Package `mathcmd`*

F. Bosisio
E-mail: `fbosisio@bigfoot.com`

1997/12/20

Abstract

Documentation for the package `mathcmd`.

1 Introduction

This package provides some useful math-commands which are simpler to use and prettier then their standard LaTeX counterpart.

In particular the "`\d`" command is redefined, so care should be taken, expecially when including this package in an already existent LaTeX file.
The original work of "`\d`" (i.e. place a dot under its argument) is now done by the "`\UnderDot`" command.

2 The options

At now, six options are available with the `mathcmd` package, which comes out in matched pairs.
The "`ThreeSubscrSum`" and "`TwoSubscrSum`" options control the placement of subscripts for the "`\Sum`" command: the latter (which is the default) puts the index under the simbol toghether with the starting point, while the former puts the index on the lower right corner.
The "`ProdVettWedge`" and "`ProdVettTimes`" options select which symbol is to be used for the *vector product*: the first one uses a "`\wedge`" (default), whilst the second uses a "`\times`".

*This is version 2.2, last revised 1997/12/20; documentation date 2005/04/09

Finally, the "VectOpStr" and "VectOpSymb" options controls whether the vector-operator commands "\Grad", "\Div" and "\Rot" should produce a roman string (default) or a "\nabla" symbol followed by an operator, respectively.

3 Math-mode commands

The mathcmd package defines a number of math-mode commands.

3.1 The text command

The "\text{...}" command is defined to be equivalent to "\mbox" (except when the "amstext" package is also used, since this package already defines this command in a better way).
It is intended for inserting pieces of text in a formula.

3.2 Commands for doing integrals

The "\Int" (capitalized) differs from the LaTeX command "\int" in that it is always printed in displaystyle and if it has only a subscript, this is somewhat lowered so that it looks better.

```
\Int_{...}^{...}   ...   \d{...}
```

The "\d{...}" command is for making the differential symbol at the end of integrals: it simply prints a "d" followed by its argument and preceded by a little space, which seems prettier.

3.3 Commands for making sums

The "\Sum{$\cdots = \cdots, \cdots$}" command works differently, depending on which option between "TwoSubscrSum" (default) and "ThreeSubscrSum" has been specified.
In the first case, it expands to "\displaystyle\sum_{...=...}^{...}", whereas in the second case the second and third argument are treated in the same way as before (i.e. as a subscript and superscript, respectively), but the first argument (the one before the "=") is placed near the lower-left edge of the "\sum" symbol and the "=" is not printed.

There is also a "\SUM{...}" command which is useful when only a subscript is desired, irrespectively of the option specified: in fact, is is equivalent to "\displaystyle\sum_{...}".

3.4 Derivatives symbols

The commands "`\DerTot{...}{...}`", "`\DerPar{...}{...}`" and "`\DerNorm{...}`" generate the symbols of total derivative, partial derivative and normal derivative, respectively.
In other words, they are the same as:

```
\displaystyle\frac{d...}{d...}
\displaystyle\frac{\partial ...}{\partial ...}
\displaystyle\frac{\partial ...}{\partial n}
```

3.5 Arrow-limits commands

The command "`\TendsTo`$[\cdots,\cdots]$" generates a right-arrow with optionally an underscript wich is another smaller right-arrow between the two comma-separeted arguments inside the square brackets.
For example, the command "`f(x) \TendsTo[x,0] 1`" generates the following output: $f(x) \underset{x\to 0}{\longrightarrow} 1$.

3.6 Vector-operators commands

Finally, the commands "`\Grad`", "`\Div`" and "`\Rot`" generates the strings "grad", "div" and "rot" in roman type and with small spaces added before and after, if the "VectOpStr" option is in effect (default). If, instead, the option "VectOpSymb" was specified, they generate "`\nabla`", "`\nabla\cdot`" and "`\nabla\ProdVett`" respectively.

The command "`\ProdVett`" is intended for making the symbol of vector product, and evaluates to "`\times`" with the option "ProdVettTimes" and to "`\wedge`" with the (default) option "ProdVettWedge".

		`VectOpStr`	`VectOpSymb`	
			`ProdVettWedge`	`ProdVettTimes`
`\Grad`	$\Rightarrow$	grad	∇	∇
`\Div`	$\Rightarrow$	div	$\nabla\cdot$	$\nabla\cdot$
`\Rot`	$\Rightarrow$	rot	$\nabla\wedge$	$\nabla\times$

4 Implementation

```
%%
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesPackage{\FileName}[\filedate\space v\fileversion\space\filedescr]
%%
\newif\if@ThreeSubscrSum
\DeclareOption{ThreeSubscrSum}{\@ThreeSubscrSumtrue}
\DeclareOption{TwoSubscrSum}{\@ThreeSubscrSumfalse}
%%
\DeclareOption{ProdVettWedge}{\let\ProdVett=\wedge}
\DeclareOption{ProdVettTimes}{\let\ProdVett=\times}
%%
\newif\if@VectOper@Symbol
\DeclareOption{VectOpStr}{\@VectOper@Symbolfalse}
\DeclareOption{VectOpSymb}{\@VectOper@Symboltrue}
%%
\ExecuteOptions{TwoSubscrSum,ProdVettWedge,VectOpStr}
```

The "`\ProcessOptions*`" command was used here instead of "`\ProcessOptions*`" in order to process the options in the "`\usepackage`" order, rather than in the declaration order

```
%%
\ProcessOptions*
```

`\text` The definition of `\text` is deferred at the `\begin{document}`, so that if the `amstext` package is loaded (even after this package), its definition of text will be used, since it seems better.

```
%%
\AtBeginDocument{\ifx\undefined\text \def\text#1{\ \mbox{#1}\ } \fi}
```

`\Int` The command `\@INT@sub@SUP` has one argument and then a mandatory superscript: il typesets an integral symbol (in `\displaystyle`) with the argument as a lower-bound and the superscript as an upper-bound.

```
%%
\def\@INT@sub@SUP#1^#2{{\displaystyle\int_{#1}^{#2}}}
```

The command `\@INT@subONLY` has only an argument, which is used as as a subscript for a `displaystyle \int`, but lowered of `0.2ex`.

```
\def\@INT@subONLY#1{{\displaystyle\int_{\raisebox{-0.2ex}{$_{#1}$}}}}
```

The command `\@INT@sub` must have a subscript and may optionally be followed by a superscript.

```
\def\@INT@sub_#1{\@ifnextchar^{\@INT@sub@SUP{#1}}{\@INT@subONLY{#1}}}
```

The command `\@INT@SUP@sub` must have a superscript and a subscript, in that order: they are passed to a `displaystyle \int` command.

```
\def\@INT@SUP@sub^#1_#2{{\displaystyle\int_{#2}^{#1}}}
```

The `\Int` command invokes `\@INT@SUP@sub` or `\@INT@sub` or simply expands to `\displaystyle\int` depending on what follows (`^`, `_` or neither of them).

```
\newcommand{\Int}{\@ifnextchar^{\@INT@SUP@sub}{\@ifnextchar_{\@INT@sub}{\displaystyl
```

`\d` The original `\d` command is saved in `\UnderDot`, then (at the `\begin{document}` to avoid conflicts with other packages) it is redefined as a "`d`" preceeded by some space an followed by its argument (it is intended for printing the differential symbol at the end of an integral).

```
%%
\AtBeginDocument{ \let\UnderDot=\d \renewcommand{\d}[1]{\:\:d#1} }
```

`\Sum` The command `\INNER@SUM` is defined differently depending on the selected option. It has 3 or 2 orguments, ended by an exclamation mark and separated by an equal sign and a comma. The two last arguments are passed as a subscrit and as a superscript to a `displaystyle \sum`, whereas the first argument (if they are 3) is typeset in `scriptstyle` and lowered so that it occours at the lower right corner of the `sum` symbol.

```
%%
\if@ThreeSubscrSum
  \def\INNER@SUM#1=#2,#3!{{\displaystyle\sum_{#2}^{#3}\raisebox{-0.6ex}{$_{\!#1}$}}}
\else
  \def\INNER@SUM#1,#2!{{\displaystyle\sum_{#1}^{#2}}}
\fi
```

The `\Sum` command is then simply a call to `\INNER@SUM`, with a question mark added to denote the end of the argument.

```
\newcommand{\Sum}[1]{\INNER@SUM#1!}
```

`\SUM` The `\SUM` command has one only argument which is used as a subscript and, again, is always printed in `\displaystyle`.

```
\newcommand{\SUM}[1]{{\displaystyle\sum_{#1}}}
```

`\DerPar` The commands "`\DerPar`", "`\DerTot`" and "`\DerNorm`" generate a `displaystyle \frac` command, with a "`\partial`" or "`d`" or "`\partial n`" added to its arguments and followed by a little space.

```
%%
\newcommand\DerPar[2]{{\displaystyle\frac{\partial #1}{\partial #2}\:}}
\newcommand\DerTot[2]{{\displaystyle\frac{d#1}{d#2}\:}}
\newcommand\DerNorm[1]{{\displaystyle\frac{\partial #1}{\partial n}\:}}
```

\TendsTo The command \@TendeQuando must be followed by a pair of square brackets, which enclose two arguments separated by a comma. It place a \rightarrow command between them and puts the whole under a bigger \rightarrow. The \mathop allows the \limits command, which, in turn, place the subscript *under* the symbol.

```
%%
\def\@TendeQuando[#1,#2]{ \mathop{\longrightarrow}\limits_{#1 \rightarrow #2} }
```

The \TendsTo command invokes \@TendeQuando if followed by a "[", or \longrightarrow otherwise.

```
\def\TendsTo{\@ifnextchar[{\@TendeQuando}{\longrightarrow}}
```

\Grad The \Grad, \Div and \Rot commands generates a symbol or a roman word, depending on the selected option.

```
%%
\if@VectOper@Symbol
  \newcommand\Grad{\nabla}
  \newcommand\Div{\nabla\cdot}
  \newcommand\Rot{\nabla\ProdVett}
\else
  \newcommand\Grad{\:\mathrm{grad}\,}
  \newcommand\Div{\:\mathrm{div}\,}
  \newcommand\Rot{\:\mathrm{rot}\,}
\fi
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Change History

v2.1

General: Added copyright notice and changed addresses 1

v2.2

General: Usage of the double-quote character (") avoided 1