mismath
Miscellaneous mathematical macros[*]

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

March 5, 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According to the International Standards ISO 31-0:1992 to ISO 31-13:1992, superseded by ISO 80000-2:2009, mathematical constants e, i, π should be typeset in roman (upright shape) and not in italic (sloping shape) like variables (see [1] [2]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 Saudrais [5] and we provide a few more macros related to vectors with arrows, in particular to

---

[*] This document corresponds to `mismath` v2.7, dated 2023/03/05. Thanks to François Bastouil for help in English translation.

improve the typesetting of the norm: $\|\overrightarrow{AB}\|$ instead of LaTeX version $\|\overrightarrow{AB}\|$ which is not vertically adjusted, or worse $\left\|\overrightarrow{AB}\right\|$ (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 Beccari [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 shape[1]. 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

---

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

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

`\e`
`\i`
`\j`

To avoid to stuff a document that contains many e or i constants with `\mathup{e}` or `\mathup{i}`, the package provides `\e` command for Euler's number and `\i` or `\j` for imaginary numbers. Let's notice that `\i` and `\j` already exist in LaTeX: using in LR mode, they produce 'ı, ȷ' 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 LaTeX 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`
`\MathNormal`

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}`. 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 $\mathrm{\pi}$ 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], libertinust1math, pxgreeks, txgreeks, 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 pi, 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`.

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

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

`$e^{i\pi} = -1$` yields $\mathrm{e}^{\mathrm{i}\pi} = -1$.

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 pi 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 family[4].

`\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` command[5], produces vectors with arrows (thanks to the esvect package of Eddie Saudrais[6]) which are more elegant than those produced by LaTeX'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:

```
\[ \boldvect \vect{v}
   =\lambda\vect{e}_x+\mu\vect{e}_y. \]
```

$$\boldsymbol{v} = \lambda\boldsymbol{e}_x + \mu\boldsymbol{e}_y.$$

`\boldvectcommand` By default `\boldvect` uses the `\boldsymbol` command[7] from amsbsy package, loaded by amsmath. But other packages producing bold italic can be preferred, e.g. `\mathbold` from fixmath package or `\mathbfit` from isomath or `\bm` from bm package. For that, redefine `\boldvectcommand`, for instance, after loading fixmath:

`\renewcommand\boldvectcommand{\mathbold}`.

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

---

[4] This 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.

[5] As 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:

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

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 LaTeX command `\vec` (from height of t letter):

- $\mathscr{P} = \vec{f} \cdot \vec{v}$, obtained with `\hvec{v}`, is better than $\mathscr{P} = \vec{f} \cdot \vec{v}$.
- $\vec{f} = m\vec{a}$, 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\Vert` and `\right\Vert` 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 $\left\|\vec{h}\right\|$. 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. $\mathrm{e}^{\|\vec{h}\|}$.

`\mathbfsfit` `\tensor` 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 $\boldsymbol{\mathsf{T}}$.

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):

```
\[ \iint xy\di x\di y \]
```

$$\iint xy\,\mathrm{d}x\,\mathrm{d}y$$

```
\[ m\frac{\di^2x}{\di t^2}
   + h\frac{\di x}{\di t} + kx = 0 \]
```

$$m\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + h\frac{\mathrm{d}x}{\mathrm{d}t} + kx = 0$$

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

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

`\P` `\E` To refer to probability[8] and expectation the proper use is to typeset capital letters P, E in roman as for any standard function identifier. This is obtained with `\P` and `\E`.

`\Par` The `\P` command already existed to refer to the end of paragraph symbol ¶ and has been redefined, but this symbol can still be obtained with `\Par`.

`\V` Variance is generally denoted by var or Var (see table below), but some authors prefer to use V, produced by `\V`.

`\MathProba` `\MathNormal` In the same way as for e, i or j, you can use `\MathUp{P}`, `\MathUp{E}` or `\MathUp{V}` to avoid typing many `\P`, `\E` or `\V`. But you can also do that in a single command with `\MathProba`, for example `\MathProba{P,E}` and we get the inverse switch with `\MathIt` for any individual letter or `\MathNormal` for a list.

`\probastyle` Some authors use "blackboard bold" font to represent probability, expectation and variance: $\mathbb{P}, \mathbb{E}, \mathbb{V}$. The `\probastyle` macro sets the appearance of `\P`, `\E` and `\V`: for instance `\renewcommand\probastyle{\mathbb}`[9] brings the previous "openwork" letters. `\mathbb` comes from amsfonts package (loaded by amssymb but also available standalone) which has to be called in the preamble.

The following standard operator names are defined in mismath:

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

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: $\mathrm{Z}(G)$ (from German Zentrum).

`\oldRe` `\oldIm` The `\Re` and `\Im` 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 `\oldRe` and `\oldIm`.

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

---

[8] LATEX provides also `\Pr` which gives Pr.

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

`\arccot`	arccot	`\arsinh`	arsinh	`\arcoth`	arcoth
`\sech`	sech	`\arcosh`	arcosh	`\arsech`	arsech
`\csch`	csch	`\artanh`	artanh	`\arcsch`	arcsch

`\bigO`
`\bigo`
`\lito`

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

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

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: $\mathbf{R},\mathbf{C},\mathbf{Z},\mathbf{N},\mathbf{Q}$, whereas “openwork” letters ($\mathbb{R},\mathbb{Z},\ldots$) are reserved for writing at blackboard [23]; and likewise to designate a field: $\mathbf{F}$ or $\mathbf{K}$ (Körper in German). We get these symbols with the macros[10]:

`\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 amsfonts package (which provides the “blackboard bold” typeface, also loaded by amssymb).

`\ds`

The `\displaystyle` command being very common, the `\ds` alias 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`
`\dsum`
`\dprod`
`\dcup`
`\dcap`

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

`$\dlim_{x\to +\infty}\frac{1}{x}$` gives $\displaystyle\lim_{x\to+\infty}\tfrac{1}{x}$.

`\lbar`
`\hlbar`

Large bars over expressions are obtained with `\overline` or, shorter, its alias `\lbar`, to get for instance $\overline{z_1z_2}$. 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.

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

---

[10] The `\C` macro is incompatible with `russian` option of babel; it will not be redefined in that case.

`\eqdef`
`\eqdef*`
The `\eqdef` macro writes equality symbol topped with 'def' or with '$\vartriangle$' for `\eqdef*` (thanks to the LaTeX command `\stackrel`):

```
$ \e^{\i\theta} \eqdef
  \cos\theta + \i\sin\theta $
```

$$\mathrm{e}^{\mathrm{i}\theta} \stackrel{\text{def}}{=} \cos\theta + \mathrm{i}\sin\theta$$

```
$ \e^{\i\theta} \eqdef*
  \cos\theta + \i\sin\theta $
```

$$\mathrm{e}^{\mathrm{i}\theta} \triangleq \cos\theta + \mathrm{i}\sin\theta$$

`\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`
`\txt`
The `\then` macro produces the symbol $\Longrightarrow$ surrounded by large spaces as the standard macro `\iff` does it with $\Longleftrightarrow$. 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:

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

$$\ln x = a \quad\Longrightarrow\quad x = \mathrm{e}^a \quad\text{rather than}\quad \ln x = a \Longrightarrow x = \mathrm{e}^a$$

`\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-

---

[11] The `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 ⟨*expr*⟩ between parentheses and puts the exponent ⟨*pow*⟩ slightly closer to the right parenthesis[12]. Compare:

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

`\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\vert` ... `\right\vert` or, more simply, the `\abs{`...`}` command which is equivalent[13].

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

```
\[ \lbar{Z} =
   \lfrac{\lbar{z_1-z_2}}{\lbar{z_1+z_2}} \]
```

$$\overline{Z} = \frac{\;\overline{z_1-z_2}\;}{\;\overline{z_1+z_2}\;}$$

`[ibrackets]` 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 ibrackets package [22] which can be optionally[14] loaded by mismath with `ibrackets` package option[15].

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

$$\begin{array}{rll} & x \in \mathopen{]}{-\pi},0\mathclose{[} \cup \mathopen{]}2\pi,3\pi\mathclose{[} & \text{with ibrackets,} \\ \text{instead of} & x \in]-\pi,0[\cup]2\pi,3\pi[& \text{without ibrackets.} \end{array}$$

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 ibrackets: 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 ibrackets for brackets management).

---

[12] This macro gives bad results with normal sized parenthesis.

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

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

[15] It'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:

```
$\begin{system}
    x=1+2t \\ y=2-t \\ z=-3-t
\end{system}$
```

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

`\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 it's 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 LaTeX: `\begin{system}[cl]` produces a two-column system, the first one being centered, the second being left aligned, such as in the following example:

```
$\begin{system}[cl]
    y & =\dfrac{1}{2}x-2 \\[1ex]
    (x,y) & \neq (0,-2)
\end{system}$
```

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

`\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 `\\[1ex]`.

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

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

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 $\vec{u}\,\left(\begin{smallmatrix}-1\\2\end{smallmatrix}\right)$.

The mathtools package enhance amsmath matrices environments and provides also a small matrix environment with parenthesis. Furthermore, with starred version `\begin{psmallmatrix*}[`⟨*col*⟩`]`, 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 $\vec{u}\,\left(\begin{smallmatrix}-1\\2\end{smallmatrix}\right)$ with $\vec{u}\left(\begin{smallmatrix}-1\\2\end{smallmatrix}\right)$.

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.

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

`\changecol` The `\changecol` 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} \\
\changecol
    & \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
`\enumber`	`\MathUp{e}`
`\inumber`	`\MathUp{i}`
`\jnumber`	`\MathUp{j}`
`\PEupright`	`\MathUp{P}\MathUp{E}`

`\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

```
\newif\ifmm@ibrackets % initialized to false
\DeclareOption{ibrackets}{\mm@ibracketstrue}
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{amsmath}}
\ProcessOptions \relax
\@ifpackageloaded{amsmath}{}{\RequirePackage{amsmath}}
\@ifpackageloaded{mathtools}{}{\RequirePackage{mathtools}}
\@ifpackageloaded{esvect}{}{\RequirePackage[b]{esvect}}
\RequirePackage{ifthen}
\RequirePackage{xparse} % for \NewDocumentCommand
\RequirePackage{xspace}
\RequirePackage{iftex}
\ifmm@ibrackets\RequirePackage{ibrackets}\fi
```

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).

```
{\catcode`\|=\z@ \catcode`\\=12 |gdef|bslash{\}} % \bslash command
```

`\mm@warning`
`\mm@macro`
`\mm@operator`

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.

```
\newcommand\mm@warning[1]{
    \PackageWarningNoLine{mismath}{
        Command \bslash #1 already exist and will not be redefined}
}
\newcommand\mm@macro[2]{
    \@ifundefined{#1}{
        \expandafter\def\csname #1\endcsname{#2}
    }{\mm@warning{#1}}
}
\NewDocumentCommand\mm@operator{O{#3}mm}{%
    \@ifundefined{#1}{
        \DeclareMathOperator{#2}{#3}
    }{\mm@warning{#1}}
}

```

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.

```
\providecommand{\mathup}[1]{{\operatorfont #1}} % also in kpfonts
\mm@macro{e}{\mathup{e}}
\AtBeginDocument{\let\oldi\i \let\oldj\j
    \renewcommand{\i}{\TextOrMath{\oldi}{\mathup{i}}}
    \renewcommand{\j}{\TextOrMath{\oldj}{\mathup{j}}} }

```

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 LuaLaTeX, 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}`.

```
\newcount\mm@charcode
\newcount\mm@charclass
\newcount\mm@charfam
\newcount\mm@charslot

\newcommand*\math@family[2]{%
    \mm@charfam=#2
    \ifluatex
        \mm@charclass=\Umathcharclass`#1
        %\mm@charfam=\Umathcharfam`#1
        \mm@charslot=\Umathcharslot`#1
        \Umathcode`#1= \mm@charclass \mm@charfam \mm@charslot
    \else
        \mm@charcode=\mathcode`#1
        % extract charclass
        \@tempcnta=\mm@charcode
        \divide\@tempcnta by "1000
        \multiply\@tempcnta by "1000 % charclass
        \mm@charclass=\@tempcnta
        % extract charslot
        \@tempcnta=\mm@charcode
        \@tempcntb=\mm@charcode
        \divide\@tempcnta by "100
        \multiply\@tempcnta by "100 % charclass + charfam
        \advance\@tempcntb by -\@tempcnta % charslot
        \mm@charslot=\@tempcntb
        % construct charcode
        \mm@charcode=\mm@charclass
        \multiply\mm@charfam by "100
        \advance\mm@charcode by \mm@charfam
        \advance\mm@charcode by \mm@charslot
        \mathcode`#1=\mm@charcode
    \fi
}

\newcommand*\MathFamily[2]{%
    \ifx\@onlypreamble\@notprerr
        \math@family{#1}{#2}
    \else % before \begin{document}
        \AtBeginDocument{\math@family{#1}{#2}}
    \fi
}

\newcommand*\MathUp[1]{\MathFamily{#1}{0}}
\newcommand*\MathIt[1]{\MathFamily{#1}{1}}

```

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.

```
\newcommand*\MathNumbers[1]{%
    \in@{e}{#1} \ifin@ \MathUp{e} \fi
    \in@{i}{#1} \ifin@ \MathUp{i} \fi
    \in@{j}{#1} \ifin@ \MathUp{j} \fi
}

\newcommand*\MathProba[1]{%
```

```
    \in@{P}{#1} \ifin@ \MathUp{P} \fi
    \in@{E}{#1} \ifin@ \MathUp{E} \fi
    \in@{V}{#1} \ifin@ \MathUp{V} \fi
}

```

`\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'.

```
\def\apply#1#2{\apply@#1#2,\apply@,}
\def\apply@#1#2,{\ifx\apply@#2\empty
    \else #1{#2}\afterfi@{\apply@#1}\fi}
\def\afterfi@#1#2\fi{\fi#1}

\newcommand*\MathNormal[1]{% list argument
    \apply\MathIt{#1}
}

```

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.

```
\newcommand{\enumber}{%
    \PackageWarning{mismath}{Command \string\enumber\space
        is deprecated, \MessageBreak
        use \bslash MathUp{e} instead}
    \MathUp{e}
}
\newcommand{\inumber}{%
    \PackageWarning{mismath}{Command \string\inumber\space
        is deprecated, \MessageBreak
        use \bslash MathUp{i} instead}
    \MathUp{i}
}
\newcommand{\jnumber}{
    \PackageWarning{mismath}{Command \string\jnumber\space
        is deprecated, \MessageBreak
        use \bslash MathUp{j} instead}
    \MathUp{j}
}
\newcommand{\PEupright}{
    \PackageWarning{mismath}{Command \string\PEupright\space
        is deprecated, \MessageBreak
        use \bslash MathUp{P} and \bslash MathUp{R} instead}
    \MathUp{P}\MathUp{E}
}

```

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.

```
\newcommand*\pinumber[1][]{
    \@ifundefined{itpi}{\let\itpi\pi}{}
    \ifthenelse{\equal{#1}{}}{
      \ifx\@onlypreamble\@notprerr
        \@ifundefined{savedpi}{
            \PackageWarning{mismath}{%
                \bslash pinumber without argument\MessageBreak
                must be used in the preamble first\MessageBreak
                to load LGR fontenc for upright pi}
        }{\let\pi\savedpi}
      \else % in the preamble
        \RequirePackage[LGR,T1]{fontenc}
        \DeclareSymbolFont{UpGr}{LGR}{lmr}{m}{n}
        \let\pi\relax
        \DeclareMathSymbol{\pi}\mathalpha{UpGr}{"70}
        \let\savedpi\pi
      \fi
    }{
        \@ifundefined{#1}{
            \PackageWarning{mismath}{%
                Value #1 must be a valid
                command name\MessageBreak for pinumber,
                but command \bslash #1\space
                is undefined.\MessageBreak
                Perhaps a missing package}
        }{\renewcommand{\pi}{%
            \csname #1\endcsname}
        }
    }
}

\newcommand{\pinormal}{\@ifundefined{itpi}{}{\let\pi\itpi}}

```

And now the commands for vectors (and tensors).

```
\newboolean{arrowvect}
\setboolean{arrowvect}{true}
\newcommand{\arrowvect}{\setboolean{arrowvect}{true}}
\newcommand{\boldvect}{\setboolean{arrowvect}{false}}
```

```
\newcommand{\boldvectcommand}{\boldsymbol} % from amsbsy package
\mm@macro{vect}{\ifthenelse{\boolean{arrowvect}}{
        \vv}{\boldvectcommand}} % doesn’t work well with \if... \fi
\newcommand*{\hvect}[1]{\vect{\vphantom{t}#1}}
\newcommand*{\hvec}[1]{\vec{\vphantom{t}#1}}

\newcommand*{\@norm}[1]{
    \mbox{\raisebox{1.75pt}{\small$\bigl\Vert$}} #1
    \mbox{\raisebox{1.75pt}{\small$\bigr\Vert$}} }
% works better than with relative length
\newcommand*{\@@norm}[1]{
    \mbox{\footnotesize\raisebox{1pt}{$\Vert$}} #1
    \mbox{\footnotesize\raisebox{1pt}{$\Vert$}} }
\newcommand*{\@@@norm}[1]{
    \mbox{\tiny\raisebox{1pt}{$\Vert$}} #1
    \mbox{\tiny\raisebox{1pt}{$\Vert$}} }
\@ifundefined{norm}{\providecommand*{\norm}[1]{
        \mathchoice{\@norm{#1}}{\@norm{#1}}{\@@norm{#1}}{\@@@norm{#1}}
        }
    }{\mm@warning{norm} } % bad result with libertinust1math

\DeclareMathAlphabet{\mathbfsfit}{\encodingdefault}{\sfdefault}{bx}{it}
\newcommand{\tensor}{\mathbfsfit} % isomath uses \mathsfbfit

```

And now all the other commands.

```
\mm@macro{di}{\mathop{}\!\mathup{d}}
\newcommand\probastyle{}
\let\Par\P % end of paragraph symbol
\renewcommand{\P}{\operatorname{\probastyle{P}}}
\mm@macro{E}{\operatorname{\probastyle{E}}}
\mm@macro{V}{\operatorname{\probastyle{V}}}

\mm@operator{\adj}{adj}
\mm@operator{\Aut}{Aut}
\mm@operator{\codim}{codim}
\mm@operator{\Conv}{Conv}
\mm@operator{\cov}{cov}
\mm@operator{\Cov}{Cov}
\mm@macro{curl}{\operatorname{\vect{\mathup{curl}}}}
\mm@operator[divg]{\divg}{div}
\mm@operator{\End}{End}

\mm@operator{\erf}{erf}
\mm@macro{grad}{\operatorname{\vect{\mathup{grad}}}}
\mm@operator{\id}{id} % mathop or mathord ?
\mm@operator{\Id}{Id}
\mm@operator{\im}{im}
\let\oldIm\Im \renewcommand{\Im}{\operatorname{Im}}
\mm@operator{\lb}{lb}
```

```
\mm@operator{\lcm}{lcm}

\mm@operator{\rank}{rank}
\let\oldRe\Re \renewcommand{\Re}{\operatorname{Re}}
\mm@macro{rot}{\operatorname{\vect{\mathup{rot}}}}
\mm@operator{\sgn}{sgn}
\mm@operator{\sinc}{sinc}
\mm@operator[spa]{\spa}{span}
\mm@operator{\tr}{tr}
\mm@operator{\var}{var}
\mm@operator{\Var}{Var}
\mm@operator[Zu]{\Zu}{Z}

\mm@operator{\arccot}{arccot}
\mm@operator{\sech}{sech}
\mm@operator{\csch}{csch}
\mm@operator{\arsinh}{arsinh}
\mm@operator{\arcosh}{arcosh}
\mm@operator{\artanh}{artanh}
\mm@operator{\arcoth}{arcoth}
\mm@operator{\arsech}{arsech}
\mm@operator{\arcsch}{arcsch}

\mm@operator[bigO]{\bigO}{\mathcal{O}}
\mm@operator[bigo]{\bigo}{O}
\mm@operator[lito]{\lito}{o}

\mm@macro{mathset}{\mathbf}
\mm@macro{R}{\mathset{R}}
\@ifpackagewith{babel}{russian}{
    \PackageWarningNoLine{mismath}{Option russian of babel
    is loaded\MessageBreak
    Command \bslash C will not be redefined}
    }{\mm@macro{C}{\mathset{C}}}
\mm@macro{N}{\mathset{N}}
\mm@macro{Z}{\mathset{Z}}
\mm@macro{Q}{\mathset{Q}}
\mm@macro{F}{\mathset{F}}
\mm@macro{K}{\mathset{K}}

\mm@macro{ds}{\displaystyle}
\mm@macro{dlim}{\lim\limits}
\mm@macro{dsum}{\sum\limits}
\mm@macro{dprod}{\prod\limits}
\mm@macro{dcup}{\bigcup\limits}
\mm@macro{dcap}{\bigcap\limits}

\mm@macro{lbar}{\overline}
\@ifundefined{hlbar}{
    \providecommand*{\hlbar}[1]{\overline{\vphantom{t}#1}}}{
```

```
    \mm@warning{hlbar} }
\newcommand\@eqdef{\stackrel{\mathup{def}}{=}}
\newcommand\@@eqdef{\stackrel{\mathrm{\Delta}}{=}}
\mm@macro{eqdef}{\@ifstar{\@@eqdef}{\@eqdef}}
\mm@macro{unbr}{\underbrace}
\mm@macro{iif}{if and only if\xspace}

```

We used `\mathurm` 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.

```
\mm@macro{then}{\ \Longrightarrow \ \mbox{} }
\@ifundefined{txt}{
    \providecommand*{\txt}[1]{\quad\text{#1}\quad} }{
    \mm@warning{txt} }
\mm@macro{mul}{\mathord{\times}}
\@ifundefined{pow}{
    \providecommand*{\pow}[2]{\left( #1 \right)^{\!#2}} }{
    \mm@warning{pow} }
\@ifundefined{abs}{
    \providecommand*{\abs}[1]{\left\vert#1\right\vert} }{
    \mm@warning{abs} }
\@ifundefined{lfrac}{
    \providecommand*{\lfrac}[2]{\frac{\;#1\;}{\;#2\;}} }{
    \mm@warning{lfrac} }

\newcommand{\systemstretch}{1.2}
\newcommand{\systemsep}{\medspace}
\newenvironment{system}[1][l]{
    \renewcommand{\arraystretch}{\systemstretch}
    \setlength{\arraycolsep}{0.15em}
    \left\{\begin{array}{@{\systemsep}#1@{}} %
}{\end{array}\right.}

\newenvironment{spmatrix}{
    \left(\begin{smallmatrix}
}{\end{smallmatrix}\right)}

\newenvironment{mathcols}{% needs multicol package
    \renewcommand{\columnseprule}{0.1pt}
    \begin{multicols}{2}
        \par\noindent\hfill
        \begin{math}\begin{aligned}\displaystyle
}{%
        \end{aligned}\end{math} \hfill\mbox{}
    \end{multicols}
}
\newcommand{\changecol}{%
    \end{aligned}\end{math} \hfill\mbox{}
    \par\noindent\hfill
    \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 2ε*, Eddie Saudrais, CTAN, v1.3 2013/07/11.

[6] *amsmath – AMS mathmatical 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 2ε*, 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 2ε*, 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-GUT*enberg*, 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 LATEX 2$_\varepsilon$*, 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.