The **overarrows** package[*]

Julien Labbé

Julien.Labbe@univ-grenoble-alpes.fr

January 17, 2023

Abstract

A LaTeX package to create custom arrows over (and under) math expressions, mainly for vectors. Arrows stretch with content, scale with math styles, and have a correct kerning when a subscript follows.

Short example:

```
\NewOverArrowCommand{overrightharpoon}{%
  end=\rightharpoonup
}

\begin{align*}
  &\overrightharpoon{v} &&\overrightharpoon{v}_{subscript} \\
  &\overrightharpoon{ABCD} &&\overrightharpoon*{v}_{subscript}
\end{align*}
```

$$\overrightarrow{v} \qquad \overrightarrow{v}_{subscript}$$
$$\overrightarrow{ABCD} \qquad \overrightarrow{v}_{subscript}$$

Predefined commands are also provided:

- to typeset vectors:
$$\vec{v} \qquad \overrightarrow{AB},$$

- to draw arrows of various shapes above math expressions:
$$\overrightarrow{AB} \quad \overleftarrow{AB} \quad \overleftrightarrow{AB} \quad \overrightarrow{AB} \quad \overleftharpoon{AB} \quad \overrightharpoon{AB} \quad \overleftharpoon{AB} \quad \overline{AB},$$

- to draw arrows of various shapes under math expressions:
$$\underrightarrow{AB} \quad \underleftarrow{AB} \quad \underleftrightarrow{AB} \quad \underrightarrow{AB} \quad \underleftharpoon{AB} \quad \underrightharpoon{AB} \quad \underleftharpoon{AB} \quad \underline{AB}.$$

Contents

1   Presentation of the package

The overarrows package allows to create commands for drawing arrows over math expressions. These arrows:

- are fully customisable, at command definition, through a key-value interface;

- stretch with the content and can cover many characters, like in $\overrightarrow{AB}$;

- scale with math styles[1], like in $\overrightarrow{v}_{\overrightarrow{u}_{\overrightarrow{w}}}$.

Commands created with the overarrows package are provided with a starred variant, that removes the extra end space generated by the arrow. This is particularly useful when the command is followed by a subscript. For example, the velocity of the center of mass can be written with exactly the same kerning when scalar v_{cm} or vector $\overrightarrow{v}_{cm}$ (no extra space before the subscript, unlike the output of the unstarred variant: $\overrightarrow{v}_{cm}$).

The overarrows package was primitively written for vectors, but in a highly customisable way. It can be used to define a large variety of arrows, using math symbols, or PGF/TikZ commands. It's also possible to create commands that draw the arrows under. Some predefined commands are provided, giving[2], for arrow over:

$$\overrightarrow{\alpha+\beta} \quad \overleftarrow{\alpha+\beta} \quad \overleftrightarrow{\alpha+\beta} \quad \overrightarrow{\alpha+\beta} \quad \overleftarrow{\alpha+\beta} \quad \overline{\alpha+\beta} \quad \overleftarrow{\alpha+\beta} \quad \overline{\alpha+\beta}$$

and for arrow under :

$$\underrightarrow{\alpha+\beta} \quad \underleftarrow{\alpha+\beta} \quad \underleftrightarrow{\alpha+\beta} \quad \underrightarrow{\alpha+\beta} \quad \underleftarrow{\alpha+\beta} \quad \underline{\alpha+\beta} \quad \underrightarrow{\alpha+\beta} \quad \underline{\alpha+\beta}.$$

2   Introduction

2.1   Vector arrows

Vectors are commonly typeset in bold face, or with an arrow above[3]. For this second convention, TeX/LaTeX provides the command \vec, which accents its content (using the \mathaccent command) with the character → (\mathchar"017E in Computer Modern font). But → isn't extensible, and gives: $\vec{v}$, $\vec{AB}$ or $\vec{\mathrm{grad}}$ (there's no command \widevec analogous to \widehat).

An extensible alternative is given by the command \overrightarrow, available in TeX/LaTeX, and which is redefined by the commonly used amsmath package. But its arrow, built with the \rightarrow symbol → , is too large with the default *Computer Modern* font: $\overrightarrow{AB}$. Another alternative is the esvect package, which provides the \vv command and a set of custom arrows: $\overrightarrow{AB}$, $\overrightarrow{AB}$, $\overrightarrow{AB}$, $\overrightarrow{AB}$, $\overrightarrow{AB}$, $\overrightarrow{AB}$, $\overrightarrow{AB}$, $\overrightarrow{AB}$.

---

[1] \displaystyle, \textstyle, \scriptstyle and \scriptscriptstyle.

[2] Displayed here with the old-arrows$^{\to\,\mathrm{P.\,14}}$ option.

[3] See, for example: International Organization for Standardization. (2019). *Quantities and units – Part 2: Mathematics* (ISO Standard No. 80000-2:2019). https://www.iso.org/standard/64973.html.

2.2 Stack and arrow macros

It worth looking at the definition of amsmath `\overrightarrow` command:

```
\long macro:->\mathpalette {\overarrow@ \rightarrowfill@ }
```

Three macros are used here:

\mathpalette adapts the output to the current math style;

\overarrow@ is the *stack macro*, that puts the arrow above the content;

\rightarrowfill@ is the *arrow macro*, that holds the content of the arrow.

The command `\vv` from esvec is defined with a very similar way, using its own stack macro (`\overvect@`) and arrow macro (`\vectfill@`).

The overarrows package uses the same mechanism. Arrow and stack macros are set, at command creation, through a key-value interface provided by the pgfkeys package (after creation, however, the command definition is static and the key-value interface is not used).

2.3 Extensible arrows

Arrows drawn by the commands `\overrightarrow` or `\vv` are built by joining math symbols, and made extensible by repetition of the central symbol[4]. Thus, the line of the macro `\overrightarrow` is made by repetition of command `\relbar` — (which simply corresponds to the minus sign), while `\vv` use its own command `\relbareda` -.

This method may generate some undesirable spacing issues, when symbols badly overlap. See, for example, the output of amsmath `\overrightarrow` (left) and esvect `\vv` (right) in `\scriptscriptstyle` math style (scaled by a factor 4):

$$\overrightarrow{long\ vector} \quad \overrightarrow{long\ vector}.$$

While the arrow on the left lets guess where the symbols — overlap, the arrow on the right present unwanted spaces and show clearly its composition as association of the symbols –, - and →.

By default, the overarrows package uses the same mechanism to extend arrows according to their contents. Settings and tools are provided to perform fine tuning and avoid spacing issues. As example, see below the `\overrightarrow` and `\vv` commands, as redefined by overarrows (in `\scriptscriptstyle` and scaled by a factor 4):

$$\overrightarrow{long\ vector} \quad \overrightarrow{long\ vector}$$

The overarrows package also provides an alternative mechanism. When used, the length `\overarrowlength` is set, according to the arrow command content, and can be employed, for example, to draw arrows using PGF/TikZ or the LaTeX picture environment.

---

[4]Using the TeX `\cleaders` command.

3   Quick start

3.1   Loading the package **overarrows**

To load the overarrows, simply add in preamble, before the "\begin{document}":

```
\usepackage{overarrows}
```

Options can be given, in a comma-separated list. For example, to use the predefined commands shown in the section 1, page 3, write:

```
\usepackage[allcommands, old-arrows]{overarrows}
```

This define the commands (described in section 4.2.5, page 18):

- \overrightarrow^{→P. 18}
- \overleftarrow^{→P. 19}
- \overleftrightarrow^{→P. 19}
- \overrightharpoonup^{→P. 19}
- \overrightharpoondown^{→P. 19}
- \overleftharpoonup^{→P. 19}
- \overleftharpoondown^{→P. 19}
- \overbar^{→P. 19}

- \underrightarrow^{→P. 19}
- \underleftarrow^{→P. 19}
- \underleftrightarrow^{→P. 19}
- \underrightharpoonup^{→P. 19}
- \underrightharpoondown^{→P. 20}
- \underleftharpoonup^{→P. 20}
- \underleftharpoondown^{→P. 20}
- \underbar^{→P. 20}

Note that the old-arrows^{→P. 14} option may give bad results, if math fonts have been changed. Simply remove the option in this case.

Many other options are available. See the complete list, page 11.

3.2   Commands creation

Commands are created with \NewOverArrowCommand^{→P. 15}. This macro take two mandatory arguments : the name of the command (without backslash), and the arrow configuration as comma-separated list of key-values. By default, a right arrow is set:

```
\NewOverArrowCommand{myovercmd}{}
$\myovercmd{test}$
```
$$\overrightarrow{test}$$

Commands are defined with a starred variant, designed to handle subscripts:

```
$ v_{sub} \qquad \myovercmd{v}_{sub} \qquad \myovercmd*{v}_{sub} $
```
$$v_{sub} \qquad \overrightarrow{v}_{sub} \qquad \overrightarrow{v_{sub}}$$

3.3   Start and end of the arrow

Extremities of the arrow are set by the keys start^{→P. 22} and end^{→P. 22}. For example, an arrow starting with a hook (symbols \lhook ⸂) and ending with two heads (symbol \twoheadrightarrow ↠) is defined by:

```
\NewOverArrowCommand{overhooktwoheadrightarrow}{%
  start=\lhook, end=\twoheadrightarrow,
}
```

Note that `\twoheadrightarrow` must be defined, as it is not in LaTeX. This can be done with the package amssymb, by adding in preamble:

```
\usepackage{amssymb}
```

With the previous definition, the result of the command `\overhooktwoheadrightarrow` is faulty:

```
$ \overhooktwoheadrightarrow{v} \qquad \overhooktwoheadrightarrow{AB} $
```

$$ \overset{\hookrightarrow\!\!\!\!\!\!\to}{v} \qquad \overset{\hookrightarrow\!\!\!\!\!\!\to}{AB} $$

The problem comes from symbols junction and the trimming used to obtain their overlap. It can be solved with the keys `trim start`[→ P. 23] and `trim end`[→ P. 23], which are numbers and set the corresponding trimming in math units (typically `1/18 em`). Appropriate values gives better results:

```
\NewOverArrowCommand{overhooktwoheadrightarrow}{%
  start=\lhook, end=\twoheadrightarrow,
  trim start=1.5, trim end=2,
}
$ \overhooktwoheadrightarrow{v} \qquad \overhooktwoheadrightarrow{AB} $
```

$$ \overset{\hookrightarrow\!\!\!\to}{v} \qquad \overset{\hookrightarrow\!\!\!\to}{AB} $$

If the math font differs from the default *Computer Modern*, the central part of the arrow may have inappropriate position or line width. This is because the default symbol used for the arrow line is `\relbareda` - from the esvect package. If needed, try to set the `middle`[→ P. 22] key with the symbol `\relbar` —. The trimming should also be adapted:

```
\NewOverArrowCommand{overhooktwoheadrightarrow}{%
  start=\lhook, end=\twoheadrightarrow, middle=\relbar, %
  trim start=0, trim end=3, trim middle=5,
}
$ \overhooktwoheadrightarrow{v} \qquad \overhooktwoheadrightarrow{AB} $
```

$$ \overset{\hookrightarrow\!\!\!\to}{v} \qquad \overset{\hookrightarrow\!\!\!\to}{AB} $$

Finding the correct values for `trim start`[→ P. 23], `trim end`[→ P. 23] and `trim middle`[→ P. 23] may need many trials. For this purpose, the macro `\TestOverArrow`[→ P. 16] displays the result of a command for different lengths and math styles:

```
\TestOverArrow{overhooktwoheadrightarrow}
```

\displaystyle	\textstyle	\scriptstyle	\scriptscriptstyle
$\overset{\hookrightarrow\to}{v}$	$\overset{\hookrightarrow\to}{v}$	$\overset{\hookrightarrow\to}{v}$	$\overset{\hookrightarrow\to}{v}$
$\overset{\hookrightarrow\to}{AB}$	$\overset{\hookrightarrow\to}{AB}$	$\overset{\hookrightarrow\to}{AB}$	$\overset{\hookrightarrow\to}{AB}$
$\overset{\hookrightarrow\to}{\text{grad}}$	$\overset{\hookrightarrow\to}{\text{grad}}$	$\overset{\hookrightarrow\to}{\text{grad}}$	$\overset{\hookrightarrow\to}{\text{grad}}$
$\overset{\hookrightarrow\to}{my\ long\ vector}$	$\overset{\hookrightarrow\to}{my\ long\ vector}$	$\overset{\hookrightarrow\to}{my\ long\ vector}$	$\overset{\hookrightarrow\to}{my\ long\ vector}$

3.4 Size and position of the arrow

A command `\OverRightarrow`, built with the symbols `\Relbar` = and `\Rightarrow` ⇒ , gives:

```
\NewOverArrowCommand{OverRightarrow}{%
  start=\Relbar,
  middle=\Relbar,
  end=\Rightarrow,
  trim=4,
}
$ \OverRightarrow{v} \qquad \OverRightarrow{AB} $
```
$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

The key `trim`[→P. 23] sets `trim start`[→P. 23], `trim middle`[→P. 23] and `trim end`[→P. 23] with the same value.

The previous arrow is visually too big. The macro `\smallermathstyle`[→P. 17] allows to obtain a better result:

```
\NewOverArrowCommand{OverRightarrow}{%
  start={\smallermathstyle\Relbar},
  middle={\smallermathstyle\Relbar},
  end=\Rightarrow,
  trim=4,
}
$ \OverRightarrow{v} \qquad \OverRightarrow{AB} $
```
$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

Note that `\smallermathstyle`[→P. 17] should not be used for `end`[→P. 22], because this last is formatted with the same math style as `start`[→P. 22].

It would be better to add an extra space between the arrow and the content of the command. This can be done with the key `space after arrow`[→P. 22]:

```
\NewOverArrowCommand{OverRightarrow}{%
  start={\smallermathstyle\Relbar},
  middle={\smallermathstyle\Relbar},
  end=\Rightarrow,
  trim=4,
  space after arrow=0.25ex,
}
$ \OverRightarrow{v} \qquad \OverRightarrow{AB} $
```
$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

Default arrows are slightly shifted to the right. For a left arrow, this should be reversed, using the keys `shift left`[→P. 21] and `shift right`[→P. 21]. These keys set the corresponding shifts, in math units. Example:

7

```
\NewOverArrowCommand{OverLeftarrow}{%
  start={\smallermathstyle\Leftarrow},
  middle={\smallermathstyle\Relbar},
  end=\Relbar,
  trim=4,
  space after arrow=0.25ex,
  shift left=0, shift right=2,
}
$ \OverLeftarrow{v} \qquad \OverLeftarrow{AB} $
```
$$\overset{\Leftarrow}{v} \qquad \overset{\Leftarrow}{AB}$$

Finally, the key `arrow under`$^{\to P.\,20}$ places the arrow below the content, instead of above (and `space before arrow`$^{\to P.\,22}$ sets the space upon it):

```
\NewOverArrowCommand{OverLeftRightarrow}{%
  start={\smallermathstyle\Leftarrow},
  middle={\smallermathstyle\Relbar},
  end=\Rightarrow,
  trim=4,
  arrow under,
  space before arrow=0.5ex,
  shift left=0, shift right=0,
}
$ \OverLeftRightarrow{v} \qquad \OverLeftRightarrow{AB} $
```
$$\underset{\Leftrightarrow}{v} \qquad \underset{\Leftrightarrow}{AB}$$

3.5 Symbols assemblage

Many LATEX math symbols are built by assemblage, using the macro `\joinrel`[5] which remove 3 math units of horizontal space. The overarrows package provides a flexible version of `\joinrel`, called `\xjoinrel`$^{\to P.\,16}$, which remove an arbitrary number of math units, given as optional argument.

Symbols association is then simple. As example, one can define a triple tail macro `\tttail` from the symbol `\succ` $\succ$:

```
\newcommand*{\tttail}{\succ\xjoinrel[10]\succ\xjoinrel[10]\succ}
$ \tttail $
```
$$\Rrightarrow$$

Thus defined, the macro `\tttail` can be used in arrow definition:

---

[5]For example, the symbol `\models` $\models$ is defined as `\mathrel{|}\joinrel\Relbar` and corresponds to the assemblage of a vertical line $|$ and the symbol `\Relbar` $=$. The command `\mathrel` modifies the spacing according to the math relation class ; `\Relbar` corresponds to the equal sign (it's definition is `\mathrel{=}`).

```
\NewOverArrowCommand{overtttailrightarrow}{%
  start={\tttail},
  end={\rightarrow},
  trim start=12,
  shift left=0, shift right=0,
  space after arrow=.2ex,
  min length=24,
}
$ \overtttailrightarrow{v} \qquad \overtttailrightarrow{AB} $
```

$$\overset{\ggg}{v} \qquad \overset{\ggg}{AB}$$

Here the `min length`[→P.20] key was added to ensure a minimum length (in math units) when the content of the command is small (as for a single character).

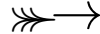The previous arrow would be better with a smaller tail, and this can be done with the macro `\smallermathstyle`[→P.17]. But a small tail and a normal sized head are not aligned; as `{\smallermathstyle\tttail}\xjoinrel[8]\rightarrow` gives:

$$\ggg\!\!\longrightarrow$$

The solution comes from the command `\vcenter` which centers materials on math axis. The tail must then be wrapped in a `\hbox`:

```
\NewOverArrowCommand{overtttailrightarrow}{%
  start={\vcenter{\hbox{$\smallermathstyle\tttail$}}},
  end={\rightarrow},
  trim start=12,
  shift left=0, shift right=0,
  space after arrow=.2ex,
  min length=24,
}
$ \overtttailrightarrow{v} \qquad \overtttailrightarrow{AB} $
```

$$\overset{\ggg}{v} \qquad \overset{\ggg}{AB}$$

Text symbols, namely symbols that are not defined in math mode, can also be used. They should yet be enclosed in the `\text` macro, from the amsmath package, to be correctly displayed and correctly scaled according to math style. With, for example, the arrow heads given by the symbols 40 and 41 of the *lasy* font:

```
\newcommand*{\leftarrowhead}{\usefont{U}{lasy}{m}{n}\symbol{40}}
\newcommand*{\rightarrowhead}{\usefont{U}{lasy}{m}{n}\symbol{41}}
\NewOverArrowCommand{overrightleftarrow}{%
  start=\text{\righttarrowhead},
  end=\text{\leftarrowhead},
  trim start=0.7, trim end=0.7,
  min length=20,
  shift leftright=-2,
}
$ \overrightleftarrow{AB} \qquad \scriptstyle\overrightleftarrow{AB} $
```

$$\overset{\rightharpoonup}{AB} \qquad \overset{\rightharpoonup}{AB}$$
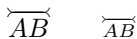
3.6 Drawing the arrow with TikZ

In addition to the default method presented previously (assemblage of symbols, as described in section 2.3, page 4), the overarrows package has an alternative method to draw the arrow. This one allows the use of graphic languages such as PGF/TikZ.

Drawing arrows with TikZ requires to load the tikz package and its library `arrows.meta`. This can be simply done by passing the `tikz`[→P.15] option to the overarrows package[6]:

```
\usepackage[tikz]{overarrows}
```

To use PGF/TikZ language, the optional argument `tikz` must be passed to `\NewOverArrowCommand`[→P.15] . TikZ picture are not extensible. That's why the overarrows package provides three lengths that can be used in TikZ commands:

- `\overarrowlength`[→P.17] for the arrow length,

- `\overarrowthickness`[→P.17] and `\overarrowsmallerthickness`[→P.18] for the arrow thickness.

These lengths are computed at each utilisation of a command created with the `tikz` optional argument.

Without any other configuration, a right arrow is drawn:

```
\NewOverArrowCommand[tikz]{overtikzarrow}{}
$ \overtikzarrow{v} \qquad \overtikzarrow{AB} $
```

$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

Keys to use Tikz are described in section 4.3.4, page 24. Main keys are: `tikz options`[→P.24], `path options`[→P.25] and `path`[→P.25]. It's also possible to append settings with `add tikz options`[→P.25] and `add path options`[→P.25]. The full TikZ command used to draw the arrow can as well be entirely redefined with the key `tikz command`[→P.25]

Here is a example of an arrow drawn with TikZ[7]:

```
\NewOverArrowCommand[tikz]{overarchedleftrightarrow}{%
  add tikz options={y=\overarrowlength},
  add tikz options={line width={\overarrowsmallerthickness}},
  path options={arrows={<[scale=0.5]->[scale=0.5]}},
  path={(0,0) arc (-250:70:0.5 and 0.1)},
  center arrow,
  min length=25,
  space after arrow=0.4ex,
}
$ \overarchedleftrightarrow{v} \qquad \overarchedleftrightarrow{ABCD} $
```

$$\overleftrightarrow{v} \qquad \overleftrightarrow{ABCD}$$

---

[6]Note that the `tikz`[→P.15] option isn't mandatory to use TikZ commands in overarrows. The tikz package and its library `arrows.meta` can be loaded independently.

[7]TikZ arrows are very powerfull, but much slower to draw than the default method using assemblage of math symbols.

3.7 Drawing the arrow with LaTeX picture environment

As well as TikZ, the LaTeX `picture` environment can be used to draw the arrow. For this, the optional argument `picture` must be passed to `\NewOverArrowCommand`[→ P. 15]. Like for TikZ, the three lengths `\overarrowlength`[→ P. 17], `\overarrowthickness`[→ P. 17] and `\overarrowsmallerthickness`[→ P. 18] can be used in `picture` commands. By default, a right vector is drawn:

```
\NewOverArrowCommand[picture]{overpictarrow}{}
$ \overpictarrow{v} \qquad \overpictarrow{AB} $
```
$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

If overarrows is loaded with the option `pstarrows`[→ P. 15], the package pict2e is used and a PSTricks style vector arrows is set. This gives:

```
\NewOverArrowCommand[picture]{overpictarrow}{}
$ \overpictarrow{v} \qquad \overpictarrow{AB} $
```
$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

Keys to use LaTeX `picture` environment are described in section 4.3.5, page 25. The main keys are `picture command`[→ P. 26], `geometry`[→ P. 26] an `line thickness`[→ P. 26]. Here is an example:

```
\NewOverArrowCommand[picture]{overbandedarrow}{
  picture command={%
    \qbezier
    (0.0\overarrowlength,0)
    (0.5\overarrowlength,0)
    (0.9\overarrowlength,0.2\overarrowlength)
    \put(0.9\overarrowlength,0.2\overarrowlength)
    {\vector(2,1){0.2\overarrowlength}}
  },
  geometry={(\overarrowlength,0.4\overarrowlength)(0,0)},
  line thickness={\overarrowsmallerthickness},
  center arrow,
  space after arrow=0.4ex,
}
$ \overbandedarrow{v} \qquad \overbandedarrow{AB} $
```
$$\overrightarrow{v} \qquad \overrightarrow{AB}$$

4 User interface

4.1 Package options

The overarrows package accepts many options, given as a comma-separated list ⟨*options*⟩ at package loading: `\usepackage[`⟨*options*⟩`]{overarrows}`.

The option `esvect` is set by default. This can be overridden with `noesvect`.

4.1.1 esvect configuration

esvect

Loads the esvect package and redefines its vector commands $\vv^{\rightarrow P.18}$ through the overarrows mechanism. Original esvect \vv macro is still available with $\esvectvv^{\rightarrow P.18}$.

The esvect package provides the symbol \relbareda `-` which is smaller and often more flexible than the classic one \relbar `—`. \relbareda fits with the standard *Computer Modern* math font, but can be unsuitable with other fonts.

The esvect package also provides the right arrow command \fldr. The shape of the arrow depends on the option passed to the esvect package: `→` (option a), `→` (option b), `→` (option c), `→` (option d), `→` (option e), `→` (option f), `→` (option g) or `→` (option h). Note that by default overarrows loads the esvect package with the option f (while esvect default is d). This can be changed with one of the eight options described bellow: esvecta, esvectb, esvectc, esvectd, esvecte, esvectf, esvectg and esvecth.

This option is set by default and can be unset with noesvect.

noesvect

Prevents the loading of the esvect package and the definition of the command $\vv^{\rightarrow P.18}$.

esvecta

Loads the esvect package with the `a` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

esvectb

Loads the esvect package with the `b` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

esvectc

Loads the esvect package with the `c` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

esvectd

Loads the esvect package with the `d` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

esvecte

Loads the esvect package with the `e` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

esvectf

Loads the esvect package with the `f` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

esvectg

Loads the esvect package with the `g` option.

\fldr corresponds the to the symbol `→`. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{\mathrm{grad}}$.

Loads the esvect package with the h option.

\fldr corresponds the to the symbol →. \vv command gives : $\vec{v}$ $\overrightarrow{AB}$ $\overrightarrow{grad}$.

4.1.2 Predefined commands

The overarrows package provides sixteen predefined commands, eight with the arrow over, and eight with the arrow under. By default, theses commands are not defined, and must be activated by the corresponding option. Beware that commands are created without checking if already defined by another package (\overleftarrow, \overrightarrow, \overleftrightarrow, \underleftarrow, \underrightarrow and \underleftrightarrow are, for example, part of the amsmath package).

Three options are also available to define set of commands.

Set of commands

allcommands

Defines all sixteen predefined commands.

overcommands

Defines all eight predefined commands with arrow over.

undercommands

Defines all eight predefined commands with arrow under.

Over arrows

overrightarrow

Defines the \overrightarrow[P. 18] command: $\overrightarrow{v}$, $\overrightarrow{AB}$, $\overrightarrow{grad}$.

overleftarrow

Defines the \overleftarrow[P. 19] command: $\overleftarrow{v}$, $\overleftarrow{AB}$, $\overleftarrow{grad}$.

overleftrightarrow

Defines the \overleftrightarrow[P. 19] command: $\overleftrightarrow{v}$, $\overleftrightarrow{AB}$, $\overleftrightarrow{grad}$.

overrightharpoonup

Defines the \overrightharpoonup[P. 19] command: $\overrightharpoonup{v}$, $\overrightharpoonup{AB}$, $\overrightharpoonup{grad}$.

overrightharpoondown

Defines the \overrightharpoondown[P. 19] command: $\overrightharpoondown{v}$, $\overrightharpoondown{AB}$, $\overrightharpoondown{grad}$.

overleftharpoonup

Defines the \overleftharpoonup[P. 19] command: $\overleftharpoonup{v}$, $\overleftharpoonup{AB}$, $\overleftharpoonup{grad}$.

overleftharpoondown

Defines the `\overleftharpoondown`[→P. 19] command: $\overleftharpoondown{v}$, $\overleftharpoondown{AB}$, $\overleftharpoondown{\text{grad}}$.

overbar

Defines the `\overbar`[→P. 19] command: $\overline{v}$, $\overline{AB}$, $\overline{\text{grad}}$.

Under arrows

underrightarrow

Defines the `\underrightarrow`[→P. 19] command: $\underrightarrow{v}$, $\underrightarrow{AB}$, $\underrightarrow{\text{grad}}$.

underleftarrow

Defines the `\underleftarrow`[→P. 19] command: $\underleftarrow{v}$, $\underleftarrow{AB}$, $\underleftarrow{\text{grad}}$.

underleftrightarrow

Defines the `\underleftrightarrow`[→P. 19] command: $\underleftrightarrow{v}$, $\underleftrightarrow{AB}$, $\underleftrightarrow{\text{grad}}$.

underrightharpoonup

Defines the `\underrightharpoonup`[→P. 19] command: $\underrightharpoonup{v}$, $\underrightharpoonup{AB}$, $\underrightharpoonup{\text{grad}}$.

underrightharpoondown

Defines the `\underrightharpoondown`[→P. 20] command: $\underrightharpoondown{v}$, $\underrightharpoondown{AB}$, $\underrightharpoondown{\text{grad}}$.

underleftharpoonup

Defines the `\underleftharpoonup`[→P. 20] command: $\underleftharpoonup{v}$, $\underleftharpoonup{AB}$, $\underleftharpoonup{\text{grad}}$.

underleftharpoondown

Defines the `\underleftharpoondown`[→P. 20] command: $\underleftharpoondown{v}$, $\underleftharpoondown{AB}$, $\underleftharpoondown{\text{grad}}$.

underbar

Defines the `\underbar`[→P. 20] command: $\underbar{v}$, $\underbar{AB}$, $\underbar{\text{grad}}$.

4.1.3   Other options

old-arrows

Loads the `old-arrows` package with its option `old`. This provides the symbols `\varleftarrow` ⟵ and `\varrightarrow` ⟶, used then by default for predefined command.

When the `old-arrows` option is set, the commands `\overrightarrow`[→P. 18], `\overleftarrow`[→P. 19], `\overleftrightarrow`[→P. 19], `\underrightarrow`[→P. 19], `\underleftarrow`[→P. 19] and `\underleftrightarrow`[→P. 19] give respectively : $\overrightarrow{AB}$, $\overleftarrow{AB}$, $\overleftrightarrow{AB}$, $\underrightarrow{AB}$, $\underleftarrow{AB}$ and $\underleftrightarrow{AB}$

tikz

Loads the package tikz with its library `arrows.meta`.

Note that TikZ arrows, drawn with the `tikz` method, are always available, even if this option is not set, provided the `tikz` package and its library are loaded independently.

pstarrows

Loads the `pict2e` package, with its option `pstarrows`. Vectors using LaTeX `picture` environment gives then $\overrightarrow{AB}$ instead of $\overrightarrow{AB}$.

Note that this affect all vectors drawn in LaTeX `picture` environments, and that this setting can be changed on the fly with the commands `\pstarrows` and `\ltxarrows` from the `pict2e` package.

subscripts

Sets the default value of the key `detect subscripts`[→P. 22] to `true`.

This option also impacts the command `\vv`[→P. 18] and all predefined commands, so that they automatically use their starred variant when a subscript follows.

debug

Writes the meaning of defined commands in LaTeX log.

4.2   Commands

4.2.1   Macro for commands creation

`\NewOverArrowCommand[`⟨*method*⟩`]{`⟨*name*⟩`}{`⟨*keys*⟩`}`
`\RenewOverArrowCommand[`⟨*method*⟩`]{`⟨*name*⟩`}{`⟨*keys*⟩`}`
`\ProvideOverArrowCommand[`⟨*method*⟩`]{`⟨*name*⟩`}{`⟨*keys*⟩`}`
`\DeclareOverArrowCommand[`⟨*method*⟩`]{`⟨*name*⟩`}{`⟨*keys*⟩`}`

Creates the command `\`⟨*name*⟩ and its starred variant `\`⟨*name*⟩`*`. The starred variant `\`⟨*name*⟩`*` removes the extra end space generated by the arrow, which is suitable, as example, when a subscript follows.

`\NewOverArrowCommand` raises an error if `\`⟨*name*⟩ is already defined.

`\RenewOverArrowCommand` raises an error if `\`⟨*name*⟩ is undefined.

`\ProvideOverArrowCommand` sets `\`⟨*name*⟩ if the command is undefined and does nothing if it is already defined, without raising any error.

`\DeclareOverArrowCommand` sets `\`⟨*name*⟩, whether the command is already defined or not, without raising any error.

The ⟨*method*⟩ used to draw the arrow must be:

symb to draw the arrow by symbols assemblage (default);

tikz to draw the arrow with PGF/TikZ;

picture to draw the arrow with the LaTeX `picture` environment.

With no ⟨*method*⟩ argument, the `symb` method is chosen.

⟨*keys*⟩ is a comma-separated list of keys-values. Available keys depends of the ⟨*method*⟩ chosen and are described in section 4.3, page 20.

```
\NewOverArrowCommand[tikz]{myoverarrow}{arrows={Bar-Bar}, center arrow}
$ \myoverarrow{v} \qquad \myoverarrow{ABCD} $
```

$$\overset{\shortmid\ \shortmid}{v} \qquad \overset{\shortmid\qquad\ }{ABCD}$$

\TestOverArrow[⟨*pattern*⟩]{⟨*name*⟩}
\TestOverArrow*[⟨*pattern*⟩]{⟨*name*⟩}

Displays the result of the command ⟨*name*⟩ for patterns of various lengths and for the four math styles. A custom ⟨*pattern*⟩ can be added to the predefined ones.

The starred variant **\TestOverArrow*** displays a full report, including kerning tests of the commands \⟨*name*⟩ and \⟨*name*⟩* .

```
\TestOverArrow*[my~pattern]{vv}
```

Test of \vv and \vv* macros

\vv for different math styles

\displaystyle	\textstyle	\scriptstyle	\scriptscriptstyle
$\vec{v}$	$\vec{v}$	$\vec{v}$	$\vec{v}$
$\overrightarrow{AB}$	$\overrightarrow{AB}$	$\overrightarrow{AB}$	$\overrightarrow{AB}$
$\overrightarrow{\mathrm{grad}}$	$\overrightarrow{\mathrm{grad}}$	$\overrightarrow{\mathrm{grad}}$	$\overrightarrow{\mathrm{grad}}$
$\overrightarrow{my\ long\ vector}$	$\overrightarrow{my\ long\ vector}$	$\overrightarrow{my\ long\ vector}$	$\overrightarrow{my\ long\ vector}$
$\overrightarrow{my\ pattern}$	$\overrightarrow{my\ pattern}$	$\overrightarrow{my\ pattern}$	$\overrightarrow{my\ pattern}$

\vv kerning

$$\vec{t}\,\vec{u}\,\vec{v} \qquad \vec{\imath}\,0 \qquad \vec{v} = \vec{v}_x + \vec{v}_y + \vec{v}_z = v_x\vec{\imath} + v_y\vec{\jmath} + v_z\vec{k}$$

\vv* kerning

$$\vec{t}\vec{u}\vec{v} \qquad \vec{\imath}0 \qquad \vec{v} = \vec{v}_x + \vec{v}_y + \vec{v}_z = v_x\vec{\imath} + v_y\vec{\jmath} + v_z\vec{k}$$

4.2.2 Useful macros for symbols assemblage

Math symbols assemblage is the default method used to draw arrows. The macros `\xjoinrel` and `\smallermathstyle` are designed to help combine and format math symbols.
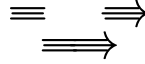
\xjoinrel[⟨*number*⟩]

Removes an horizontal space of ⟨*number*⟩ math units (`3.5 mu` by default). Must be used in math mode. Useful to assemble math symbols and create new ones.

```
\newcommand*{\triplebar}{\Relbar\xjoinrel[14]\relbar}
\newcommand*{\triplebararrow}{\Relbar\xjoinrel[15]\rightarrow}
\scalebox{2}{$ \triplebar \quad \triplebararrow $}\par
\scalebox{2}{$ \triplebar\xjoinrel\triplebararrow $}
```

$$\equiv \quad \Longrightarrow$$
$$\Longrightarrow$$

\smallermathstyle

> Applies the next math style, smaller than the current. That is:
>
> - sets \scriptstyle if the current math style is \displaystyle or \textstyle;
>
> - sets \scriptscriptstyle if the current math style is \scriptstyle;
>
> - does nothing if the current math style is \scriptscriptstyle.

```
$ \displaystyle AB \quad \textstyle AB
 \quad \scriptstyle AB \quad \scriptscriptstyle AB $\par
$ \displaystyle AB \quad \smallermathstyle AB
 \quad \smallermathstyle AB \quad \smallermathstyle AB $
```

$$AB \quad AB \quad {\scriptstyle AB} \quad {\scriptscriptstyle AB}$$
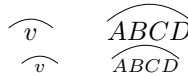$$AB \quad {\scriptstyle AB} \quad {\scriptscriptstyle AB} \quad {\scriptscriptstyle AB}$$

4.2.3 Useful lengths for TikZ or `picture` environment

Arrows drawn with graphic languages, like PGF/TikZ or the LaTeX `picture` environment, are not extensible. The three lengths \overarrowlength, \overarrowthickness and \overarrowsmallerthickness are computed at each utilisation of a command set with the `tikz` or `picture` method, so they can be used in drawing commands.

```
\NewOverArrowCommand[tikz]{overparabola}{%
  path options={x=\overarrowlength, line width=\overarrowsmallerthickness},
  path={(0,0) parabola[parabola height=0.2\overarrowlength] (1,0)},
  arrows={-}, center arrow, min length=30,
}
$\displaystyle \overparabola{v} \qquad \overparabola{ABCD} $\par
$\scriptstyle \overparabola{v} \qquad \overparabola{ABCD} $\par
```

$$\overparabola{v} \qquad \overparabola{ABCD}$$
$$\overparabola{v} \qquad \overparabola{ABCD}$$

\overarrowlength

> Is set to the width of the arrow command content, or, if larger, to the minimal arrow length set through the key `min length`[→ P. 20].

\overarrowthickness

> Is set to the default rule thickness of the current math style. That is:
>
> - \fontdimen 8 \textfont 3 in \displaystyle or \textstyle;
>
> - \fontdimen 8 \scriptfont 3 in \scriptstyle;
>
> - \fontdimen 8 \scriptscriptfont 3 in \scriptscriptstyle.

\overarrowsmallerthickness

Is set to the default rule thickness of the next smaller math style. That is:

- \fontdimen 8 \scriptfont 3 in \displaystyle or \textstyle;
- \fontdimen 8 \scriptscriptfont 3 in \scriptstyle or \scriptscriptstyle.

4.2.4 Vectors macros

The macro \vv, dedicated to vectors, is automatically defined when the option esvect$^{\rightarrow P.\,11}$ is set (which is the default). It is a clone of the \vv command provided by the esvect package, but its starred variant has a correct kerning when followed by a subscript.

\vv{⟨content⟩}
\vv*{⟨content⟩}

Draws a vector arrow upon math ⟨content⟩. The shape of the arrow depends on the corresponding options described in section 4.1.1, page 11 : esvecta$^{\rightarrow P.\,12}$, esvectb$^{\rightarrow P.\,12}$, esvectc$^{\rightarrow P.\,12}$, esvectd$^{\rightarrow P.\,12}$, esvecte$^{\rightarrow P.\,12}$, esvectf$^{\rightarrow P.\,12}$, esvectg$^{\rightarrow P.\,12}$, esvecth$^{\rightarrow P.\,13}$.

The starred variant \vv* suppresses the end space created by the arrow.

```
$ \vv{\imath}_{0} \quad \vv{e}_r \quad \vv{L}_\Delta $\par
$ \vv*{\imath}_{0} \quad \vv*{e}_r \quad \vv*{L}_\Delta $
```

$$\vec{\imath}\,_0 \quad \vec{e}_r \quad \vec{L}_\Delta$$
$$\vec{\imath}_0 \quad \vec{e}_r \quad \vec{L}_\Delta$$

\esvectvv

Is simply the backup of the original esvect \vv command.

```
$ \esvectvv{\imath}_{0} \quad \esvectvv{e}_{r} \quad \esvectvv{L}_\Delta $\par
$ \esvectvv*{\imath}{0} \quad \esvectvv*{e}{r} \quad \esvectvv*{L}{\Delta} $
```

$$\vec{\imath}\,_0 \quad \vec{e}_r \quad \vec{L}_\Delta$$
$$\vec{\imath}\,_0 \quad \vec{e}_r \quad \vec{L}_\Delta$$

4.2.5 Predefined commands

Predefined commands are defined if the corresponding option is set (see section 4.1.2, page 13). The commands \overrightarrow, \overleftarrow, \overleftrightarrow, \underrightarrow, \underleftarrow and \underleftrightarrow are affected by the option old-arrows$^{\rightarrow P.\,14}$.

Over arrows

\overrightarrow

$$\vec{v} \qquad \overrightarrow{AB} \qquad \overrightarrow{\mathrm{grad}}$$

The shape of the arrow is smaller if the option old-arrows$^{\rightarrow P.\,14}$ is set.

\overleftarrow

$$\overleftarrow{v} \qquad \overleftarrow{AB} \qquad \overleftarrow{\text{grad}}$$

The shape of the arrow is smaller if the option `old-arrows`^{→P.14} is set.

\overleftrightarrow

$$\overleftrightarrow{v} \qquad \overleftrightarrow{AB} \qquad \overleftrightarrow{\text{grad}}$$

The shape of the arrows is smaller if the option `old-arrows`^{→P.14} is set.

\overrightharpoonup

$$\overrightharpoonup{v} \qquad \overrightharpoonup{AB} \qquad \overrightharpoonup{\text{grad}}$$

\overrightharpoondown

$$\overrightharpoondown{v} \qquad \overrightharpoondown{AB} \qquad \overrightharpoondown{\text{grad}}$$

\overleftharpoonup

$$\overleftharpoonup{v} \qquad \overleftharpoonup{AB} \qquad \overleftharpoonup{\text{grad}}$$

\overleftharpoondown

$$\overleftharpoondown{v} \qquad \overleftharpoondown{AB} \qquad \overleftharpoondown{\text{grad}}$$

\overbar

$$\overline{v} \qquad \overline{AB} \qquad \overline{\text{grad}}$$

Under arrows

\underrightarrow

$$\underrightarrow{v} \qquad \underrightarrow{AB} \qquad \underrightarrow{\text{grad}}$$

The shape of the arrow is smaller if the option `old-arrows`^{→P.14} is set.

\underleftarrow

$$\underleftarrow{v} \qquad \underleftarrow{AB} \qquad \underleftarrow{\text{grad}}$$

The shape of the arrow is smaller if the option `old-arrows`^{→P.14} is set.

\underleftrightarrow

$$\underleftrightarrow{v} \qquad \underleftrightarrow{AB} \qquad \underleftrightarrow{\text{grad}}$$

The shape of the arrows is smaller if the option `old-arrows`^{→P.14} is set.

\underrightharpoonup

$$\underrightharpoonup{v} \qquad \underrightharpoonup{AB} \qquad \underrightharpoonup{\text{grad}}$$

19

\underrightharpoondown

$$\underset{\longrightarrow}{v} \qquad \underset{\longrightarrow}{AB} \qquad \underset{\longrightarrow}{\mathrm{grad}}$$

\underleftharpoonup

$$\underset{\longleftharpoonup}{v} \qquad \underset{\longleftharpoonup}{AB} \qquad \underset{\longleftharpoonup}{\mathrm{grad}}$$

\underleftharpoondown

$$\underset{\longleftharpoondown}{v} \qquad \underset{\longleftharpoondown}{AB} \qquad \underset{\longleftharpoondown}{\mathrm{grad}}$$

\underbar

$$\underline{v} \qquad \underline{AB} \qquad \underline{\mathrm{grad}}$$

4.3 Keys

The customisation of arrows is done at command creation through a key-value interface provided by the `pgfkeys` package (with `/overarrows/` as key path).

4.3.1 Arrow position and length settings

These keys are available whatever the method chosen at command creation (see section 4.2.1, page 15 for the documentation of commands creation).

Length

min length={⟨*number*⟩}   (no default, see below for the initial value)

> Sets the minimal arrow length to ⟨*number*⟩ math units. The arrow length is set from content width, or, if larger, to this value.
>
> The initial value of `min length` depends on the ⟨*method*⟩ chosen at command creation (see section 4.2.1, page 15 for the documentation of commands creation):
>
> - ⟨*number*⟩ = 0 for the `symb` method (default);
> - ⟨*number*⟩ = 12 for the `tikz` method;
> - ⟨*number*⟩ = 18 for the `picture` method.

```
\NewOverArrowCommand{overlongarrow}{min length=50}
$ \overlongarrow{v} \qquad \overlongarrow{ABCDEF} $
```
$$\overset{\longrightarrow}{v} \qquad \overset{\longrightarrow}{ABCDEF}$$

Placement

arrow under   (default `autoconfig`, initially unset)
arrow under=autoconfig|noconfig

> Places the arrow under, instead of over.

arrow under or **arrow under=autoconfig** also configures suitably the key detect subscripts^{→ P. 22} to **false** and the key before arrow^{→ P. 22} to get an additional space over the arrow.

arrow under=noconfig does not do any additional configuration.

```
\NewOverArrowCommand{underhooks}{%
  start={\lhook}, end={\rhook}, trim=1,
  arrow under, shift leftright=-4,
}
$ \underhooks{v} \qquad \underhooks{AB} $
```

$$\underset{\underline{}}{v} \qquad \underline{AB}$$

Horizontal shifts

shift left={⟨*number*⟩}   (no default, initially 2)

Shifts the left side of the arrow by ⟨*number*⟩ math units (positive number means a shift to the right).

shift right={⟨*number*⟩}   (no default, see below for the initial value)

Shifts the right side of the arrow by ⟨*number*⟩ math units (positive number means a shift to the left).

The initial value of **shift right** depends on the ⟨*method*⟩ chosen at command creation (see section 4.2.1, page 15 for the documentation of commands creation):

- ⟨*number*⟩ = 0 for the **symb** method (default);
- ⟨*number*⟩ = -2 for the **tikz** and **picture** methods.

```
\NewOverArrowCommand{lookback}{%
  start={\leftarrow}, end={\rightharpoondown},
  shift left=-50, shift right=-10,
}
$ \lookback{\text{look back}} $
```

$$\overleftarrow{\text{look back}}$$

shift leftright=[⟨*number*⟩]   (no default)

Sets **shift left** and **shift right** to the same ⟨*number*⟩ value.

center arrow

Sets **shift left** and **shift right** to zero.

left arrow   (default 2)

Sets **shift left** to zero and **shift right** to ⟨*number*⟩.

right arrow   (default 2)

Sets **shift right** to zero and **shift left** to ⟨*number*⟩.

Vertical adjunct

before arrow=`{`⟨*vertical material*⟩`}`   (initially empty)
after arrow=`{`⟨*vertical material*⟩`}`   (initially empty)

    Adds the ⟨*vertical material*⟩ before or after the arrow.

    Over and under arrow commands are typeset through the TEX `\ialign` command, which aligns contents, like a tabular. The ⟨*vertical material*⟩ is inserted *between* the rows, with TEX `\noalign` command.

    These keys are essentially used to add some extra space between the arrow and the content of the command. They can be set in a handier way with the keys `space before arrow` and `space after arrow`.

space before arrow=`{`⟨*length*⟩`}`   (no default)

    Adds a space of ⟨*length*⟩ before the arrow. This sets the keys `before arrow`.

space after arrow=`{`⟨*length*⟩`}`   (no default)

    Adds a space of ⟨*length*⟩ after the arrow. This sets the keys `after arrow`.

```
\NewOverArrowCommand{overharpoonsdown}{%
  start=\leftharpoondown, end=\rightharpoondown, center arrow,
  space before arrow=-0.2ex, space after arrow=0.3ex,
}
$ \dot{\overharpoonsdown{v}} \qquad \ddot{\overharpoonsdown{AB}}$
```
$$\dot{\overset{\leftharpoondown\rightharpoondown}{v}} \qquad \ddot{\overset{\leftharpoondown\rightharpoondown}{AB}}$$

4.3.2 Subscripts detection setting

This key is available whatever the method chosen at command creation (see section 4.2.1, page 15 for the documentation of commands creation).

detect subscripts=`true|false`   (default `true`, see below for the initial value)

    Removes automatically the extra end space created by the arrow, if a subscript immediately follows the command.

    By default, the initial value of `detect subscripts` is `false`. When the option `subscripts`^{→P. 15} is set, the initial value of `detect subscripts` is `true`.

```
\NewOverArrowCommand{autosub}{detect subscripts}
$ \imath_0 \qquad \autosub{\imath}_0 \qquad
 {\autosub{\imath}}_0 \qquad {\autosub*{\imath}}_0 $
```
$$\imath_0 \qquad \overrightarrow{\imath}_0 \qquad \overrightarrow{\imath}\,_0 \qquad \overrightarrow{\imath_0}$$

4.3.3 Symbols assemblage settings

The following keys are available for arrows drawn with the default `symb` method (see section 4.2.1, page 15 for the documentation of commands creation).

start=`{`⟨*command*⟩`}`   (no default, initially `\relbar`)
middle=`{`⟨*command*⟩`}`   (no default, initially set by `middle config=auto`)
end=`{`⟨*command*⟩`}`   (no default, see below for the initial value)

    Sets the ⟨*command*⟩ used to draw the start (left), middle (center) or end (right) part of the arrow. The `middle` one is repeated, if necessary, to extend

the arrow. It is set, initially by `middle config=auto`. By default, the `end` symbols is initially `\rightarrow` →. When the option `old-arrows`[→P. 14] is set, the initial value of `end` is `\varrightarrow` →.

`start` and `end` symbols are typeset in the same group. `middle` is typeset alone. This means that, if a command, like `\smallermathstyle`[→P. 17], is used to alter the symbols, it should be applied both to `start` and `middle` (but not to `end`).

```
\NewOverArrowCommand{smalleroverrightarrow}{%
  start={\smallermathstyle\relbar},
  middle={\smallermathstyle\relbareda},
  end={\rightarrow},
  space after arrow={0.2ex},
}
$ \smalleroverrightarrow{v} \qquad \smalleroverrightarrow{AB} $
```

$$\vec{v} \qquad \overrightarrow{AB}$$

`trim start`=`{`⟨*number*⟩`}`   (no default, initially 7)

Trims ⟨*number*⟩ math units from the right side of the `start`[→P. 22] symbol.

`trim middle`=`{`⟨*number*⟩`}`   (no default, initially set by `middle config=auto`)

Trims ⟨*number*⟩ math units from both left and right sides of the `middle`[→P. 22] symbol.

`trim end`=`{`⟨*number*⟩`}`   (no default, initially 7)

Trims ⟨*number*⟩ math units from the left side of the `end`[→P. 22] symbol.

`trim`=`{`⟨*number*⟩`}`   (no default)

Sets `trim start`, `trim middle` and `trim end` to the same ⟨*number*⟩ value.

`no trimming`

Clears `trim start`, `trim middle` and `trim end`.

`middle config`=`auto|relbar|relbareda`   (no default)

Sets a suitable configuration for the keys `middle`[→P. 22] and `trim middle`:

For `middle config = relbar`, `middle`[→P. 22] is set to `\relbar` — and `trim middle` to 2.5.

For `middle config = relbareda`, `middle`[→P. 22] is set to `\relbareda` - and `trim middle` to 1.

For `middle config = auto`, `middle`[→P. 22] is set with `middle config = relabareda` if the option `esvect`[→P. 11] is set (which is the default) and `middle config = relabar` if not.

`amsmath`   (default mimic)
`amsmath`=`mimic|strict`

Loads a configuration coherent with `amsmath` `\overrightarrow` command.

`amsmath` or `amsmath=mimic` sets the corresponding keys suitably:

23

<div style="text-align:center">

```
start={\relbar}        middle={\relbar}        end={\rightarrow}
trim start=7           trim middle=2           trim end=7
shift leftright=0      after arrow={}          before arrow={}
```

</div>

amsmath=strict makes, in addition, the command uses the internal macros of amsmath `\overrightarrow` (no trimming, fill macro=`{\arrowfill@}`, stack macro=`{\overarrow@}`). Note that many configuration keys becomes ineffective.

esvect (default `mimic`)
esvect=mimic|strict

Loads a configuration coherent with amsmath `\vv` command.

esvect or esvect=mimic sets the corresponding keys suitably:

<div style="text-align:center">

```
start={\relbaredd}        middle={\relbareda}        end={\fldr}
trim start=1.5            trim middle=0              trim end=1.5
space before arrow=-.7pt  space after arrow=-.3pt    right arrow=2
```

</div>

esvect=strict makes, in addition, the command uses the internal macros of esvect `\vv` (no trimming, fill macro=`{\traitfill@}`, stack macro=`{\overvect@}`). Note that many configuration keys becomes ineffective.

4.3.4 TikZ settings

If, at command creation (see section 4.2.1, page 15 for the documentation of commands creation), the `tikz` method is chosen, then the arrow is drawn by the command:

<div style="text-align:center">

`\tikz[tikz options]{tikz command}`

</div>

where `tikz options` and `tikz command`[→P. 25] are two keys described below. When `tikz command` is let unset, the drawing command turns into:

<div style="text-align:center">

`\tikz[tikz options]{\draw[path options] path;}`

</div>

The best way to customise `tikz` arrows is then to set the keys `tikz options`, `path options`[→P. 25] and `path`[→P. 25], preferably through the handy alternatives: `add tikz options`[→P. 25], `add path options`[→P. 25], `arrows`[→P. 25], `line thickness`[→P. 25] or `thinner`[→P. 25].

```
\NewOverArrowCommand[tikz]{overdotteddoublearrow}{%
  add tikz options={blue}, add path options={densely dotted},
  arrows={->[scale=0.5]>[scale=0.5]}, thinner,
  min length=20, space after arrow={0.3ex},
}
$ \overdotteddoublearrow{v} \qquad \overdotteddoublearrow{AB} $
```

<div style="text-align:center">

$\overset{\cdots\!\twoheadrightarrow}{v} \qquad \overset{\cdots\!\twoheadrightarrow}{AB}$

</div>

The following keys are available when the `tikz` method is chosen.

tikz options={⟨*TikZ options*⟩}
<div style="text-align:right">(no default, initially `x=\overarrowlength`, `line width=\overarrowthickness`)</div>

Sets TikZ options to ⟨*TikZ options*⟩.

path options={⟨*path options*⟩}

(no default, initially `arrows=-Classical TikZ Rightarrow, cap=round`)

Sets TikZ path options to ⟨*path options*⟩.

path={⟨*path specification*⟩}   (no default, initially `(0,0)--(1,0)`)

Sets TikZ path specification to ⟨*path*⟩ (the ending semicolon is automatically appended).

add tikz options={⟨*TikZ options*⟩}   (no default)

Appends the options ⟨*TikZ options*⟩ to the key `tikz options`^{→P. 24}.

add path options={⟨*path options*⟩}   (no default)

Appends the options ⟨*path options*⟩ to the key `path options`.

arrows={⟨*arrow specification*⟩}   (no default)

Appends the option `arrows={`⟨*arrow specification*⟩`}` to the key `path options`.

line thickness={⟨*length*⟩}   (no default)

Appends the option `line width={`⟨*length*⟩`}` to the key `path options`.

thinner

Sets the keys `line thickness` with `\overarrowsmallerthickness`.

tikz command={⟨*TikZ command*⟩}   (initially unset)

Sets the ⟨*TikZ command*⟩ used to draw the arrow. If left unset, the value `\draw[path options] path;` is used.

4.3.5 Picture environment settings

If, at command creation (see section 4.2.1, page 15 for the documentation of commands creation), the `picture` method is chosen, then the arrow is drawn with by:

```
\begin{picture}geometry%
  \linethickness{line thickness}%
  picture command%
\end{picture}%
```

where `geometry`^{→P. 26}, `line thickness`^{→P. 26} and `picture command` are three keys described below.

25

```
% ^^A \arc and \roundcap commands are from the pict2e package
% ^^A this example needs \usepackage{pict2e} in the preamble
\NewOverArrowCommand[picture]{overarc}{%
  picture command={%
    \roundcap
    \put(0.5\overarrowlength,0){\arc[180,0]{0.6\overarrowlength}}
  },
  geometry={%
    (1.2\overarrowlength,0.5\overarrowlength)(-0.1\overarrowlength,0.2ex)
  },
  thinner, center arrow,
}
$ \overarc{v} \qquad \overarc{AB} $
```

$$\overparen{v} \qquad \overparen{AB}$$

The following keys are available when the `picture` method is chosen.

picture command={⟨*picture command*⟩}

(no default, initially \put(0,0){\vector(1,0){\overarrowlength}})

Sets picture command to ⟨*picture command*⟩.

geometry={⟨*picture geometry specification*⟩}

(no default, initially (\overarrowlength,1ex)(0,-0.5ex))

Sets picture geometry to ⟨*picture geometry specification*⟩.

line thickness={⟨*length*⟩}   (no default)

Sets the picture line thickness to ⟨*length*⟩.

thinner   (no default)

Sets the keys `line thickness` with `\overarrowsmallerthickness`.

4.4 Advanced commands and keys

The following commands and keys are used in the implementation of the overar-rows package. They can also be employed for an advanced configuration of the commands created, although unnecessary in the vast majority of cases.

4.4.1 Advanced commands

\SetOverArrowsMethod[⟨*stack mechanism*⟩]{⟨*name*⟩}[⟨*pre code*⟩]{⟨*keys def*⟩}
\SetOverArrowsMethod*{⟨*name*⟩}[⟨*pre code*⟩]{⟨*keys def*⟩}

Defines the method ⟨*name*⟩, to be used in commands \NewOverArrowCommand[→P. 15],
\RenewOverArrowCommand[→P. 15], \ProvideOverArrowCommand[→P. 15] or \DeclareOverArrowCommand
When the ⟨*name*⟩ method is chosen, corresponding keys are defined by ⟨*keys
def*⟩. This must set, in particular, the keys `no stack macro hook`[→P. 27] and
`no arrow macro hook`[→P. 27]. Optional code ⟨*pre code*⟩ is evaluated before the
keys definition.

The unstarred variant automatically defines the key `no stack macro hook`[→P. 27],
according to the value of the optional ⟨*stack mechanism*⟩. This one must be:

26

fill if `arrow macro`[→P.27] creates extensible arrows (typically with `\cleaders`). In this case, the arrow macro (defined by `no arrow macro hook`[→P.27]) is called with the math style, passed as argument (it can be, for example, the macro `\rightarrowfill@` used by amsmath `\overrightarrow`). `fill` is the mechanism used by the `symb` method.

lens if `arrow macro` creates fixed-length arrows, and needs the computation of lengths `\overarrowlength`[→P.17], `\overarrowthickness`[→P.17] and `\overarrowsmallerthickness`[→P.18]. In this case, the arrow macro (defined by `no arrow macro hook`) is called without argument. `lens` is the mechanism used by the `tikz` and `picture` methods.

Without optional ⟨*stack mechanism*⟩, `fill` is used. The starred variant does not set the key `no stack macro hook`.

4.4.2 Advanced keys

stack macro={⟨*stack definition*⟩}   (no default, initially unset)

Defines the stack macro to be ⟨*stack definition*⟩. Stack macro is a command which takes three arguments: the arrow macro set by `arrow macro`, the math style, and the command content (under or over the arrow). ⟨*stack definition*⟩ can be, for example, the macro `\overarrow@` used by amsmath `\overrightarrow`.

arrow macro={⟨*arrow definition*⟩}   (no default, initially unset)

Defines the arrow macro (used in the stack macro) by to be ⟨*arrow definition*⟩.

no stack macro hook={⟨*code*⟩}   (no default)

Sets the ⟨*code*⟩ executed if `stack macro` is left unset, after user evaluation of ⟨*keys*⟩ in `\NewOverArrowCommand`[→P.15], `\RenewOverArrowCommand`[→P.15], `\ProvideOverArrowCommand`[→P.15] or `\DeclareOverArrowCommand`[→P.15].

⟨*code*⟩ must configure `stack macro` accordingly to the user keys setting.

no arrow macro hook={⟨*code*⟩}   (no default)

Sets the ⟨*code*⟩ executed if `arrow macro` is left unset, after user evaluation of ⟨*keys*⟩ in `\NewOverArrowCommand`[→P.15], `\RenewOverArrowCommand`[→P.15], `\ProvideOverArrowCommand`[→P.15] or `\DeclareOverArrowCommand`[→P.15].

⟨*code*⟩ must configure `arrow macro` accordingly to the user keys setting.

fill macro={⟨*definition*⟩}   (no default, initially unset)

Defines the fill macro to be ⟨*definition*⟩. The fill macro is used by arrows created with the `symb` method, to set `arrow macro` in `no arrow macro hook`. It is called with fours arguments: start, middle and end symbols used to draw the arrow, and the math style. ⟨*definition*⟩ can be, for example, the macro `\arrowfill@` used by amsmath `\overrightarrow`.

5 Complements

5.1 Math font issue

If the math font differs from the default *Computer Modern*, arrow drawn with the `symb` method may have a central part of the arrow with inappropriate position or line width. This is because the default symbol used for the arrow line is `\relbareda` - from the esvect package. This can be fixed with the `noesvect`$^{\rightarrow P.\,12}$ option.

5.2 Package dependencies

The following packages are used by overarrows:

- amsmath

- etoolbox

- pgfkeys

- esvect (unless the option `noesvect`$^{\rightarrow P.\,12}$ is used)

- old-arrows (when the option `old-arrows`$^{\rightarrow P.\,14}$ is used)

- tikz (when the `tikz` method or the option `tikz`$^{\rightarrow P.\,15}$ is used)

- pict2e (when the option `pstarrows`$^{\rightarrow P.\,15}$ is used)

LaTeX distributions prior to 2020/10/01 must load the xparse package before overarrows.

5.3 Alternatives

esvect package (`https://www.ctan.org/pkg/esvect`), by Eddie Saudrais, provides the fine vector macro `\vv`. This package is loaded by default by overarrows.

letterswitharrows package (`https://www.ctan.org/pkg/letterswitharrows`), by Max Teegen, provides left and right over arrows commands, which can extend to multiple characters.

overrightarrow package (`https://www.ctan.org/pkg/overrightarrow`), by Robin Fairbairns, provides the `\Overrightarrow` which is an amalgam of `\overrightarrow` and `\Rightarrow`.

harpoon package (`https://ctan.org/pkg/harpoon`), by Tobias Kuipers, provides over- and under-harpoon symbol commands.

5.4 Changelog

v1.0 Initial version.

6 Implementation

Management of options

Declaration of conditionals

```
1  \newif\ifovar@option@oldarrows@
2  \newif\ifovar@option@esvect@ \ovar@option@esvect@true \PassOptionsToPackage{f}{esvect}
3  \newif\ifovar@option@tikz@
4  \newif\ifovar@option@pstarrows@
5  \newif\ifovar@detectsubscripts@
6  \newif\ifovar@option@debug@
```

Following conditionals are for predefined commands.

```
7   \newif\ifovar@option@overrightarrow@
8   \newif\ifovar@option@underrightarrow@
9   \newif\ifovar@option@overleftarrow@
10  \newif\ifovar@option@underleftarrow@
11  \newif\ifovar@option@overleftrightarrow@
12  \newif\ifovar@option@underleftrightarrow@
13  \newif\ifovar@option@overrightharpoonup@
14  \newif\ifovar@option@underrightharpoonup@
15  \newif\ifovar@option@overrightharpoondown@
16  \newif\ifovar@option@underrightharpoondown@
17  \newif\ifovar@option@overleftharpoonup@
18  \newif\ifovar@option@underleftharpoonup@
19  \newif\ifovar@option@overleftharpoondown@
20  \newif\ifovar@option@underleftharpoondown@
21  \newif\ifovar@option@overbar@
22  \newif\ifovar@option@underbar@
```

Declaration of options

```
23  \DeclareOption{esvect}{\ovar@option@esvect@true}
24  \DeclareOption{noesvect}{\ovar@option@esvect@false}
25  \DeclareOption{esvecta}{\ovar@option@esvect@true\PassOptionsToPackage{a}{esvect}}
26  \DeclareOption{esvectb}{\ovar@option@esvect@true\PassOptionsToPackage{b}{esvect}}
27  \DeclareOption{esvectc}{\ovar@option@esvect@true\PassOptionsToPackage{c}{esvect}}
28  \DeclareOption{esvectd}{\ovar@option@esvect@true\PassOptionsToPackage{d}{esvect}}
29  \DeclareOption{esvecte}{\ovar@option@esvect@true\PassOptionsToPackage{e}{esvect}}
30  \DeclareOption{esvectf}{\ovar@option@esvect@true\PassOptionsToPackage{f}{esvect}}
31  \DeclareOption{esvectg}{\ovar@option@esvect@true\PassOptionsToPackage{g}{esvect}}
32  \DeclareOption{esvecth}{\ovar@option@esvect@true\PassOptionsToPackage{h}{esvect}}
33  \DeclareOption{old-arrows}{\ovar@option@oldarrows@true}
34  \DeclareOption{tikz}{\ovar@option@tikz@true}
35  \DeclareOption{pstarrows}{\ovar@option@pstarrows@true}
36  \DeclareOption{subscripts}{\ovar@detectsubscripts@true}
37  \DeclareOption{debug}{\ovar@option@debug@true}
```

Following options are for predefined commands.

```
38  \DeclareOption{overrightarrow}{\ovar@option@overrightarrow@true}
39  \DeclareOption{underrightarrow}{\ovar@option@overrightarrow@true}
40  \DeclareOption{overleftarrow}{\ovar@option@overleftarrow@true}
41  \DeclareOption{underleftarrow}{\ovar@option@overleftarrow@true}
42  \DeclareOption{overleftrightarrow}{\ovar@option@overleftrightarrow@true}
43  \DeclareOption{underleftrightarrow}{\ovar@option@overleftrightarrow@true}
44  \DeclareOption{overrightharpoonup}{\ovar@option@overrightharpoonup@true}
45  \DeclareOption{underrightharpoonup}{\ovar@option@overrightharpoonup@true}
46  \DeclareOption{overrightharpoondown}{\ovar@option@overrightharpoondown@true}
47  \DeclareOption{underrightharpoondown}{\ovar@option@overrightharpoondown@true}
48  \DeclareOption{overleftharpoonup}{\ovar@option@overleftharpoonup@true}
```

```
49  \DeclareOption{underleftharpoonup}{\ovar@option@overleftharpoonup@true}
50  \DeclareOption{overleftharpoondown}{\ovar@option@overleftharpoondown@true}
51  \DeclareOption{underleftharpoondown}{\ovar@option@overleftharpoondown@true}
52  \DeclareOption{overbar}{\ovar@option@overbar@true}
53  \DeclareOption{underbar}{\ovar@option@overbar@true}
```

Following options are for sets of predefined commands.

```
54  \DeclareOption{overcommands}{%
55    \ovar@option@overrightarrow@true
56    \ovar@option@overleftarrow@true
57    \ovar@option@overleftrightarrow@true
58    \ovar@option@overrightharpoonup@true
59    \ovar@option@overrightharpoondown@true
60    \ovar@option@overleftharpoonup@true
61    \ovar@option@overleftharpoondown@true
62    \ovar@option@overbar@true
63  }
64  \DeclareOption{undercommands}{%
65    \ovar@option@underrightarrow@true
66    \ovar@option@underleftarrow@true
67    \ovar@option@underleftrightarrow@true
68    \ovar@option@underrightharpoonup@true
69    \ovar@option@underrightharpoondown@true
70    \ovar@option@underleftharpoonup@true
71    \ovar@option@underleftharpoondown@true
72    \ovar@option@underbar@true
73  }
74  \DeclareOption{allcommands}{%
75    \ovar@option@overrightarrow@true
76    \ovar@option@underrightarrow@true
77    \ovar@option@overleftarrow@true
78    \ovar@option@underleftarrow@true
79    \ovar@option@overleftrightarrow@true
80    \ovar@option@underleftrightarrow@true
81    \ovar@option@overrightharpoonup@true
82    \ovar@option@underrightharpoonup@true
83    \ovar@option@overrightharpoondown@true
84    \ovar@option@underrightharpoondown@true
85    \ovar@option@overleftharpoonup@true
86    \ovar@option@underleftharpoonup@true
87    \ovar@option@overleftharpoondown@true
88    \ovar@option@underleftharpoondown@true
89    \ovar@option@overbar@true
90    \ovar@option@underbar@true
91  }
```

Options processing

```
92  \DeclareOption*{\PackageWarning{overarrows}{Unknown option: '\CurrentOption'}}
93  \ProcessOptions\relax
```

Package dependencies

LaTeX distributions prior to 2020/10/01 must add the xparse package.

```
94  \RequirePackage{amsmath}
95  \RequirePackage{etoolbox}
```

Option `old-arrows`^{→P. 14}. Configuration of arrows used for predefined commands.

```
96  \let\ovar@rightarrow\rightarrow
97  \let\ovar@leftarrow\leftarrow
98  \ifovar@option@oldarrows@
```

```
99     \RequirePackage[old]{old-arrows}
100    \let\ovar@rightarrow\varrightarrow
101    \let\ovar@leftarrow\varleftarrow
102  \fi
```

Option `esvect`[→P. 11].

```
103  \ifovar@option@esvect@
104    \RequirePackage{esvect}
105  \fi
```

Option `tikz`[→P. 15].

```
106  \ifovar@option@tikz@
107    \RequirePackage{tikz}
108    \usetikzlibrary{arrows.meta}
109  \fi
```

Option `pstarrows`[→P. 15].

```
110  \ifovar@option@pstarrows@
111    \RequirePackage[pstarrows]{pict2e}
112  \fi
```

Management of keys

Family declaration and setters

```
113  \RequirePackage{pgfkeys}
114  \pgfkeys{overarrows/.is family}
```

`\ovar@set`
```
115  \newcommand{\ovar@set}[1]{\pgfqkeys{/overarrows}{#1}}
```

`\SetOverArrowsMethod`
```
116  \NewDocumentCommand{\SetOverArrowsMethod}{ s O{fill} m O{} m }{%
117    \IfBooleanTF{#1}{%
118      \csgdef{ovar@set@#3}{#4\ovar@set{#5}}%
119    }{%
120      \csgdef{ovar@set@#3}{#4\ovar@set{%
121        no stack macro hook/.code={%
122          \ovar@set{stack macro/.expanded={%
123            \expandafter\expandonce\csname ovar@stack@#2\endcsname%
124            {\expandonce\ovar@length@min}%
125            {\expandonce\ovar@before@arrow}{\expandonce\ovar@after@arrow}%
126          }}%
127        },#5}}%
128    }%
129  }
```

Common keys

```
130  \SetOverArrowsMethod*{common}[\undef{\ovar@macro@stack}\undef{\ovar@macro@arrow}]{%
131    detect subscripts/.is if=ovar@detectsubscripts@,
```

`stack macro`[→P. 27] and `arrow macro`[→P. 27].

```
132    stack macro/.store in=\ovar@macro@stack,
133    arrow macro/.store in=\ovar@macro@arrow,
134    stack macro/.value required,
135    arrow macro/.value required,
```

`no stack macro hook`[→P. 27], `no arrow macro hook`[→P. 27]. These two keys must be redefined by the command `\ovar@set@`⟨*method*⟩.

```
136    no stack macro hook/.code={%
137      \PackageError{overarrows}{Undefined stack macro}
138      {The requested method is perhaps mispelled}
```

```
139     },
140     no arrow macro hook/.code={%
141       \PackageError{overarrows}{Undefined arrow macro}
142       {The requested method is perhaps mispelled}
143     },
```

min length$^{\rightarrow P.\,20}$.

```
144     min length/.store in=\ovar@length@min,
145     min length/.value required,
146     min length=0,
```

before arrow$^{\rightarrow P.\,22}$, after arrow$^{\rightarrow P.\,22}$, space before arrow$^{\rightarrow P.\,22}$, space after
arrow$^{\rightarrow P.\,22}$.

```
147     before arrow/.store in=\ovar@before@arrow,
148     after arrow/.store in=\ovar@after@arrow,
149     before arrow/.value required,
150     after arrow/.value required,
151     before arrow=\empty,
152     after arrow=\empty,
153     space before arrow/.code=\pgfkeysalso{before arrow={\kern ##1}},
154     space after arrow/.code=\pgfkeysalso{after arrow={\kern ##1}},
```

shift left$^{\rightarrow P.\,21}$, shift right$^{\rightarrow P.\,21}$, shift leftright$^{\rightarrow P.\,21}$, center arrow$^{\rightarrow P.\,21}$,
left arrow$^{\rightarrow P.\,21}$, right arrow$^{\rightarrow P.\,21}$.

```
155     shift left/.store in=\ovar@shift@left,
156     shift right/.store in=\ovar@shift@right,
157     shift left/.value required,
158     shift right/.value required,
159     shift leftright/.code=\pgfkeysalso{%
160       shift left=##1, shift right=##1,
161     },
162     center arrow/.code=\pgfkeysalso{shift leftright=0},
163     shift leftright/.value required,
164     center arrow/.value forbidden,
165     left arrow/.code=\pgfkeysalso{%
166       shift left=0, shift right=##1,
167     },
168     right arrow/.code=\pgfkeysalso{%
169       shift left=##1, shift right=0,
170     },
171     left arrow/.default=2,
172     right arrow/.default=2,
173     right arrow,
```

arrow under$^{\rightarrow P.\,20}$.

```
174     arrow under/.is choice,
175     arrow under/noconfig/.code={
176       \def\ovar@stack@fill{\ovar@stackunder@fill}
177       \def\ovar@stack@lens{\ovar@stackunder@lens}
178     },
179     arrow under/autoconfig/.code={
180       \pgfkeysalso{%
181         arrow under=noconfig,
182         detect subscripts=false,
183         before arrow={\kern 1.3\ex@\relax},% like underarrow@ from amsmath
184       }
185     },
186     arrow under/.default=autoconfig,
187 }
```

Keys for the symb method

```
188  \SetOverArrowsMethod{symb}[\undef{\ovar@macro@arrowfill}]{%
```

Fill macro.

```
189    fill macro/.store in=\ovar@macro@arrowfill,
190    fill macro/.value required,
```

Arrow macro.

```
191    no arrow macro hook/.code={%
192      \ifdef{\ovar@macro@arrowfill}{}{%
193        \ovar@set{%
194          fill macro/.expanded={%
195            \noexpand\ovar@arrow@fill%
196            {\expandonce\ovar@shift@left}{\expandonce\ovar@shift@right}%
197          }
198        }
199      }
200      \ovar@set{%
201        arrow macro/.expanded={%
202          \expandonce{\ovar@macro@arrowfill}%
203          {\expandonce{\ovar@arrow@start}\expandonce{\ovar@trim@start}}%
204          {\expandonce{\ovar@trim@middle}\expandonce{\ovar@arrow@middle}%
205            \expandonce{\ovar@trim@middle}}%
206          {\expandonce{\ovar@trim@end}\expandonce{\ovar@arrow@end}}%
207        }
208      }
209    },
```

start$^{\to\text{P. }22}$, middle$^{\to\text{P. }22}$, end$^{\to\text{P. }22}$.

```
210    start/.store in=\ovar@arrow@start,
211    middle/.store in=\ovar@arrow@middle,
212    end/.store in=\ovar@arrow@end,
213    start/.value required,
214    middle/.value required,
215    end/.value required,
```

trim start$^{\to\text{P. }23}$, trim middle$^{\to\text{P. }23}$, trim end$^{\to\text{P. }23}$, trim$^{\to\text{P. }23}$, no trimming$^{\to\text{P. }23}$.

```
216    trim start/.code={\def\ovar@trim@start{\xjoinrel[##1]}},
217    trim middle/.code={\def\ovar@trim@middle{\xjoinrel[##1]}},
218    trim end/.code={\def\ovar@trim@end{\xjoinrel[##1]}},
219    trim start/.value required,
220    trim middle/.value required,
221    trim end/.value required,
222    trim/.code={\pgfkeysalso{trim start={##1}, trim middle={##1}, trim end={##1}}},
223    trim/.value required,
224    no trimming/.code={%
225      \let\ovar@trim@start\empty
226      \let\ovar@trim@middle\empty
227      \let\ovar@trim@end\empty
228    },
229    no trimming/.value forbidden,
```

middle config$^{\to\text{P. }23}$.

```
230    middle config/.is choice,
231    middle config/.value required,
232    middle config/relbar/.code=\pgfkeysalso{%
233      middle={\relbar},
234      trim middle={2.5},
235    },
236    middle config/relbareda/.code={%
237      \ifundef{\relbareda}{%
238        \PackageWarning{overarrows}{Key 'middle config=relbareda' used,
```

```
239        \MessageBreak%
240        but \protect\relbareda\space is undefined; ignored.
241        \MessageBreak%
242        Load 'esvect' package, or use 'esvect' option \MessageBreak%
243        to remove this warning}
244    }{%
245      \pgfkeysalso{%
246        middle={\relbareda},
247        trim middle={1},
248      }
249    }
250  },
251  middle config/auto/.code={%
252    \ifovar@option@esvect@
253    \pgfkeysalso{middle config=relbareda}
254    \else
255    \pgfkeysalso{middle config=relbar}
256    \fi
257  },
```

amsmath$^{\rightarrow P.\,23}$.

```
258    amsmath/.is choice,%
259    amsmath/mimic/.code=\pgfkeysalso{%
260      start={\relbar}, middle={\relbar}, end={\rightarrow},
261      trim start=7,
262      trim middle=2,
263      trim end=7,
264      shift leftright=0,
265      after arrow={}, before arrow={},
266    },
267    amsmath/strict/.code=\pgfkeysalso{%
268      amsmath=mimic,
269      no trimming,
270      fill macro={\arrowfill@}, stack macro={\overarrow@},
271    },
272    amsmath/.default=mimic,
```

esvect$^{\rightarrow P.\,24}$.

```
273    esvect/.is choice,%
274    esvect/mimic/.code=\pgfkeysalso{%
275      start={\relbaredd}, middle={\relbareda}, end={\fldr},
276      trim start=1.5,
277      trim end=1.5,
278      trim middle=0,
279      right arrow=2,
280      space before arrow=-.7pt,
281      space after arrow=-.3pt,
282    },
283    esvect/strict/.code=\pgfkeysalso{%
284      esvect=mimic,
285      no trimming,
286      fill macro={\traitfill@}, stack macro={\overvect@},
287    },
288    esvect/.default=mimic,
```

Initial configuration.

```
289    amsmath, middle config=auto, end=\ovar@rightarrow, right arrow,
290  }
```

Keys for the `tikz` method

```
291  \SetOverArrowsMethod[lens]{tikz}[\undef{\ovar@tikz@command}]{%
```

Arrow macro.

```
292    no arrow macro hook/.code={%
293      \ifdef{\ovar@tikz@command}{}{%
294        \pgfkeysgetvalue{/overarrows/path options}{\ovar@tikz@pathoptions}
295        \ovar@set{%
296          tikz command/.expanded={%
297            \noexpand\draw[\expandonce\ovar@tikz@pathoptions]\expandonce\ovar@tikz@path;
298          }
299        }
300      }
301      \pgfkeysgetvalue{/overarrows/tikz options}{\ovar@tikz@options}
302      \ovar@set{%
303        arrow macro/.expanded={%
304          $\noexpand\mkern \expandonce{\ovar@shift@left} mu\noexpand\relax$%
305          \noexpand\tikz[\expandonce{\ovar@tikz@options}]{\expandonce{\ovar@tikz@command}}%
306          $\noexpand\mkern \expandonce{\ovar@shift@right} mu\noexpand\relax$%
307        }
308      }
309    },
```

TikZ parts: `tikz command`[→P. 25], `tikz options`[→P. 24], `path options`[→P. 25], `path`[→P. 25].

```
310    tikz command/.store in=\ovar@tikz@command,
311    tikz options/.initial={x=\overarrowlength, line width=\overarrowthickness},
312    path options/.initial={arrows={-Classical TikZ Rightarrow}, cap=round},
313    path/.store in=\ovar@tikz@path,
314    path={(0,0)--(1,0)},
315    tikz command/.value required,
316    tikz options/.value required,
317    path options/.value required,
318    path/.value required,
```

TikZ handy keys: `add path options`[→P. 25], `add tikz options`[→P. 25], `arrows`[→P. 25], `line thickness`[→P. 25], `thinner`[→P. 25].

```
319    add path options/.code=\pgfkeysalso{%
320      path options/.append={, ##1}},%
321    add tikz options/.code=\pgfkeysalso{%
322      tikz options/.append={, ##1}},%
323    arrows/.code=\pgfkeysalso{add path options={arrows={##1}}},%
324    line thickness/.code=\pgfkeysalso{add path options={line width=##1}},%
325    thinner/.code=\pgfkeysalso{line thickness={\overarrowsmallerthickness}},%
326    add path options/.value required,%
327    add tikz options/.value required,%
328    arrows/.value required,%
329    line thickness/.value required,%
330    thinner/.value forbidden,%
```

Initial configuration.

```
331    shift right=-2,
332    min length=12,
333  }
```

Keys for the `picture` method

```
334  \SetOverArrowsMethod[lens]{picture}{%
```

Arrow macro.

```
335    no arrow macro hook/.code={%
336      \ovar@set{%
337        arrow macro/.expanded={%
338          $\noexpand\mkern \expandonce{\ovar@shift@left} mu\noexpand\relax$%
```

```
339          \noexpand\begin{picture}\expandonce{\ovar@picture@geometry}%
340            \noexpand\linethickness{\expandonce{\ovar@picture@linethickness}}%
341            \expandonce{\ovar@picture@command}%
342            \noexpand\end{picture}%
343          $\noexpand\mkern \expandonce{\ovar@shift@right} mu\noexpand\relax$%
344        }
345      }
346    },
```

Picture parts: `picture command`[→P.26], `geometry`[→P.26], `line thickness`[→P.26].

```
347    picture command/.store in=\ovar@picture@command,
348    geometry/.store in=\ovar@picture@geometry,
349    line thickness/.store in=\ovar@picture@linethickness,
350    picture command/.value required,
351    geometry/.value required,
352    line thickness/.value required,
```

Picture handy key: `thinner`[→P.26].

```
353    thinner/.code=\pgfkeysalso{line thickness={\overarrowsmallerthickness}},
```

Initial configuration.

```
354    shift right=-2,
355    min length=18,
356    geometry={(\overarrowlength,1ex)(0,-0.5ex)},%
357    line thickness={\overarrowthickness},%
358    picture command={\put(0,0){\vector(1,0){\overarrowlength}}},%
359  }
```

Commands

Macros for symbols assemblage

`\xjoinrel`

```
360  \ifdef{\xjoinrel}{%
361    \PackageWarning{overarrows}{Command \protect\xjoinrel\space already defined.
362      \MessageBreak%
363      Previous definition will be overridden}
364  }{}
```

Use a default value of `3.5 mu`, as recommended by egreg (see `https://tex.stackexchange.com/a/471736`). `\joinrel` uses a value of 3 mu.

```
365  \DeclareRobustCommand{\xjoinrel}[1][3.5]{\mathrel{\mkern-#1mu}}
```

`\smallermathstyle`

```
366  \newcommand*{\smallermathstyle}{%
367    \mathchoice{\scriptstyle}{\scriptstyle}{\scriptscriptstyle}{}
368  }
```

`\ovar@arrow@fill`

Macro used for default `fill macro`[→P.27].

#1: left shift
#2: right shift
#3: arrow start
#4: arrow middle
#5: arrow end
#6: math style

```
369  \def\ovar@arrow@fill#1#2#3#4#5#6{%
370    $\m@th\thickmuskip0mu\medmuskip\thickmuskip\thinmuskip\thickmuskip\relax%
371    \mkern #1 mu\relax#6#3%
372    \cleaders\hbox{$#6#4$}\hfill%
373    #5\mkern #2 mu\relax$%
374  }
```

Macros for fixed length arrows

Lengths declaration.

```
375  \newlength{\overarrowlength}
376  \newlength{\overarrowthickness}
377  \newlength{\overarrowsmallerthickness}
378  \newlength{\ovar@extralength}
379  \newlength{\ovar@tempdim}
```

\ovar@set@arrowlength

Sets `\overarrowlength`$^{\rightarrow\text{P. }17}$.
#1: min length, in math units
#2: math style
#3: content

```
380  \def\ovar@set@arrowlength#1#2#3{%
381    \settowidth{\ovar@tempdim}{$\m@th#2\mskip #1 mu\relax$}%
382    \settowidth{\overarrowlength}{$\m@th#2#3$}%
383    \ifdim \overarrowlength < \ovar@tempdim \overarrowlength=\ovar@tempdim\fi%
384  }
```

\ovar@set@arrowthickness

Sets `\overarrowthickness`$^{\rightarrow\text{P. }17}$ and `\overarrowsmallerthickness`$^{\rightarrow\text{P. }18}$.
#1: arrow length
#2: math style

```
385  \def\ovar@set@arrowthickness#1{% use rule thickness=\fontdimen 8 font family 3
386    \ifx#1\displaystyle%
387      \overarrowthickness =     \fontdimen 8 \textfont 3%
388      \overarrowsmallerthickness = \fontdimen 8 \scriptfont 3%
389    \else\ifx#1\textstyle%
390      \overarrowthickness =     \fontdimen 8 \textfont 3%
391      \overarrowsmallerthickness = \fontdimen 8 \scriptfont 3%
392    \else\ifx#1\scriptstyle%
393      \overarrowthickness =     \fontdimen 8 \scriptfont 3%
394      \overarrowsmallerthickness = \fontdimen 8 \scriptscriptfont 3%
395    \else%
396      \overarrowthickness =     \fontdimen 8 \scriptscriptfont 3%
397      \overarrowsmallerthickness = \overarrowthickness%
398    \fi\fi\fi%
399  }
```

Stack macros

\ovar@stackover@@

\ovar@stackunder@@

Bases of all stack macros.
#1: min length, in math units
#2: vertical mode material before arrow
#3: vertical mode material after arrow
#4: arrow
#5: math style
#6: content

```
400  \def\ovar@stackover@@#1#2#3#4#5#6{\vbox{\ialign{##\crcr%
401      $#5\mskip #1 mu\relax$\crcr%
402      \noalign{#2\nointerlineskip}#4\crcr%
403      \noalign{#3\nointerlineskip}%
404      $\m@th\hfil#5#6\hfil$\crcr%
405    }%
406  }%
407  }
408  \def\ovar@stackunder@@#1#2#3#4#5#6{\vtop{\ialign{##\crcr%
409      $\m@th\hfil#5#6\hfil$\crcr%
```

```
410        \noalign{#2\nointerlineskip}#4\crcr%
411        \noalign{#3\nointerlineskip}%
412        $#5\mskip #1 mu\relax$\crcr%
413     }%
414   }%
415 }
```

`\ovar@stackover@`

`\ovar@stackunder@`

Stack macros without min arrow length.
#1: vertical mode material before arrow
#2: vertical mode material after arrow
#3: arrow macro
#4: math style
#5: content

```
416 \def\ovar@stackover@#1#2#3#4#5{\ovar@stackover@@{0}{#1}{#2}{#3}{#4}{#5}}
417 \def\ovar@stackunder@#1#2#3#4#5{\ovar@stackunder@@{0}{#1}{#2}{#3}{#4}{#5}}
```

`\ovar@stackover@fill`

`\ovar@stackunder@fill`

`\ovar@stack@fill`

Stack macros for extensible arrows.
#1: min length, in math units
#2: vertical mode material before arrow
#3: vertical mode material after arrow
#4: arrow filler macro
#5: math style
#6: content

```
418 \def\ovar@stackover@fill#1#2#3#4#5#6{\ovar@stackover@@{#1}{#2}{#3}{#4#5}{#5}{#6}}
419 \def\ovar@stackunder@fill#1#2#3#4#5#6{\ovar@stackunder@@{#1}{#2}{#3}{#4#5}{#5}{#6}}
```

`\ovar@stack@fill` matches the macro `\ovar@stackover@fill` by default, or `\ovar@stackunder@fill` with arrow under$^{\rightarrow P.\,20}$.

```
420 \def\ovar@stack@fill{\ovar@stackover@fill}
```

`\ovar@stackover@lens`

`\ovar@stackunder@lens`

`\ovar@stack@lens`

Stack macros for fixed-length arrows (these call `\ovar@set@arrowlength` and `\ovar@set@arrowthickness`).
#1: min length, in math units
#2: vertical mode material before arrow
#3: vertical mode material after arrow
#4: arrow content macro
#5: math style
#6: content

```
421 \def\ovar@stackover@lens#1#2#3#4#5#6{%
422   \ovar@set@arrowlength{#1}{#5}{#6}%
423   \ovar@set@arrowthickness{#5}%
424   \ovar@stackover@{#2}{#3}{#4}{#5}{#6}%
425 }
426 \def\ovar@stackunder@lens#1#2#3#4#5#6{%
427   \ovar@set@arrowlength{#1}{#5}{#6}%
428   \ovar@set@arrowthickness{#5}%
429   \ovar@stackunder@{#2}{#3}{#4}{#5}{#6}%
430 }
```

`\ovar@stack@lens` matches the macro `\ovar@stackover@lens` by default, or `\ovar@stackunder@lens` with arrow under$^{\rightarrow P.\,20}$.

```
431 \def\ovar@stack@lens{\ovar@stackover@lens}
```

Macro for commands creation

`\DeclareOverArrowCommand`

```latex
432  \NewDocumentCommand{\DeclareOverArrowCommand}{ O{symb} m m }{%
433    \begingroup
434    \ovar@set@common
435    \ifcsdef{ovar@set@#1}{%
436      \csuse{ovar@set@#1}
437    }{%
438      \PackageError{overarrows}{Unknown method #1}
439      {Try with 'symb', 'tikz' or 'picture'}
440    }
441    \ovar@set{#3 }
442    \ifdef{\ovar@macro@arrow}{}{%
443      \ovar@set{no arrow macro hook}
444    }
445    \ifdef{\ovar@macro@stack}{}{%
446      \ovar@set{no stack macro hook}
447    }
448    \csxdef{ovar@#2@normal}{%
449      \noexpand\mathpalette{%
450        \expandonce{\ovar@macro@stack}{\expandonce{\ovar@macro@arrow}}%
451      }
452    }
453    \csxdef{ovar@#2@starred}{%
454      \noexpand\mathpalette{%
455        \noexpand\ovar@starversion{%
456          \expandonce{\ovar@macro@stack}{\expandonce{\ovar@macro@arrow}}%
457        }
458      }
459    }
460    \ifovar@detectsubscripts@%
461    \csgdef{ovar@#2@auto}##1{%
462      \@ifnextchar _{%
463        \csuse{ovar@#2@starred}{##1}%
464      }{%
465        \csuse{ovar@#2@normal}{##1}%
466      }%
467    }
468    \csgdef{#2}{%
469      \@ifstar{\csuse{ovar@#2@starred}}{\csuse{ovar@#2@auto}}%
470    }
471    \else
472    \csgdef{#2}{%
473      \@ifstar{\csuse{ovar@#2@starred}}{\csuse{ovar@#2@normal}}%
474    }
475    \fi
476    \ifovar@option@debug@
477    \PackageInfo{overarrows}{%
478      Meaning of \protect\ovar@#2@normal\MessageBreak
479      used for \@backslashchar#2:\MessageBreak%
480        \expandafter\meaning\csname ovar@#2@normal\endcsname}
481    \fi
482    \endgroup
483  }
```

\ProvideOverArrowCommand
```latex
484  \NewDocumentCommand{\ProvideOverArrowCommand}{ O{symb} m m }{%
485    \ifcsdef{#2}{}{
486      \DeclareOverArrowCommand[#1]{#2}{#3}
487    }
488  }
```

\NewOverArrowCommand
```latex
489  \NewDocumentCommand{\NewOverArrowCommand}{ O{symb} m m }{%
490    \ifcsdef{#2}{%
491      \PackageError{overarrows}{Command \csname #2\endcsname already defined}%
492      {You have used \protect\NewOverArrowCommand\space with a command that
493        already has a definition. \MessageBreak%
```

```
494        Choose another name, or use instead \protect\DeclareOverArrowCommand.}
495    }{%
496      \DeclareOverArrowCommand[#1]{#2}{#3}
497    }
498 }
```

```
499 \NewDocumentCommand{\RenewOverArrowCommand}{ O{symb} m m }{%
500    \ifcsundef{#2}{%
501      \PackageError{overarrows}{Command \csname #2\endcsname undefined}%
502      {You have used \protect\RenewOverArrowCommand\space with a command that was
503        never defined. \MessageBreak%
504        Check the requested name, or use instead \protect\NewOverArrowCommand.}
505    }{%
506      \DeclareOverArrowCommand[#1]{#2}{#3}
507    }
508 }
```

Starred variant

#1: definition (stack macro + arrow macro)
#2: math style
#3: content

```
509 \def\ovar@starversion#1#2#3{%
510    #1#2{#3}%
511    \settowidth{\ovar@extralength}{$\m@th#1#2{#3}$}
512    \settowidth{\ovar@tempdim}{$\m@th#2{#3}$}
513    \deflength{\ovar@extralength}{0.5\ovar@extralength-0.5\ovar@tempdim}%
514    \kern-\ovar@extralength%
515 }
```

\vv vector command

Backup and redefinition of esvect $\vv^{\to \mathrm{P.}\,18}$ vector command.

```
516 \ifovar@option@esvect@
517    \let\esvectvv\vv
518    \undef\vv
519    \NewOverArrowCommand{vv}{esvect, middle config=auto}
520 \fi
```

Predefined commands

```
521 \ifovar@option@overrightarrow@
522    \DeclareOverArrowCommand{overrightarrow}{%
523      amsmath, middle config=relbar,
524      end=\ovar@rightarrow,
525      right arrow,
526    }
527 \fi
```
```
528 \ifovar@option@underrightarrow@
529    \DeclareOverArrowCommand{underrightarrow}{%
530      amsmath, middle config=relbar,
531      end=\ovar@rightarrow,
532      right arrow,
533      arrow under,
534    }
535 \fi
```

```
536  \ifovar@option@overleftarrow@
537    \DeclareOverArrowCommand{overleftarrow}{%
538      amsmath, middle config=relbar,
539      start=\ovar@leftarrow,
540      end=\relbar,
541      left arrow,
542    }
543  \fi
```

\underleftarrow
```
544  \ifovar@option@underleftarrow@
545    \DeclareOverArrowCommand{underleftarrow}{%
546      amsmath, middle config=relbar,
547      start=\ovar@leftarrow,
548      end=\relbar,
549      left arrow,
550      arrow under,
551    }
552  \fi
```

\overleftrightarrow
```
553  \ifovar@option@overleftrightarrow@
554    \DeclareOverArrowCommand{overleftrightarrow}{%
555      amsmath, middle config=relbar,
556      start=\ovar@leftarrow,
557      end=\ovar@rightarrow,
558      center arrow,
559    }
560  \fi
```

\underleftrightarrow
```
561  \ifovar@option@underleftrightarrow@
562    \DeclareOverArrowCommand{underleftrightarrow}{%
563      amsmath, middle config=relbar,
564      start=\ovar@leftarrow,
565      end=\ovar@rightarrow,
566      center arrow,
567      arrow under,
568    }
569  \fi
```

\overrightharpoonup
```
570  \ifovar@option@overrightharpoonup@
571    \DeclareOverArrowCommand{overrightharpoonup}{%
572      amsmath, middle config=relbar,
573      end=\rightharpoonup,
574      right arrow,
575    }
576  \fi
```

\underrightharpoonup
```
577  \ifovar@option@underrightharpoonup@
578    \DeclareOverArrowCommand{underrightharpoonup}{%
579      amsmath, middle config=relbar,
580      end=\rightharpoonup,
581      right arrow,
582      arrow under,
583    }
584  \fi
```

\overrightharpoondown
```
585  \ifovar@option@overrightharpoondown@
586    \DeclareOverArrowCommand{overrightharpoondown}{%
587      amsmath, middle config=relbar,
588      end=\rightharpoondown,
589      right arrow,
590    }
591  \fi
```

\underrightharpoondown
```
592  \ifovar@option@underrightharpoondown@
593    \DeclareOverArrowCommand{underrightharpoondown}{%
```

```
594      amsmath, middle config=relbar,
595      end=\rightharpoondown,
596      right arrow,
597      arrow under,
598    }
599  \fi
```

`\overleftharpoonup`
```
600  \ifovar@option@overleftharpoonup@
601    \DeclareOverArrowCommand{overleftharpoonup}{%
602      amsmath, middle config=relbar,
603      start=\leftharpoonup,
604      end=\relbar,
605      left arrow,
606    }
607  \fi
```

`\underleftharpoonup`
```
608  \ifovar@option@underleftharpoonup@
609    \DeclareOverArrowCommand{underleftharpoonup}{%
610      amsmath, middle config=relbar,
611      start=\leftharpoonup,
612      end=\relbar,
613      left arrow,
614      arrow under,
615    }
616  \fi
```

`\overleftharpoondown`
```
617  \ifovar@option@overleftharpoondown@
618    \DeclareOverArrowCommand{overleftharpoondown}{%
619      amsmath, middle config=relbar,
620      start=\leftharpoondown,
621      end=\relbar,
622      left arrow,
623    }
624  \fi
```

`\underleftharpoondown`
```
625  \ifovar@option@underleftharpoondown@
626    \DeclareOverArrowCommand{underleftharpoondown}{%
627      amsmath, middle config=relbar,
628      start=\leftharpoondown,
629      end=\relbar,
630      left arrow,
631      arrow under,
632    }
633  \fi
```

`\overbar`
```
634  \ifovar@option@overbar@
635    \DeclareOverArrowCommand{overbar}{%
636      amsmath, middle config=relbar,
637      start={\std@minus}, end={\std@minus},% \relbar is defined with \mathsm@sh
638      shift leftright=0,
639      space after arrow=-0.3ex,
640    }
641  \fi
```

`\underbar`
```
642  \ifovar@option@underbar@
643    \DeclareOverArrowCommand{underbar}{%
644      amsmath, middle config=relbar,
645      start={\std@minus}, end={\std@minus},% \relbar is defined with \mathsm@sh
646      shift leftright=0,
647      arrow under,
648      space before arrow=-0.3ex,
649    }
650  \fi
```

Test macros

Tabular containing the output of a command for the four math styles and different patterns.

```
651  \newcommand{\ovar@testmathstyles}[2][]{
652    \begingroup
653    \newcommand*{\ovar@row@teststyle}[1]{%
654      $\displaystyle ##1$
655      & $\textstyle ##1$
656      & $\scriptstyle ##1$
657      & $\scriptscriptstyle ##1$
658      \\
659    }
660    \renewcommand*{\arraystretch}{1.5}
661    \begin{tabular*}{0.95\linewidth}{@{\extracolsep{\fill}} cccc}
662      \hline
663      \footnotesize\texttt{\texttt{\textbackslash displaystyle}}
664      & \footnotesize\texttt{\texttt{\textbackslash textstyle}}
665      & \footnotesize\texttt{\texttt{\textbackslash scriptstyle}}
666      & \footnotesize\texttt{\texttt{\textbackslash scriptscriptstyle}}
667      \\
668      \hline
669      \ovar@row@teststyle{\csuse{#2}{v}}
670      \ovar@row@teststyle{\csuse{#2}{AB}}
671      \ovar@row@teststyle{\csuse{#2}{\mathrm{grad}}}
672      \ovar@row@teststyle{\csuse{#2}{my~long~vector}}
673      \IfValueT{#1}{\ovar@row@teststyle{\csuse{#2}{#1}}}
674      \hline
675    \end{tabular*}
676    \endgroup
677  }
```

```
678  \NewDocumentCommand{\TestOverArrow}{ s o m }{
679    \ifcsdef{#3}{}{%
680      \PackageWarning{overarrows}{Unknown name '#3' passed to
681        \protect\TestOverArrow}
682    }
683    \IfBooleanTF{#1}{%
684      \noindent\framebox{%
685        \begin{minipage}{0.95\linewidth}
686          \centering
687          \noindent\textbf{\large%
688            Test of \texttt{\textbackslash#3} and \texttt{\textbackslash#3*} macros}
689          \bigskip\par
690          \textbf{\texttt{\textbackslash#3} for different math styles}
691          \smallskip\par
692          \ovar@testmathstyles[#2]{#3}%
693          \bigskip\par
694          \textbf{\texttt{\textbackslash#3} kerning}
695          \begin{displaymath}
696            \csuse{#3}{t}_{\csuse{#3}{u}_{\csuse{#3}{v}}}
697            \qquad
698            \csuse{#3}{\imath}_0
699            \qquad
700            \csuse{#3}{v}
701            = \csuse{#3}{v}_x + \csuse{#3}{v}_y + \csuse{#3}{v}_z
702            = v_x \csuse{#3}{\imath} + v_y \csuse{#3}{\jmath} + v_z \csuse{#3}{k}
703          \end{displaymath}
704          \textbf{\texttt{\textbackslash#3*} kerning}
705          \begin{displaymath}
706            \csuse{#3}*{t}_{\csuse{#3}*{u}_{\csuse{#3}*{v}}}
707            \qquad
```

```
708          \csuse{#3}*{\imath}_0
709          \qquad
710          \csuse{#3}*{v}
711          = \csuse{#3}*{v}_x + \csuse{#3}*{v}_y + \csuse{#3}*{v}_z
712          = v_x \csuse{#3}*{\imath} + v_y \csuse{#3}*{\jmath} + v_z \csuse{#3}*{k}
713        \end{displaymath}
714      \end{minipage}%
715    }\bigskip\par
716  }{%
717    \ovar@testmathstyles[#2]{#3}%
718  }
719 }
```

Index

Entries listed in the categories "commands", "lengths", and "internal macros" also include references to package implementation.