

mismath

Miscellaneous mathematical macros*

Antoine Missier
`antoine.missier@ac-toulouse.fr`

March 1, 2023

Contents

1	Introduction	1
2	Usage	2
2.1	Mathematical constants	2
2.2	Vectors (and tensors)	4
2.3	Standard operator names	5
2.4	A few useful aliases	7
2.5	Improved spacing in mathematical formulas	8
2.6	Environments for systems of equations and small matrices	10
2.7	Displaymath in double columns	11
2.8	Deprecated commands	11
3	Implementation	12

1 Introduction

According to the International Standards ISO 31-0:1992 to ISO 31-13:1992, superseded by ISO 80000-2:2009, mathematical constants e , i , π should be typeset in roman (upright shape) and not in italic (sloping shape) like variables (see [1] [2] [3] [4]). This package provides some tools to achieve this (automatically).

Even if it is recommended to typeset vectors names in bold italic style [2] [4], they are often represented with arrows (particularly in school documents or in physics). To draw pretty arrows above vectors, we use the `esvect` package by Eddie Soudrais [5] and we provide a few more macros related to vectors with arrows, in particular to

*This document corresponds to mismath v2.6, dated 2023/03/01. Thanks to François Bastouil for help in English translation.

improve the typesetting of the norm: $\|\overrightarrow{AB}\|$ instead of \mathbb{X} version $\|\overline{AB}\|$ which is not vertically adjusted, or worse $\|\overline{AB}\|$ (and even ugly with Latin Modern font family).

The package also provides other macros for:

- tensors,
- some standard operator names,
- a few useful aliases,
- improving some spacing in mathematical formulas,
- systems of equations and small matrices,
- displaymath in double columns for long calculation.

To avoid incompatibility, most of our macros will be defined only if there is not another command with the same name in the packages loaded before mismath. If a macro is already defined, a warning message will be produced and the mismath definition will simply be ignored. To keep the mismath command or the other one, use `\let\<command>\relax`, before loading mismath, or after. If the other one is defined with `\AtBeginDocument`, do the same for `\let\<command>\relax` and for loading mismath.

`[(options)]` The amsmath package is loaded by mismath without option. For using amsmath with options (see [6]), these options can be added when calling mismath, or amsmath can be loaded with the required options before mismath.

mismath loads also the package mathtools by Morten Høgholm and Lars Madsen [7]. It provides many useful macros and improvements of amsmath package.

A recommendation, seldom observed, is to typeset uppercase Greek letters in italic shape like other variables [4]. This is automatically done with the packages fixmath by Walter Schmidt [8], isomath by Günter Milde [9] or pm-isomath by Claudio Becari [10] and optionally with many others (for instance mathpazo or mathptmx with the option `slantedGreek`), but this feature is not implemented here because this rule is conflicting to the one used in France where all mathematics capitals have to be typeset in upright shape¹. The choice of loading or not one of these packages remains thus to the user.

2 Usage

2.1 Mathematical constants

`\mathup` As for classic functions identifiers, *predefined* mathematical constants should be typeset in upright shape (generally in roman family), even if this practice is not really common and tedious to respect. First we provide the `\mathup` macro, which is better

¹The frenchmath package [21] takes this rule into account.

than `\mathrm`², to set any math text in upright shape, so one can write `\mathup{e}` to get the Euler’s number.

`\e` To avoid to stuff a document that contains many *e* or *i* constants with `\mathup{e}`
`\i` or `\mathup{i}`, the package provides `\e` command for Euler’s number and `\i` or `\j`
`\j` for imaginary numbers. Let’s notice that `\i` and `\j` already exist in \TeX : using in LR mode, they produce ‘i, j’ without the point, so you can place accents on them, and in mathematical mode they produce “LaTeX Warning: Command `\i` invalid in math mode on input line *<line>*”. The new definition of `\i` and `\j` concerns only the mathematical mode.

`\MathUp` Nevertheless, it can be tiresome to type a lot of backslashes for these constants, in a document with many formulas containing *e*, *i* or *j*. So a way is proposed here to free of it with the macro `\MathUp{<char>}`. For instance when `\MathUp{e}` is called, any future occurrence of *e* will then automatically be set in roman, without the need to type `\e`. The effect is global or local if used inside an environment or braces. This macro can also be called in the preamble for applying from the beginning of the document. Thanks to this powerful macro, you can bring a document up to the standards afterwards. In fact `\MathUp` can apply to any valid but single character (we will see another use of it with probability in section 2.3).

`\MathIt` When there are other *e*, *i* or *j* as variables, you can still get italicized *e*, *i* or *j* with \TeX commands `\mathit` or `\mathnormal`, useful for a single use. But you can also use the inverse switch `\MathIt{<char>}`, with a global effect, or a local one if used inside an environment or braces. As `\MathUp`, it can be used for any single character.

`\MathNumbers` These macros allow to set upright or normal typesetting for several letters in a single command, e.g. `\MathNumbers{e,i}` is equivalent to `\MathUp{e}\MathUp{i}`.
`\MathNormal` In `\MathNumbers` the comma separator can be changed or deleted. This macro has no effect on other letters than *e*, *i* or *j*. On the other hand `\MathNormal` can be used for probability also (see section 2.3) and can take any comma separated list argument.

`\pinumber[<command>]` The mathematical constant π should also be typeset in upright shape (see [1], [2], [4]), which differs from italicized π . This recommendation is even less observed than the one concerning *e* and *i* [1]. Several packages allow to typeset mathematical Greek letters in upright shape, let us mention `upgreek` [11], `mathdesign` [12] (used here), `kpfonts` [14], `fourier` [15], `libertinustlmath`, `pxgreek`, `txgreek`, `libgreek`, etc. A special mention for `lgrmath` of Jean-François Burnol [16] which allow to use, in math mode, any Greek LGR-encoded font. These packages provide commands like `\uppi` (`upgreek`), `\piup` (`mathdesign`, `kpfonts`, `lgrmath`), `\otherpi` (`fourier`), etc.³ To preserve default sloped lowercase Greek letters except for π , and to avoid typing a lot of `\uppi` or `\piup`, we provide the macro `\pinumber[<command>]`. It redefines `\pi` to match the optional command name given (without backslash), for instance `\piup`, assuming the appropriate package has been loaded before.

²`\mathup` is based on `\operatorfont` (from `amsopn` package, automatically loaded by `amsmath`). The `beamer` package uses a default sans serif math font, but `\mathrm` produces a font with serif in `beamer`. Therefore using `\mathup` is better than `\mathrm`.

³They also have options to typeset all the Greek lowercase letters in upright shape by default, but this is not our goal here.

By calling preliminary `\MathNumbers{ei}\pinumber[piup]` (and with the `mathdesign` package loaded) you can get for instance:

$$e^{i\pi} = -1 \quad \text{yields} \quad e^{i\pi} = -1.$$

When calling `\pinumber` without argument it defines `\pi` with the default LGR font encoding of Greek letters to produce π . In that case the appropriate option LGR for the `fontenc` package will be automatically loaded, provided that the command is called in the preamble (first). The π character will look the same as the one supplied with Günter Milde's `textalpha` package [13]. This π is particularly suitable for use with the default Computer Modern or Latin Modern font family⁴.

`\itpi` When activating `\pinumber`, the original italic π is still available with `\itpi`.
`\pinormal` In fact `\pinumber` acts as a switch and there is also an inverse switch, `\pinormal`, that can be called anywhere.

2.2 Vectors (and tensors)

`\vect` By default, the `\vect` command⁵, produces vectors with arrows (thanks to the `esvect` package of Eddie Sautrais⁶) which are more elegant than those produced by \TeX 's `\overrightarrow` command. The `esvect` package has an optional argument (one letter between a and h) defining the required type of arrow (see [5]). In `mismath`, `esvect` is loaded with the option `b`: `\vect{AB}` gives \overrightarrow{AB} . To choose another type of arrow, `esvect` must be called with the required option *before* `mismath`, e.g. `\usepackage[d]{esvect}` will give the arrows produced by default in [5].

`\boldvect` The `\vect` macro allow to typeset vector's names using bold italic (according to ISO recommendation [2] [3]) rather than arrows. For this, calling `\boldvect` will modify the behavior of `\vect`, globally or locally, depending on where `\boldvect` is placed:

$$\begin{aligned} & \text{\texttt{[\boldvect \vect{v} \\}} \\ & \text{\texttt{= \lambda \vect{e}_x + \mu \vect{e}_y. \texttt{]}}} \end{aligned} \quad \mathbf{v} = \lambda \mathbf{e}_x + \mu \mathbf{e}_y.$$

`\boldvectcommand` By default `\boldvect` uses the `\boldsymbol` command⁷ from `amssymb` package, loaded by `amsmath`. But other packages producing bold italic can be preferred, e.g. `\mathbf` from `fixmath` package or `\mathbf` from `isomath` or `\bm` from `bm` package. For that, redefine `\boldvectcommand`, for instance, after loading `fixmath`:

`\renewcommand\boldvectcommand{\mathbf}`.

According to ISO rules, symbols for matrices are also in bold italic, so you can use the same `\boldvectcommand` or create another alias.

`\arrowvect` At any moment, you can get back to the default behavior with the inverse switch

⁴This default π doesn't fit well with many text fonts, more bold than Computer Modern; the `upgreek` package [11] provides often a better π (with the `Symbol` option using Adobe Symbol font) that fits well with several text fonts, for instance Times.

⁵As for many macros of this package, the definition will take effect only if this macro is not defined before by another package.

⁶`esvect` provides the `\vv` macro used by `\vect`.

⁷`\mathbf` gives upright bold font, even if used in combination with `\mathit`.

`\arrowvect`. These switches can be placed anywhere: inside mathematical mode or inside an environment (with local effect) or outside (with global effect).

`\hvect` When vectors with arrows are typeset side by side, arrows can be set up a bit higher (with a vertical phantom box containing t) to avoid inelegant effects:

- $\overrightarrow{AB} = \vec{u} + \overrightarrow{AC}$, obtained with `\hvect{u}`, is better than $\overrightarrow{AB} = \vec{u} + \overrightarrow{AC}$;
- $\vec{a} \cdot \vec{b} = 0$, obtained with `\hvect{a}`, is better than $\vec{a} \cdot \vec{b} = 0$.

The `\boldvect` and `\arrowvect` switches have the same effect on `\hvect` than on `\vect`, and so have `boldvect` and `arrowvect` options.

`\hvec` In a similar way, `\hvec` raises the little arrow produced by the \TeX command `\vec` (from height of t letter):

- $\vec{\mathcal{P}} = \vec{f} \cdot \vec{v}$, obtained with `\hvec{v}`, is better than $\vec{\mathcal{P}} = \vec{f} \cdot \vec{v}$.
- $\vec{f} = m\vec{a}$, obtained with `\hvec{a}`, is better than $\vec{f} = m\vec{a}$.

`\norm` The norm of a vector is classically produced by the delimiters `\lVert` and `\rVert` (rather than `\|`) or `\left\lVert` and `\right\lVert` for delimiters adapting to the content. Unfortunately, these delimiters are always vertically centered, relatively to the middle of the base line, whereas vectors with arrows are asymmetric objects. The code `\norm{\vec{h}}` raises a smaller double bar to produce $\|\vec{h}\|$ instead of $\|\vec{h}\|$. Let's notice that the height of the bars don't adjust to content, but however to context: main text, subscripts or exponents, e.g. $e^{\|\vec{h}\|}$.

`\mathbfsfit` For tensors symbols, ISO rules recommend to use sans serif bold italic, but there is no such math alphabet in \TeX default mathematical style. `mismath` defines this alphabet (assuming the font encoding and package you use permits it), and provides the macro `\mathbfsfit` or its alias `\tensor`. So `\tensor{T}` produces $\mathbf{\textit{T}}$.

2.3 Standard operator names

`\di` The *differential* operator should be typeset in upright shape and not in italic, to make it different from variables (as mentioned in [1] [2] [4] [23]). For this, we provide the `\di` command. See the following examples (notice the thin spaces before the d , as for classic function's names):

$$\begin{aligned} & \int \int xy \, dx \, dy \\ & m \frac{d^2 x}{dt^2} + h \frac{dx}{dt} + kx = 0 \end{aligned}$$

This command can also stand for *distance* (hence its name):

$$\lambda d(A, \mathcal{F}) + \mu d(B, \mathcal{H}).$$

`\P` To refer to probability⁸ and expectation the proper use is to typeset capital letters
`\E` P, E in roman as for any standard function identifier. This is obtained with `\P` and `\E`.

<code>\Par</code>	The <code>\P</code> command already existed to refer to the end of paragraph symbol ¶ and has been redefined, but this symbol can still be obtained with <code>\Par</code> .
<code>\V</code>	Variance is generally denoted by <code>var</code> or <code>Var</code> (see table below), but some authors prefer to use <code>V</code> , produced by <code>\V</code> .
<code>\MathProba</code> <code>\MathNormal</code>	In the same way as for <code>e</code> , <code>i</code> or <code>j</code> , you can use <code>\MathUp{P}</code> , <code>\MathUp{E}</code> or <code>\MathUp{V}</code> to avoid typing many <code>\P</code> , <code>\E</code> or <code>\V</code> . But you can also do that in a single command with <code>\MathProba</code> , for example <code>\MathProba{P,E}</code> and we get the inverse switch with <code>\MathIt</code> for any individual letter or <code>\MathNormal</code> for a list.
<code>\probastyle</code>	Some authors use “blackboard bold” font to represent probability, expectation and variance: $\mathbb{P}, \mathbb{E}, \mathbb{V}$. The <code>\probastyle</code> macro sets the appearance of <code>\P</code> , <code>\E</code> and <code>\V</code> : for instance <code>\renewcommand\probastyle{\mathbb}</code> ⁹ brings the previous “open-work” letters. <code>\mathbb</code> comes from <code>amsmath</code> package (loaded by <code>amssymb</code> but also available standalone) which has to be called in the preamble.

The following standard operator names are defined in `mismath`:

<code>\adj</code>	<code>adj</code>	<code>\erf</code>	<code>erf</code>	<code>\Re</code>	<code>Re</code>
<code>\Aut</code>	<code>Aut</code>	<code>\grad</code>	<code>$\overrightarrow{\text{grad}}$</code>	<code>\rot</code>	<code>$\overrightarrow{\text{rot}}$</code>
<code>\codim</code>	<code>codim</code>	<code>\id</code>	<code>id</code>	<code>\sgn</code>	<code>sgn</code>
<code>\Conv</code>	<code>Conv</code>	<code>\Id</code>	<code>Id</code>	<code>\sinc</code>	<code>sinc</code>
<code>\cov</code>	<code>cov</code>	<code>\im</code>	<code>im</code>	<code>\spa</code>	<code>span</code>
<code>\Cov</code>	<code>Cov</code>	<code>\Im</code>	<code>Im</code>	<code>\tr</code>	<code>tr</code>
<code>\curl</code>	<code>$\overrightarrow{\text{curl}}$</code>	<code>\lb</code>	<code>lb</code>	<code>\var</code>	<code>var</code>
<code>\divg</code>	<code>div</code>	<code>\lcm</code>	<code>lcm</code>	<code>\Var</code>	<code>Var</code>
<code>\End</code>	<code>End</code>	<code>\rank</code>	<code>rank</code>	<code>\Zu</code>	<code>Z</code>

By default, operators returning vectors, `\grad` and `\curl` (or its synonym `\rot` rather used in Europe), are written with an arrow on the top. When `\boldvect` is activated, they are typeset in bold style: **grad**, **curl**, **rot**. For the variance, the covariance and the identity function, two notations are proposed, with or without a first capital letter, because they are both very common. On the other hand, ‘im’ stands for the image of a linear transformation (like ‘ker’ for the kernel) whereas ‘Im’ is the imaginary part of a complex number. Notice that `\div` already exist (÷) and `\span` is a \TeX primitive (used in `\multicolumn`); they haven’t been redefined, therefore the macros `\divg` (divergence) and `\spa` (span of a set of vectors) ; `\Z` is used for the set of integers (see 2.4), therefore we used `\Zu`, to designate the center of a group: $Z(G)$ (from German Zentrum).

<code>\oldRe</code> <code>\oldIm</code>	The <code>\Re</code> and <code>\Im</code> macros already existed, to refer to real and imaginary part of a complex number, producing outdated symbols \Re and \Im . They have been redefined according to actual use, as mentioned in the above table, but it’s still possible to get the old symbols with <code>\oldRe</code> and <code>\oldIm</code> .
--	--

Some (inverse) circular or hyperbolic functions, missing in \TeX , are also provided by `mismath`:

⁸ \TeX provides also `\Pr` which gives \Pr .

⁹As for `\boldvect` and `\arrowvect`, effect is local to the container environment.

<code>\arccot</code>	<code>arccot</code>	<code>\arsinh</code>	<code>arsinh</code>	<code>\arcoth</code>	<code>arcoth</code>
<code>\sech</code>	<code>sech</code>	<code>\arcosh</code>	<code>arcosh</code>	<code>\arsech</code>	<code>arsech</code>
<code>\csch</code>	<code>csch</code>	<code>\artanh</code>	<code>artanh</code>	<code>\arcsch</code>	<code>arcsch</code>

`\bigO` Asymptotic comparison operators (in Landau notation) are obtained with `\bigO`
`\bigo` or `\bigo` and `\lito` commands:
`\lito`

$$n^2 + \mathcal{O}(n \log n) \quad \text{or} \quad n^2 + O(n \log n) \quad \text{and} \quad e^x = 1 + x + o(x^2).$$

2.4 A few useful aliases

In the tradition of Bourbaki and D. Knuth, proper use requires that classic sets of numbers are typeset in bold roman: **R, C, Z, N, Q**, whereas “openwork” letters ($\mathbb{R}, \mathbb{Z}, \dots$) are reserved for writing at blackboard [23]; and likewise to designate a field: **F** or **K** (Körper in German). We get these symbols with the macros¹⁰:

`\R`, `\C`, `\Z`, `\N`, `\Q`, `\F`, `\K`.

`\mathset` The `\mathset` command enables to change the behavior of all these macros in a global way: by default, `\mathset` is an alias for `\mathbf`, but if you prefer openwork letters, just place `\renewcommand\mathset{\mathbb}` where you want, for instance in the preamble, after loading `amsmath` package (which provides the “blackboard bold” typeface, also loaded by `amssymb`).

`\ds` The `\displaystyle` command being very common, alias `\ds` is provided. Not only it eases typing but also it makes source code more readable.

Symbols with limits behave differently for in-line formulas or for displayed equations. In the latter case, “limits” are put under or above whereas for in-line math mode, they are placed on the right, as subscript or exponent. Compare: $\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$ with

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}.$$

`\dlim` With in-line math mode, `\displaymath` behavior can be forced with `\displaystyle`
`\dsum` or its alias `\ds`, but then, all the rest of the current mathematical environment will be
`\dprod` set in `\displaymath` mode too (in the previous example, the fraction will be expanded).
`\dcup` Just as the `\dfrac` command `\dfrac` only transforms the required fraction in display
`\dcap` style, we can limit the display style effect to the affected symbol, by using the
following macros: `\dlim`, `\dsum`, `\dprod`, `\dcup`, `\dcap`. So

$$\text{\$}\text{\dlim_{x\to +\infty}}\text{\frac{1}{x}}\text{\$} \quad \text{gives} \quad \lim_{x \rightarrow +\infty} \frac{1}{x}.$$

`\lbar` Large bars over expressions are obtained with `\overline` or, shorter, its alias
`\hlbar` `\lbar`, to get for instance $\overline{z_1 z_2}$. Such as for vectors, you can raise the bar (from the
height of *h*) with the `\hlbar` command, in order to correct uneven bars heights.

$$\overline{z + z'} = \overline{z} + \overline{z'}, \text{ obtained with } \text{\hlbar{z}}, \text{ is better than } \overline{z} + \overline{z'} = \overline{z} + \overline{z'}.$$

¹⁰The `\C` macro is incompatible with `rustian` option of `babel`; then it will not be defined by `mismath`.

`\eqdef` The `\eqdef` macro writes equality symbol topped with ‘def’ or with ‘Δ’ for
`\eqdef*` `\eqdef*` (thanks to the `\stackrel` command `\stackrel`):

`$ \e~{\i\theta} \eqdef`
`\cos\theta + \i\sin\theta $` $e^{i\theta} \stackrel{\text{def}}{=} \cos \theta + i \sin \theta$

`$ \e~{\i\theta} \eqdef*`
`\cos\theta + \i\sin\theta $` $e^{i\theta} \stackrel{\Delta}{=} \cos \theta + i \sin \theta$

`\unbr` `\unbr` is an alias for `\underbrace`¹¹, making source code more compact.

`$ (QAP)^n = \unbr{QAP\mul QAP\mul`
`\cdots\mul QAP}_n\text{ times} $` $(QAP)^n = \underbrace{QAP \times QAP \times \cdots \times QAP}_{n \text{ times}}$

`\iif` `\iif` is an alias for “if and only if”, to be used in text mode.

2.5 Improved spacing in mathematical formulas

`\then` The `\then` macro produces the symbol \implies surrounded by large spaces as the stan-
`\txt` dard macro `\iff` does it with \iff . In a similar way, `\txt`, based on the `\text` macro from the `amstext` package (loaded by `amsmath`), leaves `em quad` spaces (`\quad`) around the text. See the following example:

`\[\ln x=a \then x=e^a \txt{rather than}`
`\ln x=a \Longrightarrow x=e^a \]`

$\ln x = a \implies x = e^a$ rather than $\ln x = a \implies x = e^a$

`\mul` The multiplication symbol obtained with `\times` produces the same spacing than addition or subtraction operators, whereas division obtained with `/` is closer to its operands. This actually hides the priority of the multiplication on `+` and `-`. This is why we provide the `\mul` macro, behaving like `/` (ordinary symbol) and leaving less space around than `\times`:

$\lambda + \alpha \times b - \beta \times c$, obtained with `\mul`, is better than $\lambda + \alpha \times b - \beta \times c$.

When using `\mul` before a function name or around a `\left...\right` structure, the space may be too large on one side of `\mul`. To get the same amount of space on the two sides of `\mul`, you can use thin negative spaces `\!` or enclose the function or the structure with braces:

$x \times \sin x$, obtained with `x\mul{\sin x}`, is slightly better than $x \times \sin x$.

`$\sin\!\{\left(\frac{\pi}{3} \right)\} \mul 2$` gives
 $\sin\left(\frac{\pi}{3}\right) \times 2$ which is better than $\sin\left(\frac{\pi}{3}\right) \times 2$.

The thin negative space after the function name is not relative to `\mul`, but is due to the fact that spaces around a `\left...\right` structure are bigger than those produced by single parenthesis `(...)`.

`\pow` In the same way, when typesetting an exponent after a closing *big* parenthesis pro-

¹¹The `mathtools` package by Morten Høgholm and Lars Madsen [7] provides a new improved version of `\underbrace` command (as many other usefull macros); it is loaded by `mismath`.

duced by `\right)`, the exponent is little to far from the parenthesis. The command `\pow{⟨expr⟩}{⟨pow⟩}` sets $\langle expr \rangle$ between parentheses and puts the exponent $\langle pow \rangle$ slightly closer to the right parenthesis¹². Compare:

$$e^a \sim \left(1 + \frac{a}{n}\right)^n \quad \text{may be better than} \quad e^a \sim \left(1 + \frac{a}{n}\right)^n.$$

`\abs` Absolute value (or modular for a complex number) should be typeset with `\lvert` ... `\rvert` rather than `|` which doesn't respect correct spaces for delimiters; for bars whose height has to adapt to content, we use `\left\lvert` ... `\right\rvert` or, more simply, the `\abs{...}` command which is equivalent¹³.

`\lfrac` This macro behaves like `\frac` but with thick spaces around the arguments, so the corresponding fraction bar is perceptibly a little bit longer:

$$\left[\frac{\lvert \bar{z} \rvert}{\lvert \bar{z}_1 - z_2 \rvert} \right] \quad \bar{z} = \frac{z_1 - z_2}{z_1 + z_2}$$

`[ibackets]` Open intervals are usually represented with parenthesis, e.g. $(0, +\infty)$, but sometimes we find also square brackets, for example in French mathematics. In that case the space around them is often unsuitable, e.g. $x \in]0, +\infty[$. We have redefined brackets in the `ibackets` package [22] which can be optionally¹⁴ loaded by `mismath` with `ibackets` package option¹⁵.

Simply type `$x \in]-\pi, 0[\cup]2\pi, 3\pi[` to get

$$x \in]-\pi, 0[\cup]2\pi, 3\pi[\quad \text{with } \texttt{ibackets},$$

instead of $x \in]-\pi, 0[\cup]2\pi, 3\pi[$ without `ibackets`.

In our code, `[` and `]` symbols become “active” and are not defined by default as delimiters. Thereby a line break could occur between the two brackets, but it is always possible to transform them into delimiters with `\left` and `\right`.

With `ibackets`: a bracket becomes an ordinary character but an open delimiter when it is immediately followed by a `+` or `-` character. Thus, when the left bound contains an operator sign, *you don't have to leave a space between the first bracket and the sign*, otherwise, the spaces surrounding the operator will be too large: e.g. `$x \in]-\infty, 0$` yields $x \in]-\infty, 0]$. Contrariwise, when you want to write algebra on intervals then *you must leave a blank space between the second bracket and the +/- operations*, e.g. `$[a, b] + [c, d$` yields $[a, b] + [c, d]$ but `$[a, b]+ [c, d$` yields $[a, b]+[c, d]$.

Let us also mention other approaches with the `\interval` macro from the `interval` package [17], or `\DeclarePairedDelimiters` from the `mathtools` package [7] (but the latter is incompatible with `ibackets` for brackets management).

¹²This macro gives bad results with normal sized parenthesis.

¹³Another solution is to define `\abs` with the `\DeclarePairedDelimiter` command from the `mathtools` package [7].

¹⁴This functionality is optional because it causes error when using a command defined by `\DeclarePairedDelimiter` [7] with square brackets.

¹⁵It's the only option of the `mismath` package.

2.6 Environments for systems of equations and small matrices

`system` The `system` environment produces a system of equations:

$$\begin{array}{l} \text{\$}\backslash\text{begin}\{\text{system}\} \\ \quad x=1+2t \quad \backslash\backslash \quad y=2-t \quad \backslash\backslash \quad z=-3-t \\ \text{\$}\backslash\text{end}\{\text{system}\} \end{array} \quad \left\{ \begin{array}{l} x = 1 + 2t \\ y = 2 - t \\ z = -3 - t \end{array} \right.$$

`\systemsep` This first example could also have been produced with `cases` environment from `amsmath` package, although `cases` places mathematical expressions closer to the bracket (which makes sense considering its use). `\systemsep` enables to set the gap between the bracket and the expressions, set by default to `\medspace`. This gap may be reduced, for instance: `\renewcommand{\systemsep}{\thinspace}`, or enlarged with `\thickspace` (and with `\renewcommand{\systemsep}{}` we obtain what `cases` do).

`system[(coldef)]` By default, a system is written like an array environment with only one column, left aligned. The environment has an optional argument to create several columns, specifying their alignment, with the same syntax than the `array` environment of `TeX`: `\begin{system}[cl]` produces a two-column system, the first one being centered, the second being left aligned, such as in the following example:

$$\begin{array}{l} \text{\$}\backslash\text{begin}\{\text{system}\}[cl] \\ \quad y \ \& \ = \dfrac{1}{2}x - 2 \quad \backslash\backslash[1ex] \\ \quad (x,y) \ \& \ \neq (0,-2) \\ \text{\$}\backslash\text{end}\{\text{system}\} \end{array} \quad \left\{ \begin{array}{l} y = \frac{1}{2}x - 2 \\ (x,y) \neq (0,-2) \end{array} \right.$$

`\systemstretch` Default spacing between the lines of a `system` environment has been slightly enlarged compared to the one from `array` environments (from 1.2 factor). This spacing may be changed by typing `\renewcommand{\systemstretch}{\stretch}`, inside the current mathematical environment (for a local change) or outside (for a global change). By default, `stretch`'s value is 1.2. In addition we can use the end of line with a spacing option such as it has been done above with `\backslash[1ex]`.

Another example with `\begin{system}[rl@{\quad}l]`¹⁶:

$$\left\{ \begin{array}{lll} x + 3y + 5z = 0 & R_1 \\ 2x + 2y - z = 3 & R_2 \\ 3x - y + z = 2 & R_3 \end{array} \right. \iff \left\{ \begin{array}{lll} x + 3y + 5z = 0 & R_1 \\ 4y + 11z = 3 & R_2 \leftarrow 2R_1 - R_2 \\ 5y + 7z = -1 & R_3 \leftarrow \frac{1}{2}(3R_1 - R_3) \end{array} \right.$$

Let's mention the `systeme` package [18] which deals with linear systems with a lighter syntax and automatic alignments on +, -, =, and also the `spalign` package [19] which moreover produces nice alignments for matrices (with spaces and semicolons as delimiters).

`spmatrix` The `amsmath` package provides various environments to typeset matrices: for instance `pmatrix` surrounds the matrix with parenthesis or `smallmatrix` typesets a small matrix that can even be inserted in a text line. We provide a combination of the

¹⁶@{...} sets inter-column space.

two with `spmatrix`:

`\vec{u}\begin{spmatrix}-1\\2\end{spmatrix}` yielding $\vec{u}\begin{pmatrix}-1\\2\end{pmatrix}$.

The `mathtools` package enhance `amsmath` matrices environments and provides also a small matrix environment with parenthesis. Furthermore, with starred version `\begin{psmallmatrix*}[\langle col \rangle]`, you can choose the alignment inside the columns (c, l or r). But sadly, the space before the left parenthesis is too narrow regarding to the space inside the parenthesis. Compare previous $\vec{u}\begin{pmatrix}-1\\2\end{pmatrix}$ with $\vec{u}\begin{pmatrix}-1\\2\end{pmatrix}$.

For typesetting various kind of matrices, let's mention the awesome `nicematrix` package by François Pantigny [20].

2.7 Displaymath in double columns

`mathcols` The `mathcols` environment enables to arrange “long” calculation in double columns, separated with a central rule, as shown in the following example. But the `multicol` package must be loaded in the preamble. It activates the mathematical mode in display style and with an `aligned` environment.

$$\begin{array}{l|l} \frac{1}{2 \times \left(\frac{1}{4}\right)^n + 1} \geq 0.999 & \iff 4^n \geq 1998 \\ \iff 1 \geq 1.998 \left(\frac{1}{4}\right)^n + 0.999 & \iff n \ln 4 \geq \ln(1998) \\ \iff 0.001 \geq \frac{1.998}{4^n} & \iff n \geq \frac{\ln(1998)}{\ln 4} \approx 5.4 \\ & \iff n \geq 6 \end{array}$$

`\changeacol` The `\changeacol` macro causes a change of column; alignment is produced using the classic delimiters `&` and `\\`.

```
\begin{mathcols}
& \frac{1}{2 \mul {\pow{\frac{1}{4}}{n}} + 1} \geq 0.999 \\
\iff & 1 \geq 1.998 \pow{\frac{1}{4}}{n} + 0.999 \\
\iff & 0.001 \geq \frac{1.998}{4^n} \\
\changeacol
& \iff 4^n \geq 1998 \\
& \iff n \ln 4 \geq \ln(1998) \\
& \iff n \geq \frac{\ln(1998)}{\ln 4} \approx 5.4 \\
& \iff n \geq 6
\end{mathcols}
```

2.8 Deprecated commands

Here we present a summary table of deprecated commands, used until version 2.2. They produce a warning message but are still working and will be maintained for now. These deprecated commands worked only in the preamble, globally, and there was no inverse switch. Therefore they are replaced by the more powerful and more general macro `\MathUp` which can be placed anywhere and has an inverse switch `\MathIt`.

Deprecated command	New alternative
<code>\enumber</code>	<code>\MathUp{e}</code>
<code>\inumber</code>	<code>\MathUp{i}</code>
<code>\jnumber</code>	<code>\MathUp{j}</code>
<code>\PEupright</code>	<code>\MathUp{P}\MathUp{E}</code>

`\MathNumbers` may be used instead of `\MathUp` with an argument containing all the constants you want to be typeset in roman (among ‘e, i, j’). And `\MathProba{P,E}` may be used instead of `\MathUp{P}\MathUp{E}` and you can add also V in its argument to refer to variance.

In version 2.3 we tried a way to replace these deprecated commands by package options based on keyval. This less efficient method is abandoned. And thus the command `\mismathset` is obsolete. Another command, `\paren`, used before version 2.0, is no longer supported.

3 Implementation

```

1 \newif\ifmm@ibrackets % initialized to false
2 \DeclareOption{ibrackets}{\mm@ibracketstrue}
3 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{amsmath}}
4 \ProcessOptions \relax
5 \@ifpackageloaded{amsmath}{}{\RequirePackage{amsmath}}
6 \@ifpackageloaded{mathtools}{}{\RequirePackage{mathtools}}
7 \@ifpackageloaded{esvect}{}{\RequirePackage[b]{esvect}}
8 \RequirePackage{ifthen}
9 \RequirePackage{xparse} % for \NewDocumentCommand
10 \RequirePackage{xspace}
11 \RequirePackage{iftex}
12 \ifmm@ibrackets\RequirePackage{ibrackets}\fi
13

```

The above conditional packages loading avoids “option clash” errors if these packages have been previously loaded with other options.

`\bslash` The `\bslash` macro comes from Frank Mittelbach’s `doc.sty` package. It can also be used in other documents instead of `\textbackslash` (which doesn’t work inside warnings).

```

14 {\catcode'\=\z@ \catcode'\=12 \gdef\bslash{\}} % \bslash command
15

```

`\mm@warning` The three following internal macros are meta commands for conditional macro definition with a warning message if the macro already exists. They should be useful in other packages.

`\mm@macro`

`\mm@operator`

```

16 \newcommand\mm@warning[1]{
17   \PackageWarningNoLine{mismath}{
18     Command \bslash #1 already exist and will not be redefined}

```

```

19 }
20 \newcommand\mm@macro[2]{
21   \ifundefined{#1}{
22     \expandafter\def\csname #1\endcsname{#2}
23   }\mm@warning{#1}}
24 }
25 \NewDocumentCommand\mm@operator{0{#3}mm}{%
26   \ifundefined{#1}{
27     \DeclareMathOperator{#2}{#3}
28   }\mm@warning{#1}}
29 }
30

```

To produce the correct upright shape font when working with the beamer package, you don't have to use `\mathrm` but `\mathup` (based on `\operatorfont` from the `amsopn` package). This command works also fine with other sans serif fonts like `cmbright`.

Moreover for beamer, which changes the family default font (sans serif) `\e`, `\i`, `\j` have no effect without `\AtBeginDocument`.

`\AtBeginDocument` is also necessary to redefine `\i` when calling the `hyperref` package which overwrites the `\i` definition.

```

31 \providecommand{\mathup}[1]{\operatorfont #1} % also in kpfonts
32 \mm@macro{e}{\mathup{e}}
33 \AtBeginDocument{\let\oldi\i \let\oldj\j
34   \renewcommand{\i}{\TextOrMath{\oldi}{\mathup{i}}}
35   \renewcommand{\j}{\TextOrMath{\oldj}{\mathup{j}}}}
36

```

The following macros are switches that transform in roman vs italic any chosen letter in math mode. They can be used anywhere. To get a letter in roman instead of italic, we have to change the digit of mathcode that represent the family: 1 to 0.

For example, except for `Lua®TeX`, mathcode of the 'e' letter is: 'e'="7165 (decimal 29029), the second digit '1' meaning "italic". To get a roman 'e', we have to change his mathcode in "7065.

When called in the preamble, `\AtBeginDocument` is necessary for using with the beamer package. In the preamble, `\math@family{#1}{0}` is equivalent to `\DeclareMathSymbol{#1}{\mathalpha}{operators}{' #1}`.

```

37 \newcount\mm@charcode
38 \newcount\mm@charclass
39 \newcount\mm@charfam
40 \newcount\mm@charslot
41
42 \newcommand*\math@family[2]{%
43   \mm@charfam=#2
44   \ifluatex
45     \mm@charclass=\Umathcharclass'#1
46     \% \mm@charfam=\Umathcharfam'#1
47     \mm@charslot=\Umathcharslot'#1
48     \Umathcode'#1= \mm@charclass \mm@charfam \mm@charslot

```

```

49   \else
50     \mm@charcode=\mathcode'#1
51     % extract charclass
52     \@tempcnta=\mm@charcode
53     \divide\@tempcnta by "1000
54     \multiply\@tempcnta by "1000 % charclass
55     \mm@charclass=\@tempcnta
56     % extract charslot
57     \@tempcnta=\mm@charcode
58     \@tempcntb=\mm@charcode
59     \divide\@tempcnta by "100
60     \multiply\@tempcnta by "100 % charclass + charfam
61     \advance\@tempcntb by -\@tempcnta % charslot
62     \mm@charslot=\@tempcntb
63     % construct charcode
64     \mm@charcode=\mm@charclass
65     \multiply\mm@charfam by "100
66     \advance\mm@charcode by \mm@charfam
67     \advance\mm@charcode by \mm@charslot
68     \mathcode'#1=\mm@charcode
69   \fi
70 }
71
72 \newcommand*\MathFamily[2]{%
73   \ifx\@onlypreamble\@notprerr
74     \math@family{#1}{#2}
75   \else % before \begin{document}
76     \AtBeginDocument{\math@family{#1}{#2}}
77   \fi
78 }
79
80 \newcommand*\MathUp[1]{\MathFamily{#1}{0}}
81 \newcommand*\MathIt[1]{\MathFamily{#1}{1}}
82

```

On the same model we could also create some other macros to set any letter in bold or sans serif, but unfortunately there is no family number associated by default to these typefaces. It depends on the font package that is loaded and it can also depend on which `\DeclareSymbolFont` is used.

In complement to `\MathUp` and `\MathIt`, we provide the two following commands to set in roman a group of letters among 'e, i, j' for mathematical constants or 'P, E, V' for probability operators.

```

83 \newcommand*\MathNumbers[1]{%
84   \in@{e}{#1} \ifin@ \MathUp{e} \fi
85   \in@{i}{#1} \ifin@ \MathUp{i} \fi
86   \in@{j}{#1} \ifin@ \MathUp{j} \fi
87 }
88
89 \newcommand*\MathProba[1]{%

```

```

90   \in@{P}{#1} \ifin@ \MathUp{P} \fi
91   \in@{E}{#1} \ifin@ \MathUp{E} \fi
92   \in@{V}{#1} \ifin@ \MathUp{V} \fi
93 }
94

```

`\apply` The inverse global switch `\MathNormal` acts on any comma separated list thanks to the `\apply` macro which works as follows: `\apply\macro{comma,list}` expands to `\macro{comma}\macro{list}`. We could also use `\apply\MathUp{e,i,j}` instead of `\MathNumbers{e,i,j}`. I found this macro on Tex.RSSing.com by searching “TeX How to iterate over a comma separated list?” The answer was given by “wipet” on 2021/02/26. I don’t know who is wipet but I thank him for this powerful macro! Unfortunately usual loop instructions like `\@for` or `\foreach` do not work and produce the error message “! Improper alphabetic constant”. Indeed `\def\letter{A} \MathUp{\letter}` fails, the control sequence `\letter` is not considered as the single character ‘A’.

```

95 \def\apply#1#2{\apply@#1#2,\apply@,}
96 \def\apply@#1#2,{\ifx\apply@#2\empty
97   \else #1{#2}\afterfi@{\apply@#1}\fi}
98 \def\afterfi@#1#2\fi{\fi#1}
99
100 \newcommand*\MathNormal[1]{% list argument
101   \apply\MathIt{#1}
102 }
103

```

The following commands are deprecated but still work. They were intended to set some letters in upright shape by default in math mode, but worked only in the preamble. This is now managed by the more powerful `\MathUp` command. The old commands are maintained for now for compatibility reasons.

```

104 \newcommand{\enumber}{%
105   \PackageWarning{mismath}{Command \string\enumber\space
106     is deprecated, \MessageBreak
107     use \bslash MathUp{e} instead}
108   \MathUp{e}
109 }
110 \newcommand{\inumber}{%
111   \PackageWarning{mismath}{Command \string\inumber\space
112     is deprecated, \MessageBreak
113     use \bslash MathUp{i} instead}
114   \MathUp{i}
115 }
116 \newcommand{\jnumber}{%
117   \PackageWarning{mismath}{Command \string\jnumber\space
118     is deprecated, \MessageBreak
119     use \bslash MathUp{j} instead}
120   \MathUp{j}
121 }
122 \newcommand{\PEupright}{%

```

```

123 \PackageWarning{mismath}{Command \string\PEupright\space
124 is deprecated, \MessageBreak
125 use \bslash MathUp{P} and \bslash MathUp{R} instead}
126 \MathUp{P}\MathUp{E}
127 }
128

```

The Greek letter pi must be managed in a different way. The switches are called `\pinumber` and `\pinormal`. When given without argument, `\pinumber` uses the LGR font encoding. A particularity of the fontenc package is that it can be loaded several times with different options without “option clash” error.

```

129 \newcommand*\pinumber[1][]{
130   \ifundefined{itpi}{\let\itpi\pi}{}
131   \ifthenelse{\equal{#1}{} }{
132     \ifx\@onlypreamble\@notprerr
133       \ifundefined{savedpi}{
134         \PackageWarning{mismath}{%
135           \bslash pinumber without argument\MessageBreak
136           must be used in the preamble first\MessageBreak
137           to load LGR fontenc for upright pi}
138       }\let\pi\savedpi
139     \else % in the preamble
140       \RequirePackage[LGR,T1]{fontenc}
141       \DeclareSymbolFont{UpGr}{LGR}{lmr}{m}{n}
142       \let\pi\relax
143       \DeclareMathSymbol{\pi}\mathalpha{UpGr}{"70}
144       \let\savedpi\pi
145     \fi
146   }{
147     \ifundefined{#1}{
148       \PackageWarning{mismath}{%
149         Value #1 must be a valid
150         command name\MessageBreak for pinumber,
151         but command \bslash #1\space
152         is undefined.\MessageBreak
153         Perhaps a missing package}
154     }\renewcommand{\pi}{%
155       \csname #1\endcsname}
156   }
157 }
158 }
159
160 \newcommand{\pinormal}{\ifundefined{itpi}{}{\let\pi\itpi}}
161

```

And now the commands for vectors (and tensors).

```

162 \newboolean{arrowvect}
163 \setboolean{arrowvect}{true}
164 \newcommand{\arrowvect}{\setboolean{arrowvect}{true}}
165 \newcommand{\boldvect}{\setboolean{arrowvect}{false}}

```



```

166 \newcommand{\boldvectcommand}{\boldsymbol} % from amsbsy package
167 \mm@macro{vect}{\ifthenelse{\boolean{arrowvect}}{
168     \vv}{\boldvectcommand}} % doesn't work well with \if... \fi
169 \newcommand*{\hvect}[1]{\vect{\vphantom{t}#1}}
170 \newcommand*{\hvec}[1]{\vec{\vphantom{t}#1}}
171
172 \newcommand*{\@norm}[1]{
173     \mbox{\raisebox{1.75pt}{\small$\bigl\Vert$}} #1
174     \mbox{\raisebox{1.75pt}{\small$\bigr\Vert$}} }
175 % works better than with relative length
176 \newcommand*{\@@norm}[1]{
177     \mbox{\footnotesize\raisebox{1pt}{\small$\Vert$}} #1
178     \mbox{\footnotesize\raisebox{1pt}{\small$\Vert$}} }
179 \newcommand*{\@@@norm}[1]{
180     \mbox{\tiny\raisebox{1pt}{\small$\Vert$}} #1
181     \mbox{\tiny\raisebox{1pt}{\small$\Vert$}} }
182 \ifundefined{norm}{\providecommand*{\norm}[1]{
183     \mathchoice{\@norm{#1}}{\@norm{#1}}{\@@norm{#1}}{\@@@norm{#1}}
184     }
185     }{\mm@warning{norm}} % bad result with libertineustmath
186
187 \DeclareMathAlphabet{\mathbfsfit}{\encodingdefault}{\sfdefault}{bx}{it}
188 \newcommand{\tensor}{\mathbfsfit} % isomath uses \mathsfbfit
189

```

And now all the other commands.

```

190 \mm@macro{di}{\mathop{}}!\mathup{d}}
191 \newcommand\probastyle{}
192 \let\Par\P % end of paragraph symbol
193 \renewcommand{\P}{\operatorname{\probastyle{P}}}
194 \mm@macro{E}{\operatorname{\probastyle{E}}}
195 \mm@macro{V}{\operatorname{\probastyle{V}}}
196
197 \mm@operator{adj}{adj}
198 \mm@operator{Aut}{Aut}
199 \mm@operator{codim}{codim}
200 \mm@operator{Conv}{Conv}
201 \mm@operator{cov}{cov}
202 \mm@operator{Cov}{Cov}
203 \mm@macro{curl}{\operatorname{\vect{\mathup{curl}}}}
204 \mm@operator{divg}{\divg}{div}
205 \mm@operator{End}{End}
206
207 \mm@operator{erf}{erf}
208 \mm@macro{grad}{\operatorname{\vect{\mathup{grad}}}}
209 \mm@operator{id}{id} % mathop or mathord ?
210 \mm@operator{Id}{Id}
211 \mm@operator{im}{im}
212 \let\oldIm\Im \renewcommand{\Im}{\operatorname{Im}}
213 \mm@operator{lb}{lb}

```

```

214 \mm@operator{\lcm}{lcm}
215
216 \mm@operator{\rank}{rank}
217 \let\oldRe\Re \renewcommand{\Re}{\operatorname{Re}}
218 \mm@macro{rot}{\operatorname{\vect{\mathup{rot}}}}
219 \mm@operator{\sgn}{sgn}
220 \mm@operator{\sinc}{sinc}
221 \mm@operator[spa]{\spa}{span}
222 \mm@operator{\tr}{tr}
223 \mm@operator{\var}{var}
224 \mm@operator{\Var}{Var}
225 \mm@operator[Zu]{\Zu}{Z}
226
227 \mm@operator{\arccot}{arccot}
228 \mm@operator{\sech}{sech}
229 \mm@operator{\csch}{csch}
230 \mm@operator{\arsinh}{arsinh}
231 \mm@operator{\arcosh}{arcosh}
232 \mm@operator{\artanh}{artanh}
233 \mm@operator{\arcoth}{arcoth}
234 \mm@operator{\arsech}{arsech}
235 \mm@operator{\arcsch}{arcsch}
236
237 \mm@operator[bigO]{\bigO}{\mathcal{O}}
238 \mm@operator[bigo]{\bigo}{O}
239 \mm@operator[lito]{\lito}{o}
240
241 \mm@macro{mathset}{\mathbf}
242 \mm@macro{R}{\ensuremath{\mathset{R}}\xspace}
243 %\mm@macro{C}{\ensuremath{\mathset{C}}\xspace}
244 \AtBeginDocument{\@ifpackagewith{babel}{russian}{\mm@warning{C}
245     }{\mm@macro{C}{\ensuremath{\mathset{C}}\xspace}}}
246 }
247 \mm@macro{N}{\ensuremath{\mathset{N}}\xspace}
248 \mm@macro{Z}{\ensuremath{\mathset{Z}}\xspace}
249 \mm@macro{Q}{\ensuremath{\mathset{Q}}\xspace}
250 \mm@macro{F}{\ensuremath{\mathset{F}}\xspace}
251 \mm@macro{K}{\ensuremath{\mathset{K}}\xspace}
252
253 \mm@macro{ds}{\displaystyle}
254 \mm@macro{dlim}{\lim\limits}
255 \mm@macro{dsum}{\sum\limits}
256 \mm@macro{dprod}{\prod\limits}
257 \mm@macro{dcup}{\bigcup\limits}
258 \mm@macro{dcap}{\bigcap\limits}
259
260 \mm@macro{lbar}{\overline}
261 \@ifundefined{hlbar}{
262     \providecommand*{\hlbar}[1]{\overline{\vphantom{t}\#1}}}{
263     \mm@warning{hlbar} }

```

```

264 \newcommand\@eqdef{\stackrel{\mathup{def}}{=}}
265 \newcommand\@@eqdef{\stackrel{\mathrm{\Delta}}{=}}
266 \mm@macro{eqdef}{\@ifstar{\@eqdef}{\@eqdef}}
267 \mm@macro{unbr}{\underbrace}
268 \mm@macro{iif}{if and only if\hspace}
269

```

We used `\mathrm` before `\Delta` in the case of defining capital Greek letters in italic (for example with the `fixmath` package). Without `\mbox{}`, space produced by `\` in macro `\then` would be suppressed in tables.

```

270 \mm@macro{then}{\ \Longrightarrow \ \mbox{ } }
271 \@ifundefined{txt}{
272   \providecommand*\txt[1]{\quad\text{\#1}\quad} }{
273   \mm@warning{txt} }
274 \mm@macro{mul}{\mathord{\times}}
275 \@ifundefined{pow}{
276   \providecommand*\pow[2]{\left( \#1 \right)^{\!\#2} }{
277   \mm@warning{pow} }
278 \@ifundefined{abs}{
279   \providecommand*\abs[1]{\left\vert\!#1\right\vert }{
280   \mm@warning{abs} }
281 \@ifundefined{lfrac}{
282   \providecommand*\lfrac[2]{\frac{\;#1\;}{\;#2\;}} }{
283   \mm@warning{lfrac} }
284
285 \newcommand{\systemstretch}{1.2}
286 \newcommand{\systemsep}{\medspace}
287 \newenvironment{system}[1][1]{
288   \renewcommand{\arraystretch}{\systemstretch}
289   \setlength{\arraycolsep}{0.15em}
290   \left\{\begin{array}{@{\systemsep}\#1@{}} %
291 }\end{array}\right.}
292
293 \newenvironment{spmatrix}{
294   \left(\begin{smallmatrix}
295 }\end{smallmatrix}\right)}
296
297 \newenvironment{mathcols}{% needs multicol package
298   \renewcommand{\columnseprule}{0.1pt}
299   \begin{multicols}{2}
300     \par\noindent\hfill
301     \begin{math}\begin{aligned}\displaystyle
302 }{%
303     \end{aligned}\end{math} \hfill\mbox{ }
304   \end{multicols}
305 }
306 \newcommand{\change col}{%
307   \end{aligned}\end{math} \hfill\mbox{ }
308   \par\noindent\hfill
309   \begin{math}\begin{aligned}\displaystyle

```

References

- [1] *Typesetting mathematics for science and technology according to ISO 31/XI*, Claudio Beccari, TUGboat Volume 18 (1997), No. 1.
<http://www.tug.org/TUGboat/tb18-1/tb54becc.pdf>.
- [2] *Typefaces for Symbols in Scientific Manuscripts*,
<https://www.physics.nist.gov/cuu/pdf/typefaces.pdf>.
- [3] *Guide for the Use of the International System of Units (SI)*, NIST (National Institute of Standards and Technology), updated March 4, 2020
<https://www.nist.gov/pml/special-publication-811>.
- [4] *On the Use of Italic and up Fonts for Symbols in Scientific Text*, I.M. Mills and W.V. Metanomski, ICTNS (Interdivisional Committee on Terminology, Nomenclature and Symbols), dec 1999, https://old.iupac.org/standing/idcns/italic-roman_dec99.pdf.
- [5] *esvect – Typesetting vectors with beautiful arrow with \LaTeX* , Eddie Sautrais, CTAN, v1.3 2013/07/11.
- [6] *amsmath – $\mathcal{A}\mathcal{M}\mathcal{S}$ mathematical facilities for \LaTeX* , Frank Mittelbach, Rainer Schöpf, Michael Downes, Davis M. Jones, David Carlisle, CTAN, v2.17n 2022/04/08.
- [7] *The mathtools package*, Morten Høgholm, Lars Madsen, CTAN, v1.29 2022/06/29.
- [8] *The fixmath package for \LaTeX* , Walter Schmidt, CTAN, v0.9 2000/04/11.
- [9] *isomath – Mathematical style for science and technology*. Günter Milde, CTAN, v0.6.1 2012/09/04.
- [10] *PM-ISOMath, The Poor Man ISO math bundle*, the pm-isomath package by Claudio Beccari, CTAN, v1.2.00 2021/08/04.
- [11] *The upgreek package for \LaTeX* , Walter Schmidt, CTAN, v2.0 2003/02/12.
- [12] *The mathdesign package*, Paul Pichaureau, CTAN, v2.31 2013/08/29.
- [13] *The textalpha package* (part of the greek-fontenc bundle), Günter Milde, CTAN, v2.1 14/06/2022.
- [14] *Kp-Fonts – The Johannes Kepler project*, Christophe Caignaert, CTAN, v3.34 20/09/2022.
- [15] *Fourier-GUTenberg*, Michel Bovani, CTAN, v1.3 30/01/2005.
- [16] *The lgrmath package*, Jean-François B., CTAN, v1.0 2022/11/16.

- [17] *The interval package*. Lars Madsen, CTAN, v0.4 2019/03/06.
- [18] *L'extension pour \TeX et \LaTeX systeme*, Christian Tellechea, CTAN v0.32 2019/01/13.
- [19] *The spalign package*, Joseph Rabinoff, CTAN, 2016/10/05.
- [20] *The package nicematrix*, François Pantigny, CTAN, v6.14 2023/02/18.
- [21] *L'extension frenchmath*, Antoine Missier, CTAN, v2.5 2023/02/24.
- [22] *Intelligent brackets – The ibrackets package* Antoine Missier, CTAN, v1.1, 2022/12/26.
- [23] *The Not So Short Introduction to $\text{\LaTeX}2_{\epsilon}$* , the lshort package by Tobias Oetiker, Hubert Partl, Irene Hyna and Elisabeth Schlegl, CTAN, v6.4 2021/04/09.
<http://tug.ctan.org/info/lshort/english/lshort.pdf>.
- [24] *The \LaTeX Companion*, Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, Chris Rowley, 2nd edition, Pearson Education, 2004.