LaTeX maths and graphics

Tim Love

May 26, 2004

This handout assumes that you have already read the Advanced LaTeX[1] document handout, so if you're unsure about 'environments', read no further.

Note that there's an alternative way of producing maths in LaTeX - $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX. See the online manual[2] for details.

If you want to more more about graphics, see Using Imported Graphics in LaTeX2e Documents[3] by Keith Reckdahl.

Comments and bug reports to Tim Love (tpl@eng.cam.ac.uk).

Contents

[1] http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/latex_advanced/latex_advanced.html
[2] http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/amslatex.dvi
[3] http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/epslatex.ps

1 Maths

There's more to maths typesetting than meets the eye. Many conventions used in the typesetting of plain text are inappropriate to maths. LaTeX goes a long way to help you along with the style. For example, in a LaTeX maths environment, letters come out in *italics*, `'-'` as '$-$' (minus) instead of the usual '-' (dash), '*' becomes $*$, `'` becomes $'$ and spacing is changed (less around '/', more around '+').

Many of the usual LaTeX constructions can still be used in maths environments but their effect may be slightly different; eg `\textbf{ }` only affects letters and numbers. '`{`' and '`}`' are still special characters; they're used to group characters.

As usual in LaTeX you can override the defaults, but think before doing it: maths support in LaTeX has been carefully thought out and is quite logical though the LaTeX source text may not be very readable. It's a good idea to write out the formulae on paper before you start LaTeXing, and try not to overdo the use of the '`\frac`' construction; use '`/`' instead.

1.1 Environments

There are 2 environments to display one-line equations.

equation:- Equations in this environment are numbered.

```
\begin{equation}
 x + iy
\end{equation}
```

$$x + iy \tag{1}$$

displaymath:- These won't be numbered. `\[`, `\]` can be used as abbreviations for `\begin{displaymath}` and `\end{displaymath}`.

```
\begin{displaymath}
 x + iy
\end{displaymath}
```

$$x + iy$$

Never leave a blank line before these equations; it starts a new paragraph and looks ugly. '`\displaystyle`' is the font type used to print maths in these *display* environments. Other relevant environments are:-

math:- For use in text. `\(` and `\)` can be used to delimit the environment, as can the TeX constructions `$` and `$`. For example, `$x=y^2$` gives $x = y^2$.

eqnarray:- This is like a 3 column tabular environment. Each line by default is numbered. You can use the `eqnarray*` variant to suppress numbering altogether.

```
\begin{eqnarray}
a1 & = & b1 + c1\nonumber\\
a2 & = & b2 - c2
\end{eqnarray}
```

$$\begin{aligned} a1 &= b1 + c1 \\ a2 &= b2 - c2 \qquad (2) \end{aligned}$$

Maths in these 2 sorts of environments have different default sizes for some characters and other behavioural differences so that a line of maths won't impinge on text lines below or above. If you want to put some non-maths text in amongst maths then enclose it in an `\mbox{...}`.

1.2 Special Characters

The `amssymb` package offers more symbols if the following aren't enough. The symbols[4] document (a postscript file) has a bigger list.

1.2.1 Greek

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

1.2.2 Miscellaneous

$\ldots$	`\ldots`	$\cdots$	`\cdots`	$\vdots$	`\vdots`
$\ddots$	`\ddots`	$\pm$	`\pm`	$\mp$	`\mp`
$\times$	`\times`	$\div$	`\div`	$\ast$	`\ast`
$\star$	`\star`	$\circ$	`\circ`	$\bullet$	`\bullet`
$\cdot$	`\cdot`	$\cap$	`\cap`	$\bigcap$	`\bigcap`
$\cup$	`\cup`	$\bigcup$	`\bigcup`	$\uplus$	`\uplus`
$\biguplus$	`\biguplus`	$\sqcap$	`\sqcap`	$\sqcup$	`\sqcup`
$\bigsqcup$	`\bigsqcup`	$\vee$	`\vee`	$\bigvee$	`\bigvee`
$\wedge$	`\wedge`	$\bigwedge$	`\bigwedge`	$\setminus$	`\setminus`
$\wr$	`\wr`	$\diamond$	`\diamond`	$\bigtriangleup$	`\bigtriangleup`
$\bigtriangledown$	`\bigtriangledown`	$\triangleleft$	`\triangleleft`	$\triangleright$	`\triangleright`
$\oplus$	`\oplus`	$\bigoplus$	`\bigoplus`	$\ominus$	`\ominus`
$\otimes$	`\otimes`	$\bigotimes$	`\bigotimes`	$\oslash$	`\oslash`
$\odot$	`\odot`	$\bigodot$	`\bigodot`	$\bigcirc$	`\bigcirc`
$\amalg$	`\amalg`	$\leq$	`\leq`	$\prec$	`\prec`
$\preceq$	`\preceq`	$\ll$	`\ll`	$\subset$	`\subset`
$\subseteq$	`\subseteq`	$\sqsubseteq$	`\sqsubseteq`	$\in$	`\in`
$\vdash$	`\vdash`	$\geq$	`\geq`	$\succ$	`\succ`
$\succeq$	`\succeq`	$\gg$	`\gg`	$\supset$	`\supset`

[4]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/symbols.ps

$\supseteq$	`\supseteq`	$\sqsupseteq$	`\sqsupseteq`	$\ni$	`\ni`
$\dashv$	`\dashv`	$\equiv$	`\equiv`	$\sim$	`\sim`
$\simeq$	`\simeq`	$\asymp$	`\asymp`	$\approx$	`\approx`
$\cong$	`\cong`	$\neq$	`\neq`	$\doteq$	`\doteq`
$\propto$	`\propto`	$\models$	`\models`	$\perp$	`\perp`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\bowtie$	`\bowtie`
$\smile$	`\smile`	$\frown$	`\frown`	$\aleph$	`\aleph`
$\hbar$	`\hbar`	$\imath$	`\imath`	$\jmath$	`\jmath`
ℓ	`\ell`	$\wp$	`\wp`	$\Re$	`\Re`
$\Im$	`\Im`	$\prime$	`\prime`		`\empty`
∇	`\nabla`	$\surd$	`\surd`	$\top$	`\top`
$\bot$	`\bot`	$\Vert$	`\\|`	$\angle$	`\angle`
$\forall$	`\forall`	$\exists$	`\exists`	$\neg$	`\neg`
$\flat$	`\flat`	$\natural$	`\natural`	$\sharp$	`\sharp`
$\backslash$	`\backslash`	∂	`\partial`	∞	`\infty`
$\triangle$	`\triangle`	$\sum$	`\sum`	$\prod$	`\prod`
$\coprod$	`\coprod`	$\int$	`\int`	$\oint$	`\oint`

1.2.3 Arrows

$\leftarrow$	`\leftarrow`	$\Leftarrow$	`\Leftarrow`	$\rightarrow$	`\rightarrow`
$\Rightarrow$	`\Rightarrow`	$\leftrightarrow$	`\leftrightarrow`	$\Leftrightarrow$	`\Leftrightarrow`
$\mapsto$	`\mapsto`	$\hookleftarrow$	`\hookleftarrow`	$\leftharpoonup$	`\leftharpoonup`
$\leftharpoondown$	`\leftharpoondown`	$\rightleftharpoons$	`\rightleftharpoons`	$\longleftarrow$	`\longleftarrow`
$\Longleftarrow$	`\Longleftarrow`	$\longrightarrow$	`\longrightarrow`	$\Longrightarrow$	`\Longrightarrow`
$\longleftrightarrow$	`\longleftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\longmapsto$	`\longmapsto`
$\hookrightarrow$	`\hookrightarrow`	$\rightharpoonup$	`\rightharpoonup`	$\rightharpoondown$	`\rightharpoondown`
$\uparrow$	`\uparrow`	$\Uparrow$	`\Uparrow`		
$\downarrow$	`\downarrow`	$\Downarrow$	`\Downarrow`	$\updownarrow$	`\updownarrow`
$\nearrow$	`\nearrow`	$\searrow$	`\searrow`	$\swarrow$	`\swarrow`
$\nwarrow$	`\nwarrow`				

1.2.4 Calligraphic

These characters are available if you use the `\cal` control sequence.

```
${\cal A B C D E F G H I J K L M N O P Q R S T U V W X Y Z}$
```

gives $\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

1.2.5 Character Modifiers

`\hat{e}`	$\hat{e}$	`\widehat{easy}`	$\widehat{easy}$
`\tilde{e}`	$\tilde{e}$	`\widetilde{easy}`	$\widetilde{easy}$
`\check{e}`	$\check{e}$	`\breve{e}`	$\breve{e}$
`\acute{e}`	$\acute{e}$	`\grave{e}`	$\grave{e}$
`\bar{e}`	$\bar{e}$	`\vec{e}`	$\vec{e}$
`\dot{e}`	$\dot{e}$	`\ddot{e}`	$\ddot{e}$
`\not e`	$\not e$		

Note that the wide versions of `hat` and `tilde` cannot produce very wide alternatives. The '`\not`' operator hasn't properly cut the following letter. The *Fine Tuning* section on page 9 describes how to adjust this.

If you want to place one character above another, you can use `\stackrel`, which prints its first argument in small type immediately above the second

```
$ a \stackrel{def}{=} b + c $
```

gives $a \stackrel{def}{=} b + c$

See the *Macros* section for how to stack characters using `atop`.

1.2.6 Common functions

In a maths environment, LaTeX assumes that variables will have single-character names. Function names require special treatment. The advantage of using the following control sequences for common functions is that the text will not be put in math italic and subscripts/superscripts will be made into limits where appropriate.

```
\arccos  \arcsin  \arctan  \arg  \cos  \cosh    \cot
\coth    \csc     \deg     \det  \dim  \exp     \gcd
\hom     \inf     \ker     \lg   \lim  \liminf  \ln
\log     \max     \min     \Pr   sec   \sin     \sinh
\sup     \tan     \tanh
```

1.3 Subscripts and superscripts

These are introduced by the '^' and '_' characters. Depending on the base character and the current style, the sub- or superscripts may go to the right of or directly above/below the main character. With letters it goes to the right.

```
$F_2^3$
```

produces 'F_2^3'. Note that the sub- and superscripts aren't in line. To make them so, you can add an invisible character after the 'F'. `$F{}_2^3$` produces $F{}_2^3$.

With $\sum$ the default behaviour is different in display and text styles.

```
$\sum_{i=0}^2 $
```

produces $\sum_{i=0}^2$ (text style) but

```
\[\sum_{i=0}^2 \]
```

produces (in display style)

$$\sum_{i=0}^2$$

This default behaviour can be overridden, if you really need to. For example in text mode,

```
$\sum\limits_{i=0}^2$
```

produces $\sum\limits_{i=0}^2$

1.4 Overlining, underlining and bold characters

```
$\underline{one} \overline{two}$
```

produces $\underline{one}\overline{two}$. This is not a useful facility if it's used more than once on a line. The lines are produced so that they don't quite overlap the text; lines over or under different words won't in general be at the same height.

To be able to reproduce bold maths, it's best to use the `bm` package. `$E = \bm{mc^2}$` produces $E = \boldsymbol{mc^2}$.

Alternatively, you can use `\mathbf{}` to create bold characters - `$\mathbf{F}_2^3$` produces $\mathbf{F}_2^3$. or you can use the following idea

```
\usepackage{amsbsy} % This loads amstext too
\begin{document}
$\omega + \boldsymbol{\omega}$

% Use the following if whole expressions need to be in bold
{\boldmath $\omega $}
\end{document}
```

1.5 Roots

```
$\sqrt{4} + \sqrt[3]{x + y}$
```

gives $\sqrt{4} + \sqrt[3]{x + y}$.

1.6 Fractions

Three constructions for putting expressions above others are

frac:- `$\frac{1}{(x + 3)}$` produces $\frac{1}{(x+3)}$.

choose:- `${n + 1 \choose 3}$` produces $\binom{n+1}{3}$.

atop:- `${x \atop y}$` produces ${x \atop y}$.

These constructions can be used with ones described earlier. *E.g.,*

```
\[ \sum_{-1\le i \le 1 \atop 0 < j < \infty} f(i,j)\]
```

gives

$$\sum_{-1\le i \le 1 \atop 0 < j < \infty} f(i,j)$$

1.7 Delimiters

these	are made by these	*and these*	are made by these
$($	`(`	$)$	`)`
$[$	`[`	$]$	`]`
$\{$	`\{`	$\}$	`\}`
$\lfloor$	`\lfloor`	$\rfloor$	`\rfloor`
$\lceil$	`\lceil`	$\rceil$	`\rceil`
$\langle$	`\langle`	$\rangle$	`\rangle`
$/$	`/`	$\backslash$	`\backslash`
$\vert$	`|`	$\Vert$	`\\|`
$\uparrow$	`\uparrow`	$\Uparrow$	`\Uparrow`
$\downarrow$	`\downarrow`	$\Downarrow$	`\Downarrow`
$\updownarrow$	`\updownarrow`	$\Updownarrow$	`\Updownarrow`

This table shows the standard sizes. To get bigger sizes, use these prefices

(for left delimiters)	(for right delimiters)	magnification
`\bigl`	`\bigr`	a bit bigger, but won't overlap lines
`\Bigl`	`\Bigr`	150% times `big`
`\biggl`	`\biggr`	200% times `big`
`\Biggl`	`\Biggr`	250% times `big`

For example,

```
$\Biggl\{2\Bigl(x(3+y)\Bigr)\Biggr\}$
```

gives $\Biggl\{2\Bigl(x(3+y)\Bigr)\Biggr\}$. If you're not using the default text size these commands might not work correctly. In that case try the `exscale` package.

It's preferable to let LaTeX choose the delimiter size for you by using `\left` and `\right`. These will produce delimiters just big enough for the formulae inbetween.

```
$\left( \frac{(x+iy)}{\{\int x\}} \right)$
```

gives $\left(\frac{(x+iy)}{\{\int x\}} \right)$

The left and right delimiters needn't be the same type. It's sometimes useful to make one of them invisible

```
\[ z = \left\{
              \begin{array}{ll}
                   1 & (x>0)\\
                   0 & (x<0)
              \end{array}
       \right.
\]
```

produces

$$z = \left\{ \begin{array}{ll} 1 & (x>0)\\ 0 & (x<0) \end{array} \right.$$

Over- and underbracing works too.

```
$\overbrace{\alpha \ldots \omega}^{\mbox{greek}}
 \underbrace{a \ldots z}_{\mbox{english}}$
```

produces $\overbrace{\alpha \ldots \omega}^{\mbox{greek}} \underbrace{a \ldots z}_{\mbox{english}}$. The use of `\mbox` stops the text appearing in math italic.

1.8 Numbering and labelling

Numbering happening automatically when you display equations. If you *don't* want an equation numbered, use `\nonumber` beside the equation. Equation numbers appear to the right of the maths by default. To make them appear on the left use the `leqno` class option (i.e., use `\documentclass[leqno,....]{....}`).

Use `\label{}` to label an equation (or figure, section etc) in order to reference from elsewhere.

```
\begin{equation}
W_{\bf S}(t,\omega) = \int\limits_{-\infty}^{\infty} {
  {\cal R}_{\bf S}(t,\tau) e^{-j\omega\tau} \,d \tau }
\label{LABELLING}
\end{equation}
```

$$W_{\mathbf{S}}(t,\omega) = \int\limits_{-\infty}^{\infty} \mathcal{R}_{\mathbf{S}}(t,\tau) e^{-j\omega\tau} \,d\tau \tag{3}$$

Now the following text

```
refers back to equation \ref{LABELLING}
```

refers back to equation 3 by number, and

```
refers back to the equation on page \pageref{LABELLING}
```

refers back to the equation on page 8.

A file will have to be LaTeX'ed twice before the references, both forwards and backwards, will be correctly produced.

1.9 Matrices

The `array` environment is like LaTeX's `tabular` environment except that each element is in `math` mode. The number and alignment of columns is controlled by the arguments - use `l`, `c` or `r` to represent each column with either left, center or right alignment. The default font style used is `\textstyle` but you can override this by changing the `\displaystyle`.

```
\begin{math}
\begin{array}{clrr} %
      a+b+c & uv & x-y & 27 \\
       x+y  & w  & +z  & 363
\end{array}
\end{math}
```

produces $\begin{array}{clrr} a+b+c & uv & x-y & 27 \\ x+y & w & +z & 363 \end{array}$

The rows are arranged so that their centres are aligned. You can align their tops or bottoms instead by using a further argument when you create the array.

```
\begin{array}{clrr}[t]
```

would produce top-aligned lines, and `'[b]'` would produce bottom-aligned ones. The *Delimiters* section of this document shows how to bracket matrices.

TeX has a few maths facilities not mentioned in the LaTeX book. The following TeX construction might be useful.

```
\begin{math}
\bordermatrix{&a_1&a_2&...&a_n\cr
          b_1 & 1.2  & 3.3  & 5.1  & 2.8  \cr
          c_1 & 4.7  & 7.8  & 2.4  & 1.9  \cr
          ... & ...  & ...  & ...  & ...  \cr
          z_1 & 8.0  & 9.9  & 0.9  & 9.99  \cr}
\end{math}
```

$$\begin{array}{cc} & \begin{array}{cccc} a_1 & a_2 & ... & a_n \end{array} \\ \begin{array}{c} b_1 \\ c_1 \\ ... \\ z_1 \end{array} & \begin{pmatrix} 1.2 & 3.3 & 5.1 & 2.8 \\ 4.7 & 7.8 & 2.4 & 1.9 \\ ... & ... & ... & ... \\ 8.0 & 9.9 & 0.9 & 9.99 \end{pmatrix} \end{array}$$

1.10 Macros

These aid readability, save on repetitive typing and offer ways of producing stylistic variations on standard LaTeX formats.

```
\def\bydefn{\stackrel{def}{=}}
\def\convf{\hbox{\space \raise-2mm\hbox{$\textstyle
    \bigotimes \atop \scriptstyle \omega$} \space}}
```

produce $\stackrel{def}{=}$ and $\bigotimes\atop\scriptstyle\omega$ when `$\bydefn$` and `$\convf$` are typed.

1.11 Packages

The following packages may be of help

- easybmat[5] - easy block matrices
- easyeqn[6] - easy equations.
- easymat[7] - easy matrices
- easytable[8] - easy tables
- easyvector[9] - easy vectors
- delarray[10] - nested arrays
- theorem[11] - gives more choice in theorem layout
- `subeqnarray` - Renumbering of sub-arrays in math-mode
- `subeqn` - Different numbering sub-arrays

1.12 Fine tuning

It's generally a good idea to keep punctuation outside math mode; LaTeX's normal handling of spacing around punctuation is suspended during maths. Sometimes you might want to adjust the spacing in a formula (*e.g.*, you might want to add space before *dx*). Use these symbols :-

`a\, b`	$(a\,b)$	thin space
`a\> b`	$(a\>b)$	medium space
`a\; b`	$(a\;b)$	thick space
`a\! b`	$(a\!b)$	negative thin space

Long math expressions aren't broken automatically unless you use the `breqn`[12] package, which is still a little experimental. In an `eqnarray` environment you may want to break a long line manually. You can do this by putting

```
y & = & a + b \nonumber \\
  &   & + k
```

[5]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/easybmat.dvi
[6]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/easyeqn.dvi
[7]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/easymat.dvi
[8]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/easytable.dvi
[9]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/easyvector.dvi
[10]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/delarray.dvi
[11]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/theorem.dvi
[12]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/breqndoc.dvi

to give

$$\begin{eqnarray} y & = & a+b \nonumber \\ & & +k \end{eqnarray} \tag{4}$$

but the spacing around the '+' on the 2nd line is wrong because LaTeX thinks it's a unary operator. You can fool LaTeX into treating it as a binary operator by inserting a hidden character.

```
y & = & a + b \nonumber \\
  &   & \mbox{} + k
```

gives

$$\begin{eqnarray} y & = & a+b \nonumber \\ & & \mbox{} + k \end{eqnarray} \tag{5}$$

You can use the `\lefteqn` construction to format long expressions so that continuation lines are differently indented.

```
\begin{eqnarray}
\lefteqn{x+ iy=}\\
 & & a + b + c + d + e + f + g + h + i + j + k +\nonumber\\
 & & l + m \nonumber
\end{eqnarray}
```

$$x+iy= \tag{6}$$
$$a+b+c+d+e+f+g+h+i+j+k+$$
$$l+m$$

If you want more vertical spacing around a line you can create an invisible vertical "struct" in LaTeX. `\rule[-.3cm]{0cm}{1cm}` creates a box of width 0, height 1cm which starts .3cm below the usual line base. By adjusting these values you should be able to create as much extra space below/above the maths as you like. "${A \over B}$ and" is created by

```
$A \over B$  \rule[-.3cm]{0cm}{1cm}{and}
```

1.13 Maths and Postscript fonts

It's easy to use a postscript font (like helvetica) for the text of a LaTeX document. What's harder is using the same font for maths. An easy, reasonable option is to use the `mathptmx` package to put the maths into the postscript *Times* and *symbol* fonts where possible.

Alternatively, use

- the `mathpazo` package (loads Palatino as the text font family and a mixture of the Pazo and CM fonts for math).
- the `mathpple` package (loads Palatino as the text font family and a mixture of artificially obliqued Euler fonts and CM fonts for math).

Commercial and free alternatives are under development.

1.14 Matlab and LaTeX

Matlab[13] has some support for LaTeX production. For example

[13] http://www-h.eng.cam.ac.uk/help/tpl/programs/matlab.html

```
latex('(sin(x)+2*x+3*x^2)/(5*x+6*x^2)','math.tex')
```

puts the LaTeX representation of the expression into a file called `math.tex`. Type "`help latex`" inside matlab for details.

1.15 Examples

-
  ```
  \begin{equation}
  \hat{\theta}_{w_i} = \hat{\theta}(s(t,{\cal U}_{w_i})).
  \end{equation}
  ```

  gives

  $$\hat{\theta}_{w_i} = \hat{\theta}(s(t,{\cal U}_{w_i})). \tag{7}$$

-
  ```
  \begin{eqnarray}
  {\cal M}^2(\hat{\theta},\theta) &=& E[(\hat{\theta} - \theta)^2]
  \nonumber \\
  {\cal M}^2(\hat{\theta},\theta) &=& {\rm var}^2(\hat{\theta}) +
  {\cal B}^2(\hat{\theta}).
  \end{eqnarray}
  ```

  gives

  $$\begin{aligned} {\cal M}^2(\hat{\theta},\theta) &= E[(\hat{\theta} - \theta)^2] \\ {\cal M}^2(\hat{\theta},\theta) &= {\rm var}^2(\hat{\theta}) + {\cal B}^2(\hat{\theta}). \end{aligned} \tag{8}$$

-
  ```
  \begin{equation}
  \hat{W}_{s}(t,\omega;\phi) \bydefn
  \int\limits_{-\infty}^{\infty}
  {\hat{\cal R}_s(t,\tau;\psi) e^{-j\omega \tau}
  \, d \tau }
  \end{equation}
  ```

  gives

  $$\hat{W}_{s}(t,\omega;\phi) \overset{def}{=} \int\limits_{-\infty}^{\infty} \hat{\cal R}_s(t,\tau;\psi) e^{-j\omega \tau} \, d \tau \tag{9}$$

-
  ```
  \begin{eqnarray}
  {\cal B}(t,\omega) & \approx &
  {1 \over 4\pi}
  {\cal D}_t^2 W_{\bf S}(t, \omega)
  {{{\scriptstyle \infty} \atop
  {\displaystyle \int \! \int \!
  }}\atop {\scriptstyle -\infty}}
  t_1^2
  \phi(t_1,\omega_1) \, dt_1 d\omega_1
  \nonumber \\
  && +
  {1 \over 4\pi}
  {\cal D}_\omega^2 W_{\bf S}(t, \omega)
  {{{\scriptstyle \infty} \atop
  {\displaystyle \int \! \int \!
  }}\atop {\scriptstyle -\infty}}
  \omega_1^2
  \phi(t_1,\omega_1) \, dt_1 \, d\omega_1.
  ```

```
\label{F4}
\end{eqnarray}
```

gives

$$\mathcal{B}(t,\omega) \approx \frac{1}{4\pi}\mathcal{D}_t^2 W_{\mathbf{S}}(t,\omega)\int\int_{-\infty}^{\infty} t_1^2\phi(t_1,\omega_1)\,dt_1 d\omega_1 + \frac{1}{4\pi}\mathcal{D}_\omega^2 W_{\mathbf{S}}(t,\omega)\int\int_{-\infty}^{\infty} \omega_1^2\phi(t_1,\omega_1)\,dt_1\,d\omega_1. \qquad (10)$$

-
  ```
  \newsavebox{\DERIVBOXZLM}
  \savebox{\DERIVBOXZLM}[2.5em]{$\Longrightarrow\hspace{-1.5em}
  \raisebox{.2ex}{*}
  \hspace{-.7em}\raisebox{-.8ex}{\scriptsize lm}\hspace{.7em}$}
  \newcommand{\Deriveszlm}{\usebox{\DERIVBOXZLM}}

  \Deriveszlm
  ```

  gives

  $\overset{*}{\underset{\text{lm}}{\Longrightarrow}}$

2 Graphics

LaTeX has a `picture` environment in which pictures can be drawn, but you'll find graph paper handy. `xfig` can create code for the `picture` environment but the resulting graphics still suffer several limitations: only certain slopes and circles can be reproduced. The best method presently available on Unix is to use `xfig` to produce postscript files, which have no such limitations, but require a postscript printer or equivalent.

Whatever graphics you want to add, you should use the `figure` environment so that LaTeX can cope sensibly with situations where, for example, you attempt to insert near the bottom of a page a graphic that's half a page high. The `figure` environment will *float* the graphic to the top or bottom of the page, or on the next page, with preferences that you can provide.

`h`	here
`t`	top of page
`b`	bottom of page
`p`	on a page with no text

Putting `!` as the first argument in the square brackets will encourage LaTeX to do what you say, even if the result's sub-optimal. See the online hints about floats in LaTeX[14] for further details.

```
\begin{figure}[htbp]
   \vspace{0.5in}
   \caption{0.5 inch of space}
\end{figure}
```

It's possible to have more than one graphic in a *figure*. See the example later on.

[14]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/float_hint.html

Figure 1: 0.5 inch of space

2.1 Postscript

`pdflatex` supports `JPEG`, `PNG` and`PDF` images - but *not* postscript. `latex` supports Postscript files as long as they have a proper bounding box comment; *i.e.* LaTeX requires full Encapsulated Postscript[15] as produced by (for example) `xv` and `xfig` on Unix. If the file hasn't got a `BoundingBox` line near the top, you can use `ps2epsi` to generate one. Wherever the postscript file comes from, simply use

```
\documentclass[dvips]{article}
\usepackage{graphicx}
```

then include the postscript file using the following commands

```
\begin{figure}[htbp]
\includegraphics{yourfile.ps}
\end{figure}
```

LaTeX can cope with compressed postscript files too, but since latex can't read the `BoundingBox` line from the compressed file, you need to provide it. If your compressed file's called `yourfile.ps.gz`, copy the `BoundingBox` line into a file called `yourfile.ps.bb`. Then the following works

```
\begin{figure}[htbp]
\includegraphics{yourfile.ps.gz}
\end{figure}
```

Just about all of the following facilities use postscript. You'll need to run `latex` to generate '`foo.dvi`'. This file can be viewed by the latest `xdvi` program, which can cope with embedded postscript. Run `dvips -o foo.ps foo.dvi` to convert the resulting `DVI`/postscript file to pure postscript. This will produce a file that can be previewed with `ghostview` or `gs`. On the teaching system this file can be printed out using `plotview` or `lp`.

See the *Creating and Printing graphics on PC, Mac, SUN and HP machines*[16] and *Xfig*[17] handouts for more details.

2.1.1 `psfrag`: adding maths to postscript files

Many packages that produce postscript output don't provide good maths facilities. It's often easier to add the maths in later using the `psfrag` package. This lets you replace text in a postscript file (produced with xfig, matlab, etc) by a fragment of LaTeX. For example, doing

```
\usepackage{psfrag}
...
\begin{figure}
\psfrag{MATHS}{$x^2$}
\includegraphics{foo.eps}
\end{figure}
```

[15]http://www-h.eng.cam.ac.uk/help/tpl/graphics/postscript.html
[16]http://www-h.eng.cam.ac.uk/help/tpl/graphics/graphics2/graphics2.html
[17]http://www-h.eng.cam.ac.uk/help/tpl/graphics/xfig/xfig.html

would display the file with `MATHS` replaced by x^2. See the online documentation[18] for details.

2.1.2 Postscript from PCs/Macs

Modern applications should generate a conforming EPS file. Under windows when you're printing to file, look at the PostScript properties (or Advanced options), and choose (depending on the driver you have) either 'Archive Format', 'Encapsulated PS', 'Optimize for Portability' or 'Page Independence'. People seem to have more luck with the free Adobe postscript driver than with the Microsoft one.

The resulting file should begin with `%!PS`. If it doesn't it's not postscript. Remove any characters that are before `%!PS`, and (to be on the safe side) remove anything after the final `%%EOF` line.Platform-specific considerations do crop up. The EPS generated on Macintoshes will use `ASCII 13` line terminators, while Unix will use `ASCII 10` (and DOS will use both). If this causes trouble, use `emacs` to convert, or try (in Unix)

```
    tr "\015" "\012" <original.ps >new.ps
```

2.2 Scaling, rotation, clipping, wrap-around and shadows

To scale, use some optional arguments

```
  \includegraphics[width=5cm,height=10cm]{yourfile.ps}
```

would rescale the postscript so that it was 5cm wide and 10cm high. To make the picture 5cm wide and scale the height in proportion use

```
  \includegraphics[width=5cm]{yourfile.ps}
```

To rotate anticlockwise by the specified number of degrees, use

```
  \includegraphics[angle=150]{yourfile.ps}
```

These options can be combined - note that order matters. The following examples demonstrate how to combine these features and how to use the `subfig` package to have more than one graphic in a figure.

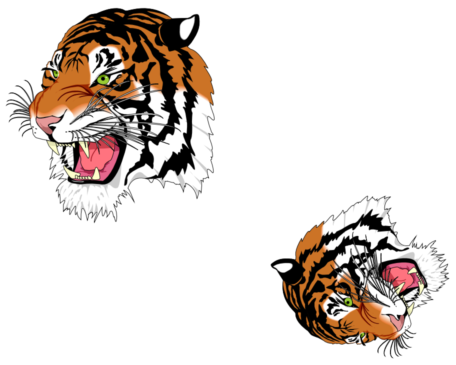
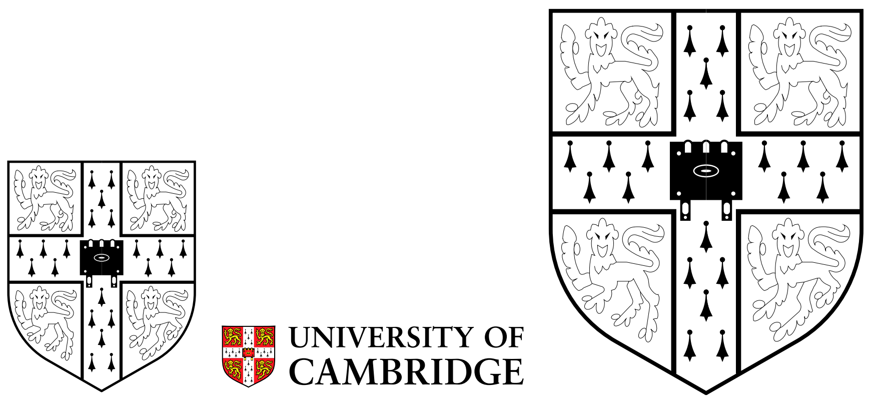


Figure 2: Tigers

[18]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/psfrag.ps

```
\centering
\begin{figure}[hbtp]
  \includegraphics[height=40mm]{/export/ghostfonts/tiger.eps}
  \includegraphics[angle=120, height=20mm]{/export/ghostfonts/tiger.eps}
\caption{Tigers}
\end{figure}

% remember to do \usepackage{subfig} at the top of the document!
\begin{figure}[hbtp]
\centering
\subfloat[Small]% \quad on the next line adds spacing
{\includegraphics[height=30mm]{/export/ghostfonts/crest.eps}}\quad
\subfloat[Medium]
{\includegraphics[width=40mm]{CUni3.eps}}\quad
\subfloat[Large]
{\includegraphics[height=50mm]{/export/ghostfonts/crest.eps}}
\caption{3 crests}
\end{figure}
```



(a) Small (b) Medium (c) Large

Figure 3: 3 crests

To clip the postscript image use the `viewpoint` argument. The following fragment would display only part of the image. The viewport coordinates are in the same units as the bounding box.

```
\begin{figure}[htbp]
\includegraphics[viewport=200 400 400 600,width=5cm,clip]%
{CUniv3.eps}
\end{figure}

% Use the floatflt package
\begin{floatingfigure}[l]{4cm}
\includegraphics[width=2cm]{/export/ghostfonts/crest.eps}
\caption{Using floatingfigure}
\end{floatingfigure}
```

The `floatflt`[19] package lets you insert a graphic and have the text wrap around it. You can provide 2 arguments to the `floatingfigure` command: the first (`l` or `r`) selects whether you want the graphic to be on the left or right of the

[19]http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/floatflt.dvi

page. The 2nd argument gives the width of the graphic. Not all text will flow perfectly around (for example, `verbatim` text fails, as illustrated below) so check the final output carefully.

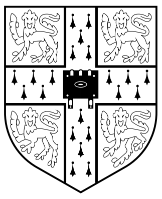


Figure 4: Using floating-figure

Using the `fancybox` package gives you access to `\shadowbox`, `\ovalbox`, `\Ovalbox` and `\doublebox` commands, which can be used with text or with graphics. For example, `\shadowbox{shadow package}` produces shadow package and

```
\ovalbox{\includegraphics[height=10mm]{CUni3.eps}}
```

produces
UNIVERSITY OF
CAMBRIDGE
. Unfortunately, the `fancybox` package as supplied suppresses the table of contents. The locally produced `contents-fancybox` solves this, but may introduce graphics problems.

2.3 GIF and jpeg files

CUED users can include a file called `keyboard.gif`, for example, by doing `gif2ps -b keyboard.gif` (as long as you don't change the `GIF` file you need only do this once) and then including the `GIF` file as you would a postscript one.

For JPEG files run "`jpeg2ps -h` *file.jpg* > *file.eps*" then include the postscript file in the usual way. The resulting `eps` file will be little bigger than the original file. Alternatively, if you put the 'Bounding Box' line from the `eps` file and put it in *file.jpg.bb* you can include JPEG files in the same way that you do postscript files.