The bropd package*

Benjamin Tatlock†

October 14, 2014

Abstract

The bropd package simplifies the process of writing differential operators and brackets in LaTeX. The commands facilitate the easy manipulation of equations involving brackets and allow partial differentials to be expressed in alternate forms.

1 Introduction

The bropd package introduces a bracket command that automatically selects parentheses, square brackets or braces as appropriate based on the number of brackets. As such, brackets can be freely added and removed without alterations to the rest of an equation. Commands for typing ordinary and partial differentials are also created, with two optional concise forms for partial derivatives. Equations can therefore be written in a standard form and their style altered globally at a later date.

2 Usage

This section outlines the use of the three commands provided by the bropd package.

2.1 Brackets

The bracket command replaces `\left(` and `\right)` with the single command, `\br` to produce brackets in a math environment. The input `\br{x}` produces

$$\left(x\right).$$

When nested, the type of bracket is determined automatically, for example, `r=\br{\br{x-a}^2+\br{y-b}^2}^{\frac{1}{2}}` results in

$$r=\left[\left(x-a\right)^2+\left(y-b\right)^2\right]^{\frac{1}{2}}.$$

---

[*] This documentation corresponds to bropd v1.2, dated 2014/10/14.

[†] `Benjamin.Tatlock@mail.com`

2.2 Differential operators

2.2.1 Ordinary differential

The `\od` command produces ordinary differentials, such as `\od{y}{x}` which is displayed as

$$\frac{\mathrm{d}y}{\mathrm{d}x}.$$

The order of the differential can be specified as an optional argument, illustrated here with the command `\od[2]{u}{x}=-\omega^2u` which results in

$$\frac{\mathrm{d}^2u}{\mathrm{d}x^2} = -\omega^2 u.$$

When using `\od` with a sans-serif font, the package should be loaded with the optional argument `sans`.

2.2.2 Partial differential

The command `\pd` follows the same form as `\od`, but instead produces partial differentials. By default, `\pd{u}{t}=6u\pd{u}{x}-\pd[3]{u}{x}` produces

$$\frac{\partial u}{\partial t} = 6u\frac{\partial u}{\partial x} - \frac{\partial^3 u}{\partial x^3}.$$

Loading the package with the option `pdshort` results in an alternative format for partial differentials:

$$u_t = 6uu_x - u_{xxx}$$

If the optional argument is not a positive integer, the default form is reverted to, as with `\pd[n]xy`:

$$\frac{\partial^n x}{\partial y^n}$$

A second alternative form is produced when the option `pdalt` is specified:

$$\partial_t u = 6u\partial_x u - \partial_{xxx} u$$

The syntax for partial differentials is extended to allow higher order terms such as

$$\frac{\partial^3 u}{\partial x^2 \partial t}$$

to be easily typeset with `\pd{u}{x,x,t}`. When applying a partial derivative to a longer term, the first argument can be left blank and the longer string written as an additional argument, as demonstrated by `\pd{}{z}{x+y}` which results in

$$\frac{\partial}{\partial z}\left(x+y\right).$$

To prevent brackets from being placed after a partial differential with an empty numerator, use `!` for the first argument, for example, `\pd{!}{x}`:

$$\frac{\partial}{\partial x}$$

2.3 Further examples

Combining the commands provided by this package allows more complex terms to be written concisely. The `\br` allows equations with multiple brackets to be constructed and manipulated quickly and easily. An example of this is `\br{x-y}\br{Q-\frac12\br{P+\br{x-1}\br{\br{a+b}\br{u+c}-\od st}}}`, which results in

$$(x-y)\left(Q-\frac{1}{2}\left\{P+(x-1)\left[(A+B)\,(u+c)-\frac{\mathrm{d}s}{\mathrm{d}t}\right]\right\}\right).$$

The form of the partial differentials written with the `\pd` command can be changed at any time. `\pd Et+\sum_{i=1}^3\pd{}{x_i}{\br{E+p}u_i}` produces

$$\frac{\partial E}{\partial t}+\sum_{i=1}^{3}\frac{\partial}{\partial x_i}\left[(E+p)\,u_i\right]$$

by default. With `pdshort`, the same input instead creates

$$E_t+\sum_{i=1}^{3}\left[(E+p)\,u_i\right]_{x_i},$$

while with `pdalt` the output is

$$\partial_t E+\sum_{i=1}^{3}\partial_{x_i}\left[(E+p)\,u_i\right].$$

3 Implementation

Announce the name, package version and LaTeX 2_ε requirement

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{bropd}
[2014/10/14 1.2 automated typing of brackets and differential operators]
```

Initiate conditionals and counters

```
\newif\if@bropd@a\newif\if@bropd@b\newif\if@bropd@s\newif\if@bropd@u
\newcounter{@bropd@c}\newcounter{@bropd@d}
\newcounter{@bropd@e}\newcounter{@bropd@f}
```

Provide optional arguments for alternative form partial differential and sans-serif fonts

```
\DeclareOption{pdshort}{\@bropd@atrue\@bropd@ufalse}
\DeclareOption{pdalt}{\@bropd@atrue\@bropd@utrue}
\DeclareOption{sans}{\@bropd@strue}\ProcessOptions
```

Define character for `\pd` with empty numerator and no brackets following

```
\def\@bropd@t{!}
```

`\@bropd@g` Internal command for default partial differential form

```
\def\@bropd@g#1#2#3#4{\def\@bropd@h{#1}
```

Create numerator

```
\ifx\@bropd@o\@bropd@t\let\@bropd@o\empty\fi
```

Reset counters

```
\ifx\@bropd@h\empty
\setcounter{@bropd@c}0\setcounter{@bropd@d}1
\let\@bropd@h\empty
```

Determine order of partial derivative and display as power of partial if greater than one

```
\@for\@bropd@i:=#3\do{\stepcounter{@bropd@c}}
\frac{#4\ifnum\value{@bropd@c}>1
^{\arabic{@bropd@c}}\fi\@bropd@o}
```

Cycle through each term for the denominator. If a term is the same as the previous, add to counter, otherwise print along with counter value

```
{\@for\@bropd@i:=#3\do{
\ifx\@bropd@i\@bropd@h
\stepcounter{@bropd@d}
\else
\@bropd@h
\ifnum\value{@bropd@d}>1
^{\arabic{@bropd@d}}
\fi
#4\setcounter{@bropd@d}1
\fi
\let\@bropd@h\@bropd@i}
\@bropd@h
\ifnum\value{@bropd@d}>1
^{\arabic{@bropd@d}}\fi}
\else
```

If optional argument provided, display partial differential to stated order

```
\frac{#4^{#1}\@bropd@o}{#4\noexpand#3^{#1}}
\fi}
```

`\@bropd@j` Internal command for brackets. Evaluate terms within brackets to determine the highest bracket level

```
\def\@bropd@j#1{\setcounter{@bropd@f}0
\begingroup\renewcommand{\br}[1]{\stepcounter{@bropd@e}##1
\ifnum\value{@bropd@e}>\value{@bropd@f}
\setcounter{@bropd@f}{\value{@bropd@e}}
\fi
\addtocounter{@bropd@e}{-1}}\sbox0{$#1$}\endgroup
\loop
\ifnum\value{@bropd@f}>2
\addtocounter{@bropd@f}{-3}
\repeat
```

Detemine bracket type for current level

```
\ifnum\value{@bropd@f}>0
\ifnum\value{@bropd@f}>1
\def\@bropd@k{\left\lbrace#1\right\rbrace}
\else
\def\@bropd@k{\left[#1\right]}
\fi
\else
\def\@bropd@k{\left(#1\right)}
\fi}
\if@bropd@a
\let\@bropd@l\empty
```

`\br` External command for brackets

```
\newcommand\br[1]{\let\@bropd@m\@bropd@l
\@bropd@j{\let\@bropd@l\empty#1\let\@bropd@l\@bropd@m}
\ifx\@bropd@l\empty
\def\@bropd@n{\@bropd@k}
\else
\let\@bropd@m\@bropd@l
\def\@bropd@n{\@bropd@k_{\@bropd@m}}
\fi
\let\@bropd@l\empty
\@bropd@n}
\if@bropd@u
```

`\@bropd@q` Internal command for first alternative partial differential command

```
\def\@bropd@q#1#2#3{\def\@bropd@h{#1}
\ifx\@bropd@h\empty
```

```
\@bropd@bfalse
\else
```

Default partial differential if character provided as optional argument

```
\if!\ifnum9<1#1!\else_\fi
\@bropd@bfalse
\else
\@bropd@btrue
\fi
\fi
\ifx\@bropd@o\@bropd@t
\@bropd@btrue\let\@bropd@o\empty
\fi
\if@bropd@b\@bropd@g{#1}\@bropd@o{\noexpand#3}\partial
\else
```

Construct alternate form partial differential

```
\partial_{\ifx\@bropd@h\empty
\@for\@bropd@i:=#3\do\@bropd@i\else\setcounter{@bropd@c}0
\loop
\ifnum\value{@bropd@c}<#1
#3\stepcounter{@bropd@c}
\repeat
\fi}
\@bropd@o
\fi}
\else
```

Internal command for second alternative partial differential command

```
\def\@bropd@q#1#2#3{\def\@bropd@h{#1}
\ifx\@bropd@h\empty
\@bropd@bfalse
\else
```

Default partial differential if character provided as optional argument

```
\if!\ifnum9<1#1!\else_\fi
\@bropd@bfalse
\else
\@bropd@btrue
\fi
\fi
\ifx\@bropd@o\@bropd@t
\@bropd@btrue\let\@bropd@o\empty
\fi
\if@bropd@b\@bropd@g{#1}\@bropd@o{\noexpand#3}\partial
\else
```

Construct alternate form partial differential

```
\def\@bropd@p{\ifx\@bropd@h\empty
\@for\@bropd@i:=#3\do\@bropd@i\else\setcounter{@bropd@c}0
\loop
\ifnum\value{@bropd@c}<#1
#3\stepcounter{@bropd@c}
\repeat
\fi}
\ifx\@bropd@o\empty
\let\@bropd@l\@bropd@p\else{\@bropd@o}{_{\@bropd@p}}\fi\fi}
\fi
\else
```

Internal command for default partial differentials

```
\def\@bropd@q#1#2#3{\@bropd@g{#1}{#2}{\noexpand#3}\partial}
```

`\br` External command for brackets

```
\newcommand\br[1]{\@bropd@j{#1}\@bropd@k}
\fi
```

`\pd`

```
\newcommand\pd[3][]{\def\@bropd@o{#2}
\ifx\@bropd@o\empty
\let\@bropd@v\br
\else
\let\@bropd@v\empty
\fi
\@bropd@q{#1}{#2}{#3}\@bropd@v}
```

`\@bropd@r` Format ordinary differential based on font type

```
\if@bropd@s
\def\@bropd@r{\mathsf d}
\else
\def\@bropd@r{\mathrm d}
\fi
```

`\od` External command for ordinary differentials

```
\newcommand\od[3][]{\def\@bropd@h{#1}
\frac{\@bropd@r\ifx\@bropd@h\empty\else^{#1}\fi#2}
{\@bropd@r#3\ifx\@bropd@h\empty\else^{#1}\fi}}
\endinput
```

Change History

v1.0

- General: Initial version 4

v1.1

- `\@bropd@g`: Fixed higher order differentials with denominators containing commands 5
- `\@bropd@q`: Added second alternative form for partial differentials 5
  - Allowed partial differential with empty numerator without need for bracket command after 6
- `\pd`: Brackets automatically added after empty numerator 7

v1.2

- `\@bropd@j`: Changed `\bgroup` to `\begingroup` and `\egroup` to `\endgroup` to remove erroneous introduction of space before and after brackets 5