Proposition-Definition}
```

The optional argument `[prop]` says that Proposition-Definitions should be numbered in the same series as Propositions.

```
\begin{prop}[Lucas, 1977]
Swordplay obviates acting.
\end{prop}
\begin{propdef}[Lucas, 1999]
There exist painfully inane characters.  A maximal such
character in a film is \emph{of Jar-Jar type}.
\end{propdef}
```

Several times, one counter (continued)

Proposition-Definitions and Propositions are *not* numbered separately:

Proposition 1 (Lucas, 1977)**.** *Swordplay obviates acting.*

Proposition-Definition 2 (Lucas, 1999)**.** *There exist painfully inane characters. A maximal such character in a film is* of Jar-Jar type*.*

Note also that we used `emph` instead of `textit`.

Counting within sections, subsections, etc.

You can do

```
\newtheorem{remark}{Remark}[section]
```

to make Remarks numbered separately within each section ("Remark 5.3" for the third Remark in Section 5).

Similarly `\newtheorem{remark}{Remark}[subsection]` would make "Remark 5.2.1" and the like.

More with theorems: styles and numbering

You can use three built-in “styles” to specify how you want your theorem heading to appear. You can also define your own if these aren’t enough. Additionally, using an * makes a theorem type unnumbered.

```
\theoremstyle{plain}% default
\newtheorem{thm}{Theorem}[section]
\newtheorem{lem}[thm]{Lemma}
\newtheorem{prop}[thm]{Proposition}
\newtheorem*{cor}{Corollary}
\newtheorem*{KL}{Kleins Lemma}
```

More with theorems (continued)

```
\theoremstyle{definition}
\newtheorem{defn}{Definition}[section]
\newtheorem{conj}{Conjecture}[section]
\newtheorem{exmp}{Example}[section]

\theoremstyle{remark}
\newtheorem*{rem}{Remark}
\newtheorem*{note}{Note}
\newtheorem{case}{Case}
```

Proof environment

The proof environment automatically places a QED symbol at the end of your proof:

```
\begin{proof}[Proof of the Main Theorem]
Combining Lemma~\ref{lem:smith} and Definition~\ref{def:smith},
we see that $\mu(x)=17$.
\end{proof}
```

Proof of the Main Theorem. Combining Lemma 1 and Definition 3, we see that $\mu(x) = 17$. □

For some reason, `amsmath` doesn't appear to contain the proof environment, so you might need to add `\usepackage{amsthm}` to `preamble.tex`.

Moving the QED

Sometimes, the box appears on an empty line:

Proof.

$$G(t) = L\gamma!\,t^{-\gamma} + t^{-\delta}\eta(t)$$

□

```
\begin{proof}
\begin{equation*}
G(t)=L\gamma!\,t^{-\gamma}+t^{-\delta}\eta(t) \qedhere
\end{equation*}
\end{proof}
```

Proof.

$$G(t) = L\gamma!\,t^{-\gamma} + t^{-\delta}\eta(t) \qquad \square$$

Part II: breaking and stacking

- `multline`, for breaking lines in long equations
- `gather` and `align`, for several equations in one display
- `pmatrix` and variants, for matrices
- `\substack`, for stacking conditions under big operators

Breaking lines in displayed equations

Don't try to use \\ to force a line break in a displayed equation. It won't work. Instead, use the `multline` environment:

```
\begin{multline}
X = a + b + c + d + e + f + g\\
+ h + i + j + k + l
\end{multline}
```

$$X = a + b + c + d + e + f + g + h + i + j + k + l + m \\ + n + o + p + q + r + s + t + u \quad (2)$$

The variant `multline*` gives an unnumbered equation.

In a display, always break the line *before* a binary operator.

More lines in `multline`

More than two lines are allowed. The first is flushed left; the last is flushed right; the others are centered.

```
\begin{multline*}
X = a + b + c + d + e + f + g + h + i + j + k + l + m\\
 + n + o + p + q + r + s + t + u + v\\
 + w + x + y + z + 1 + 2 + 3 + 4 + 691
\end{multline*}
```

$$X = a + b + c + d + e + f + g + h + i + j + k + l + m$$
$$+ n + o + p + q + r + s + t + u + v$$
$$+ w + x + y + z + 1 + 2 + 3 + 4 + 691$$

Rejustifying `multline`

Put a line in `\shoveleft{...}` or `\shoveright{...}` to change the justification.

```
\begin{multline*}
X = a + b + c + d + e + f + g + h + i + j + k + l + m\\
 \shoveright{+ n + o + p + q + r + s + t + u + v}\\
 + w + x + y + z + 1 + 2 + 3 + 4 + 691
\end{multline*}
```

$$\begin{multline*}
X = a + b + c + d + e + f + g + h + i + j + k + l + m\\
 \shoveright{+ n + o + p + q + r + s + t + u + v}\\
 + w + x + y + z + 1 + 2 + 3 + 4 + 691
\end{multline*}$$

Several equations in one display

You can use the `gather` (numbered) or `gather*` (unnumbered) environments to put several equations in a display, each centered independently of the others.

```
\begin{gather}
\label{foo} x + y = z\\
\label{bar} a = b + c + d + e + f
\end{gather}
```

$$x + y = z \tag{3}$$

$$a = b + c + d + e + f \tag{4}$$

But it is usually more tasteful to align your equations.

Aligned equations

To align several equations in one display, use `align` (numbered) or `align*` (unnumbered).

```
\begin{align}
f(x) &= g(x^2)\\
a + b + c &= d + e + f
\end{align}
```

$$f(x) = g(x^2) \tag{5}$$

$$a + b + c = d + e + f \tag{6}$$

Multicolumn `align`

You can have several aligned columns; use extra ampersands to separate them.

```
\begin{align}
x + y &= z + w & a + b &= c + d\\
X + Y &= Z + W & A + B &= C + D
\end{align}
```

$$x + y = z + w \qquad a + b = c + d \tag{7}$$
$$X + Y = Z + W \qquad A + B = C + D \tag{8}$$

Align: making only some lines numbered

Option 1: Use `\begin{align}` with `\notag` just before the `\\` on the lines you do not want numbered.

Option 2: Use `\begin{align*}` with `\tag{}` just before the `\\` on the lines you do want numbered.

```
\begin{align*}
x+1 &= 18 \\
x &= 17 \tag{1}\label{result}
\end{align*}
We can see from \eqref{result} that\ldots
```

$$\begin{aligned} x+1 &= 18 \\ x &= 17 \end{aligned} \tag{1}$$

We can see from (1) that. . .

Cases constructions

```
\begin{equation}
F_{n} = \begin{cases}
  0 & \text{if $n = 0$;}\\
  1 & \text{if $n = 1$;}\\
  F_{n - 1} + F_{n - 2} & \text{if $n > 0$.}
\end{cases}
\end{equation}
```

$$F_n = \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F_{n-1} + F_{n-2} & \text{if } n > 1. \end{cases} \tag{9}$$

Matrices

```
\[ \begin{pmatrix}
   2 & 3 & 4\\
   5 & 6 & 7\\
   8 & 9 & 10 \end{pmatrix} v = 0 \]
```

$$\begin{pmatrix} 2 & 3 & 4\\ 5 & 6 & 7\\ 8 & 9 & 10 \end{pmatrix} v = 0$$

For square brackets use `bmatrix`. For single or double vertical lines use `vmatrix` or `Vmatrix` respectively.

For small matrices in running text like $\bigl(\begin{smallmatrix} 2 & 3 \\ 4 & 5 \end{smallmatrix} \bigr)$:

```
$\bigl( \begin{smallmatrix} 2 & 3 \\
4 & 5 \end{smallmatrix} \bigr)$
```

The \substack command

\substack lets you stack expressions in subscripts.

```
\[ \min_{\substack{ x \in S \\ x \ge 20 }} f(x) \]
```

$$\min_{\substack{ x \in S \\ x \ge 20 }} f(x)$$

```
\[ \sum_{\substack{ a + b + c = 20 \\
     a,b,c \ge 0 \\ a + 2b \ge 5 }} abc \]
```

$$\sum_{\substack{ a + b + c = 20 \\ a,b,c \ge 0 \\ a + 2b \ge 5 }} abc$$

Part III: the fine points of looking good

- Math in text
- Dots
- Big delimiters
- Spacing tricks
- Common ugly constructions

Putting it all in math mode

A common mistake:

```
We see that $x$ $+$ $y$ $=$ $z$ $+$ $23$.
```

We see that x $+$ y $=$ z $+$ 23.

Put the *entire equation or expression* in math mode:

```
We see that $x + y = z + 23$.
```

We see that $x + y = z + 23$.

Punctuation around formulae

In *text math mode,* commas and periods that are part of the sentence should be outside math mode:

```
Because $f(a,b) = 0$, it follows that $f(2a, b-a) = 4$.
```

Because $f(a, b) = 0$, it follows that $f(2a, b - a) = 4$.

In *display math mode,* there should (usually) be a comma, semicolon, or period *inside* the display.

```
We find therefore that \[ x^{n} + y^{n} \ne z^{n}. \]
```

We find therefore that

$$x^n + y^n \neq z^n.$$

Dots

The AMS packages have introduced convenient abbreviations for different kinds of dots in math mode. This is a good idea, since then it is possible to change all of the style consistently throughout a document by adding a single line to the preamble (instead of doing a search and replace potentially across several files). Here they are:

- `\dotsc` for dots with commas, as in `$a_1, \dotsc, a_n$` for $a_1, \dotsc, a_n$;

- `\dotsm` for dots with multiplication, as in `$1 \cdot 2 \dotsm n$` for $1 \cdot 2 \dotsm n$;

More dots

- `\dotsb` for dots with binary operators or relations, as in `$1 + \dotsb + n$` for $1 + \dots + n$;

- `\dotsi` for dots with integrals, as in `\[ \int_A \int_B \dotsi \int_Z \]` for

$$\int_A \int_B \dots \int_Z;$$

- `\dotso` for "other."

Even more dots

- In text mode just use `\dots`; it'll make something like this. . .
- To force low horizontal dots in math mode, use `\ldots`.
- To force centered horizontal dots in math mode, use `\cdots`.
- In matrices with rows or columns left out, you may need `\vdots` ($\vdots$) and `\ddots` ($\ddots$) as well.

Dots in a matrix

```
\[ \begin{matrix}
a_{11} & \cdots & a_{1n} \\
\vdots & \ddots & \vdots \\
a_{n1} & \cdots & a_{nn}
\end{matrix} \]
```

$$\begin{matrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{matrix}$$

Delimiter height, part 1: automatic expansion

Using `\left` and `\right` in front of delimiters (parentheses, brackets, etc.) makes them expand to enclose a tall expression (fraction, sum, integral, etc.).

Thus `$\left(\frac{x^2}{y^2}\right)$` gives $\left(\frac{x^2}{y^2}\right)$.

By contrast `$(\frac{x^2}{y^2})$` gives $(\frac{x^2}{y^2})$.

`\left` and `\right` must be properly nested.

Delimiter height, part 2: doing it yourself

You can (and sometimes should) choose the size yourself:

`$(\ldots)$`	$(\ldots)$
`$\bigl(\ldots\bigr)$`	$\bigl(\ldots\bigr)$
`$\Bigl(\ldots\Bigr)$`	$\Bigl(\ldots\Bigr)$
`$\biggl(\ldots\biggl)$`	$\biggl(\ldots\biggl)$
`$\Biggl(\ldots\Biggr)$`	$\Biggl(\ldots\Biggr)$

These need not be properly nested (so be careful).

Delimiter height, part 3: examples

```
f\bigl((x + 1)^{4} + 1\bigr)
```

$$f\bigl((x+1)^{4}+1\bigr)$$

```
\biggl(\sum_{n\ge 1} a_{n} n^{-s}\biggr)^2
```

$$\biggl(\sum_{n\ge 1} a_{n} n^{-s}\biggr)^2$$

Spacing adjustments

Normally LaTeX puts tasteful amounts of whitespace into expressions without your help. In certain situations, though, you can make formulae look neater by adding or deleting space.

- `\,` — thin space (`$a\,b$` is $a\,b$)
- `\!` — thin backspace (`$a\!b$` is $a\!b$)

Example. The code `$x^{n}/n!$` gives $x^{n}/n!$, with an apparent gap between the exponent and slash. Better is `$x^{n}\!/n!$`, which gives $x^{n}\!/n!$.

Spacing adjustments in integrals

Multiple integrals sometimes look bad without a bit of backspacing (\!). A thin space (\,) before a differential is often tasteful.

```
\[ \int_{0}^{2\pi}\!\!
   \int_{0}^{\infty} e^{-r^2}r\,dr\,d\theta = \pi \]
```

$$\int_{0}^{2\pi}\!\!\int_{0}^{\infty} e^{-r^2}r\,dr\,d\theta = \pi$$

Ugly mistakes

- Italicizing text by putting it in math mode. This looks $awful$. Use `\emph{...}` instead: *awful*.

- Writing long variable names without `\mathit{...}` or `\text`.

- Putting two displayed equations one after the other. Use an alignment to line up the equality signs and space more tastefully.

- Forgetting to use `\left` and `\right` or explicit resizing commands when you put parentheses around a tall formula.