Options

In this package there are four categories of options (examples and differences will be shown further)

1. for interval notation
   - `isointerval` for using standardized format of interval described in **ISO 31-11**
   - `isoointerval` for using standardized alternative format of interval described in **ISO 31-11**
   - `fnspeinterval` for using special notation used at FNSPE CTU in Prague
2. for tensor notation (now for vectors and matrices)
   - `isotensor` for using standardized format of tensor
   - `undertensor` for using underline notation of tensor
   - `arrowtensor` for using arrow notation of tensor
3. for complex notation (real and complex part)
   - `isocomplex` for using standardized format of complex and real part
   - `oldcomplex` for using old LaTeX default format of complex and real part
4. for definition notation
   - `deftext` for definition using *def.* over the equal
   - `defcolon` for definition using the colon with equal

Macros

Interval

Let a and b be real numbers.

Closed interval

Using of macro

```
\ci{a}{b}
```

as **c**losed **i**nterval.

- `isointerval`

$$[a, b]$$

- `isoointerval` (same as for `isointerval`)

$$[a, b]$$

- `fnspeinterval`

$$\langle a, b \rangle$$

Opened interval

Using of macro

```
\oi{a}{b}
```

as **o**pened **i**nterval.

- `isointerval`

$$]a, b[$$

- `isoointerval`

$$(a, b)$$

- `fnspeinterval` (same as for `isoointerval`)

$$(a, b)$$

Right closed interval

Using of macro

```
\rci{a}{b}
```

as **r**ight **c**losed **i**nterval.

- `isointerval`

$$]a, b]$$

- `isoointerval`

$$(a, b]$$

- `fnspeinterval`

$$(a, b\rangle$$

Left closed interval

Using of macro

```
\lci{a}{b}
```

as **l**eft **c**losed **i**nterval.

- `isointerval`

$$[a, b[$$

- `isoointerval` (same as for `isointerval`)

$$[a, b)$$

- `fnspeinterval`

$$\langle a, b)$$

Using in text

All these macros can be used directly in text (thanks to the command *ensuremath*). Therefore one can use this syntax

```
Let $x$ be in \ci{a}{b}
```

which casts: Let x be in $[a, b]$.

Tensor

Let x be vector and A be matrix.

Vector

Using of macro

```
\vec{x}
```

as **vec**tor.

- `isotensor` - small letter with italic boldface

  $\mathbf{x}$

- `undertensor`

  $\underline{x}$

- `arrowtensor`

  $\vec{x}$

Matrix

Using of macro

```
\mat{x}
```

as **mat**rix.

- `isotensor` - capital letter with italic boldface

  $\mathbf{A}$

- `undertensor`

  $\underline{\underline{A}}$

- `arrowtensor`

  $\overleftrightarrow{A}$

Using in text

All these macros can be used directly in text (thanks to the command *ensuremath*). Therefore one can use this syntax

```
Let \vec{x} be real.
```

which casts: Let $\vec{x}$ be real.

Macro for set

Set of natural numbers from 1 to n

Using of macro

```
\allset{n}
```

as **all** natural number up to n **set** leads to

$$\{1, 2, \ldots, n\}.$$

Set of natural numbers from 0 to n

Using of macro

```
\allsetzero{n}
```

as **all** natural number up to n **set** with **zero** leads to

$$\{0, 1, \ldots, n\}.$$

Differentiability class

Just symbol

Using of macro

```
\cclass
```

as **C class** leads to

$$\mathscr{C}.$$

C infinity

Using of macro

```
\ccinf
```

as **C class** of **inf**inity leads to

$$\mathscr{C}^{\infty}.$$

C of order d

Using of macro

```
\ccof{d}
```

as **C class of** order leads to

$$\mathscr{C}^{d}.$$

Complex

Let $z \in \mathbb{C}$.

Real part

Using of macro

```
\Re{x}
```

as **Re**al.

- `oldcomplex`

$$\mathfrak{Re}\,\{z\}$$

- `isocomplex`

$$\operatorname{Re} z$$

Imaginary part

Using of macro

```
\Im{x}
```

as **Im**aginary.

- `oldcomplex`

$$\mathfrak{Im}\,\{z\}$$

- `isocomplex`

$$\operatorname{Im} z$$

Using in text

All these macros can be used directly in text (thanks to the command *ensuremath*). Therefore one can use this syntax

```
Let $x$ equal to \Re{z}.
```

which casts: Let x equal to Re z.

Subscript

Subscript text with two or more characters should be written in roman style (not italic as default). One can use prefix ! which makes the word after it in roman style. Using of macro

```
A_{!unique}
```

which leads to

$$A_{\text{unigue}}$$

instead of classic

$$A_{unique}$$

Floor and ceiling functions

Floor function

Macro

```
\floor{x}
```

as **floor** function leads to

$$\lfloor x \rfloor$$

Ceil function

Macro

```
\ceil{x}
```

as **ceil** function leads to

$$\lceil x \rceil$$

Definition operator

There are two ways to set a definition operator. First with *text* and the second with *colon*.

Text definition

Macro

```
x \df a
```

- `deftext`

$$x \stackrel{\text{def.}}{=} a$$

- `defcolon`

$$x{:=}a$$

Special sets of numbers

Natural number

Macro

```
\natun
```

as **natu**ral **n**umber leads to

$$\mathbb{N}$$

Natural number with zero included

Macro

```
\nnzero
```

as **n**atural **n**umber **zero** leads to

$$\mathbb{N}_0$$

Integers

Macro

```
\inte
```

as **inte**regers leads to

$$\mathbb{Z}$$

Rational number

Macro

```
\ratin
```

as **rati**onal **n**umber leads to

$$\mathbb{Q}$$

Real number

Macro

```
\realn
```

as **real n**umber leads to

$$\mathbb{R}$$

Complex number

Macro

```
\compn
```

as **comp**ex **n**umber leads to

$$\mathbb{C}$$

Using in text

All these macros can be used directly in text (thanks to the command *ensuremath*). Therefore one can use this syntax

```
Let $n$ be in \natun
```

which casts: Let n be in $\mathbb{N}$.

Derivative

It is derived from *physics* package. The manual is here.

Operator

Partially derived from *physics* package.

Gradient

Macro

```
\grad
```

as **grad**ient leads to

$$\boldsymbol{\nabla}$$

Divergence

Macro

```
\div
```

as **div**ergence leads to

$$\boldsymbol{\nabla\cdot}$$

Derived from *physics* package, the original meaning of this command as a maths symbol for dividing has alias

```
\divisionsymbol
```

which cast

$$\div$$

Rotation

In English literature as **curl** operator has macro

```
\rot
```

as **rot**ation and leads to

$$\boldsymbol{\nabla\times}$$

One can also use *physics* package command

```
\curl
```

Laplacian

Macro

```
\lapl
```

as **lapl**acian leads to

$$\Delta$$

One can also use *physics* package notation

$$\nabla^2$$

which is cast by macro

```
\laplacian
```

Degree

Macro

```
\degree
```

as **degree** leads to °. Can be used without math mode.

Physics unit

Variable unit

Macro

```
\varun{m}{kg}
```

as **var**iable **un**it leads to

$$[m] = \mathrm{kg}$$

This macro can be used directly in text (thanks to the *ensure* function). Therefore one can use

```
where \varun{m}{kg} is the mass.
```

which casts: where $[m] = \mathrm{kg}$ is the mass.

Unit

Macro

```
m\unit{kg}
```

as **unit** leads to

$$m\ \mathrm{kg}$$

This macro looks as

```
\;\mathrm{kg}
```

the space before the roman characters is very important in science publications.

Expected value

Macro

```
\expv{x}
```

as **exp**ected **v**alue leads to

$$\langle x \rangle$$

Shortcuts

One half

Macro

```
\hlf
```

as **h**a**lf** leads to

$$\frac{1}{2}$$

One over

Macro

```
\oover{x}
```

as **o**ne **over** leads to

$$\frac{1}{x}$$

Spaces

Horizontal space

Macro

```
\hem[width]
```

as **h**space{**em**} leads to horizontal space of specific width (multiples of em). Special case is 1em

```
\mathrm{text}\hem\mathrm{text}
```

which leads to

$$\mathrm{text}\quad\mathrm{text}$$

or shortcut form space with 2em width

```
\mathrm{text}\htem\mathrm{text}
```

which casts

$$\mathrm{text}\qquad\mathrm{text}$$

Implies with em spaces

Macro

```
\impem
```

as **implies** with **em** spaces leads to

$$\mathrm{text}\quad\Rightarrow\quad\mathrm{text}$$