Miscellaneous mathematical macros
The mismath package[*]

Antoine Missier
antoine.missier@ac-toulouse.fr

May 04, 2024

Contents

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According to the International Standards ISO 31-0:1992 to ISO 31-13:1992 (superseded by ISO 80000-2:2009), mathematical *constants* e, i, π should be typeset in roman (upright shape) and not in italic like variables (see [1] [2] [3] [4]).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

Even though it is recommended to typeset vectors names in bold italic style [2] [4], they are often represented with arrows, especially in school documents or in physics. To draw nice arrows above vectors, we use the esvect package by Eddie Saudrais [5]. Additionally we provide a few more macros related to vectors with arrows, particularly to improve the typesetting of the norm: $\|\overrightarrow{AB}\|$ instead of the LaTeX version $\|\overrightarrow{AB}\|$, which is not vertically adjusted, or worse $\left\|\overrightarrow{AB}\right\|$ (when using `\left...\right`).

[*] This document corresponds to mismath v3.0, dated 2024/05/04. Thanks to François Bastouil for initial help in English translation.

The package also offers other macros to typeset:

- tensors in sans serif bold italic shape (ISO recommendation [1] [2] [3]),
- some standard operator names,
- several commands with useful aliases,
- improved spacings in mathematical formulas,
- systems of equations and small matrices,
- displaymath in double columns for lengthy calculations with short expressions.

To avoid compatibility issues, most of our macros will only be defined if there isn't already a command with the same name in the packages loaded before mismath. If a macro is already defined, a warning message will be displayed and the mismath definition will be ignored. If you wish to keep the mismath or the existing command, you can use `\let\`⟨*command*⟩`\relax`, before loading mismath, or after.

`[`⟨*options*⟩`]` The mismath package loads the mathtools[1] package by Morten Høgholm and Lars Madsen [6] which in turn loads the amsmath package [7]. If you want to use amsmath or mathtools with specific options, you can include these options as options of mismath, or you can load amsmath or mathtools with the desired options before loading mismath. When using the package unicode-math [8], mismath should be loaded before unicode-math, just like amsmath.

An ISO recommendation, although rarely followed, is to typeset uppercase Greek letters in italic shape, as for other variables [4]. This is automatically achieved, for some particular fonts, with packages such as fixmath by Walter Schmidt [9], isomath by Günter Milde [10] or pm-isomath by Claudio Beccari [11] and optionally with many others (such as mathpazo or mathptmx with the option `slantedGreek`). When running through LuaLaTeX or XeLaTeX you can also get this result with the option `math-style=ISO` provided by the unicode-math package. We also have the new mathgreeks package [12] which offers a wide range of fonts and different settings with Greek letters. However this feature is not implemented here due to a conflicting rule in France, where all capital letters in mathematics are required to be typeset in upright shape[2]. The user is free to choose loading one of these packages or not.

2 Usage

2.1 Mathematical constants

`\mathup` As for classic functions identifiers, *predefined* mathematical constants should be typeset in upright shape (typically in roman family), but this practice is not sufficiently respected, probably because it's a bit tedious. A first solution is to use the `\mathup`

[1]The mathtools package offers numerous helpful macros and improvements of the amsmath package.
[2]The frenchmath package [32] ensures to follow the recommended French rules.

macro, which is preferable to `\mathrm`[3], for setting any group of letters in roman. For example you can use `\mathup{e}` to get the Euler's number.

`\e` `\i` `\j` To avoid cluttering a document that contains many occurrences of Euler's number e, or imaginary number i, with `\mathup{e}` or `\mathup{i}`, the package provides the `\e` command for Euler's number and `\i` or `\j` for imaginary numbers. Let us notice that `\i` and `\j` already exist in LaTeX. In LR (left-to-right) mode, they produce 'ı, ȷ' without the dot, allowing you to place accents on them. However, in mathematical mode, they produce the warning "`LaTeX Warning: Command \i invalid in math mode on input line` ⟨*line*⟩". With the new definition provided by the package, `\i` and `\j` will be redefined specifically for mathematical mode.

`\MathUp` Indeed, typing a lot of backslashes for constants like e, i, or j in a document with numerous formulas can become tiresome. To alleviate this, the package proposes another solution with the macro `\MathUp{`⟨*char*⟩`}`. For example, when `\MathUp{e}` is called, any subsequent occurrence of e will automatically be set in roman (upright shape), without the need to type `\e` explicitly. The effect of this macro can be either global or local, depending on whether it is used outside or inside an environment or braces. Furthermore, you can also call this macro in the preamble, then the change will apply from the beginning of the document. This powerful command allows you to bring a document up to the standards effortlessly and without changing anything in your mathematical formulas. In fact, `\MathUp` can be applied to any valid single character, offering flexibility for various use cases[4].

`\MathIt` When there are other occurrences of e, i or j as variables, you can still obtain italicized e, i or j using LaTeX commands `\mathit` or `\mathnormal`, which are useful for a single use. However, you also have the option to use the inverse switch `\MathIt{`⟨*char*⟩`}`, which has a global effect when used outside environments or braces, or a local effect when used inside them. Similar to `\MathUp`, `\MathIt` can be applied to any single character.

`\MathNumbers` `\MathNormal` These macros enable you to set upright or normal (italic) typesetting for multiple letters in a single command. For instance, `\MathNumbers{e,i}` is equivalent to `\MathUp{e}\MathUp{i}`. In `\MathNumbers`, the comma separator between letters can be modified or removed as needed. In fact, this macro only affects the letters e, i, or j; it has no effect on other characters. On the other hand, `\MathNormal` accepts any comma-separated list of arguments. This means you can apply the normal italic math mode typesetting to various letters at once using `\MathNormal`.

`\enumber` `\inumber` `\jnumber` These three commands, used until version 2.2 but only functioning within the preamble, serve now as aliases for the commands `\MathUp{e}`, `\MathUp{i}` or `\MathUp{j}`, so they can be used anywhere in the document or preamble and has an inverse switches with `\MathIt`.

[3]The `\mathup` macro is based on `\operatorfont`, which comes from the amsopn package, automatically loaded by amsmath. In beamer, the default math font is sans serif, but `\mathrm` produces a font with serifs, which might not match the overall style of the presentation. Hence, using `\mathup` is indeed a better choice in beamer presentations to ensure that mathematical constants are typeset in upright shape and consistent with the default sans serif math font.

[4]Another use of it with probability will be presented in section 2.3.

`\pinumber[⟨option⟩]` The constant π should also be typeset in upright shape (see [1], [2], [4]), which is different from italicized π. However, this recommendation is even less commonly followed compared to the one concerning e and i [1]. Thanks to the `\pinumber` command, the italic π will be replaced with an upright π each time `\pi` is called. Thus `\pinumber` makes your document compliant with standards without changing the source code of your mathematical formulas. It functions in two different ways.

1. You can load a Greek letters package that provides the glyphs in upright shape. There are many available. Notably, let us mention upgreek [13], mathdesign [14], kpfonts [15], fourier [16] (used in the present document), but also pxgreeks (using pxfonts [17]), txgreeks (using txfonts [18])[5], libertinust1math [19], libgreek, etc. A special mention goes to lgrmath of Jean-François Burnol [20] which allows the use of any Greek LGR-encoded font in math mode, an idea taken up in mathgreeks [12]. Also note newtxmath [21] which has several font options. These packages provide commands like `\uppi` (upgreek), `\piup` (mathdesign, kpfonts, lgrmath), `\otherpi` (fourier), etc.[6]

   In this case, `\pinumber` must be called in the preamble with an optional argument being the name of the command, *without the backslash*, giving access to the upright pi: piup, uppi, otherpi... However, installing such a Greek letters package will modify all the other Greek letter glyphs.

   By calling in the preamble `\MathNumbers{ei}\pinumber[otherpi]` (assuming the fourier package is loaded), you can achieve the following result:

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

2. Without installing a package, it is possible to change only the glyph of pi without altering the other Greek letters, which are typically in italics.

   In this case, `\pinumber` must be called in the preamble with an optional argument of the `key=value` type. The key name corresponds to a package providing the same glyph. The following table summarizes the available options. When a key is given without a value, `\pinumber` will choose a default value specified in the following text (depending on the key).

Option `lgrmath=...`	Result
`Alegreya-LF`	π
`Cochineal-LF`	π
`LibertinusSerif-LF`	π
`LibertinusSans-LF`	π
`lmr`	π
`lmss`	π
`gentium`	π
`lato-LF`	π

Other options	Result
`fontspec=...`	...
`upgreek=Euler`	π
`upgreek=Symbol`	π
`mathdesign`	π
`kpfonts`	π
`fourier`	π
`pxfonts`	π
`txfonts`	π

[5] When using pxgreeks or txgreeks, they should be loaded *after* mismath to avoid an error due to conflict with the existing macros `\iint`, `\iiint`, `\iiiint`, `\idotsint` in amsmath.

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

- With the `lgrmath` key, we actually have numerous possibilities for values (any Greek letters math font in LGR encoding). The documentation of the lgrmath package explains how to check an visualize all available fonts on your distribution. We have only presented seven of them. The default value is `lmr`. Other interesting values are `NotoSerif-LF`, `Clara-TLF`, `droidserif`, `fct`, `llcmss`.
- When `\pinumber` is called without an argument in the preamble, it corresponds to the option `lgrmath=lmr`. This π character is well-suited for use with the Latin Modern font family[7].
- With the `fontspec` key, there are also many possible values, corresponding to the TrueType or OpenType fonts installed on your system (works with LuaLaTeX or XeLaTeX). See the mathgreeks documentation for examples.
- With the `upgreek` key, the default value is `Symbol`. There is a third possible value, `Symbolsmallscale`, which provides the same character as `Symbol` but reduced in size by 10 %.
- With the `mathdesign` key, there are actually 3 possible values: `Utopia`, `Garamond` or `Charter` (the default value), but the glyphs obtained for pi look quite similar.
- With the `kpfonts` key, we have two possible values: `normal` (default) and `light`. The option `kpfonts=light` provides a slightly less bold character.
- The last keys, `fourier` (based on Utopia), `pxfonts` (based on Palatino), `txfonts` (based on Times) are booleans whose default value is `true` (when called). The `txfonts` option yields the same glyph than `lgrmath=txr`.

The unicode-math package [8] provides `\uppi`, and you can use `\pinumber[uppi]` to produce automatic upright pi in the selected math font, but `\pinumber[uppi]` must be called *after* unicode-math, and a math font must have been explicitly chosen with `\setmathfont`. You can also use the `fontspec` key option to obtain pi in any font that is supported by unicode-math e.g. `\pinumber[fontspec=STIX Two Math]`.

For other fonts, it can be quite complicated to make Greek letters packages work with unicode-math. In any case, such a package must be loaded after unicode-math and in `\AtBeginDocument`. However, `\pinumber` supports unicode-math very well with any previous `key=value` option, by calling `\pinumber[`⟨*option*⟩`]` after unicode-math.

`\itpi` When you activate `\pinumber`, the original italic π is still accessible using `\itpi`.

`\pinormal` In fact, `\pinumber` is a toggle, with its inverse toggle being `\pinormal`. The latter restores the `\pi` command to its default behavior. Thus, `\pinumber` can be used anywhere in the document (like `\pinormal`), but then without arguments and provided it has been initially called in the preamble, according to the procedures outlined above.

2.2 Vectors (and tensors)

`\vect` By default, the `\vect` command[8], produces vectors with arrows (thanks to the esvect

[7] It will look the same as the one provided by `lgrmath=cmr` or by Günter Milde's textalpha package [22].

package by Eddie Saudrais[9]) which are more elegant than those produced by LaTeX's `\overrightarrow` command. The esvect package has an optional argument (a single letter between `a` and `h`) to define the desired type of arrow (see [5]). In mismath, esvect is loaded with the option `b`: `\vect{AB}` gives $\overrightarrow{AB}$. If you wish to use a different type of arrow, you must call esvect with the appropriate option *before* loading mismath. For example, using `\usepackage[d]{esvect}` will provide the same arrows that are used by default in [5].

`\boldvect` The `\vect` macro allows vector names to be typeset using bold italic font, as recommended by ISO [2] [3], instead of using arrows. By using the `\boldvect` command, you can modify the behavior of `\vect` locally or globally, depending on its placement in the document (inside or outside a group or an environment):

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

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

`\boldvectcommand` By default `\boldvect` uses the `\boldsymbol` command[10] from the amsbsy package, which is automatically loaded by amsmath. However, you may prefer other packages that produce bold italic fonts, such as fixmath with the `\mathbold` command, isomath with `\mathbfit` or bm with the `\bm` command; unicode-math provides the `\symbfit` command. To use an alternative command instead of `\boldsymbol` in mismath, redefine `\boldvectcommand`, for instance after loading fixmath:

```
\renewcommand\boldvectcommand{\mathbold}
```

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

`\arrowvect` At any moment, you can revert to the default behavior using the inverse switch `\arrowvect`. These switches can be placed anywhere, whether inside mathematical mode or within an environment (with a local effect) or outside (with a global effect).

`\hvect` When vectors with arrows are typeset side by side, the arrows can be set up slightly higher using `\hvect` (which places a vertical phantom box containing 'A') to avoid inelegant effects. For example, writing

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

This adjustment ensures a nicer appearance when vectors with arrows are combined in an equation[11]. The `\boldvect` and `\arrowvect` switches have the same effect on `\hvect` as they do on `\vect`.

`\hvec` In a similar way, `\hvec` raises the little arrow produced by the LaTeX command `\vec`, to the height of the letter 'A' (but `\boldvect` have no effect on `\vec` nor `\hvec`):

[8] The definition of most macros in this package, will only take effect if the macro has not been previously defined by another package. This ensures compatibility and avoids conflicts when using the mismath package with other LaTeX packages.

[9] esvect provides the `\vv` macro used by `\vect`.

[10] `\mathbf` produces upright bold font, even when used in combination with `\mathit`.

[11] For a fine tuning you can also use the `\vstrut` or `\cstrut` macros from the spacingtricks package [23].

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

`\norm` The norm of a vector is conventionally represented using the delimiters `\lVert` and `\rVert` (or `\|` unless a plus (+) or minus (-) sign follows the opening delimiter) or `\left\Vert` and `\right\Vert` for adaptive delimiters. Unfortunately, these delimiters are always vertically centered, relatively to the mathematical center line, whereas vectors with arrows are asymmetric objects. The code `$\norm{\vec{h}}$` raises a smaller double bar to produce $\|\vec{h}\|$ instead of $\|\vec{h}\|$ or $\left\|\vec{h}\right\|$. 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. $X^{\|\vec{h}\|}$. This macro is useful only for arguments of special height, such as $\vec{h}$ or $\overrightarrow{AB}$ and may give bad results in other situations.

`\mathbfsfit` `\tensor` For tensors symbols, ISO rules recommend using sans serif bold italic, but there is no such math alphabet in the default LaTeX mathematical style. However, the mismath package defines this alphabet (assuming the font encoding and package you use permits it) and provides the macro `\mathbfsfit` or its alias `\tensor`. By writing `\tensor{S}\otimes\tensor{T}`, you get $\boldsymbol{\mathsf{S}} \otimes \boldsymbol{\mathsf{T}}$.

2.3 Standard operator names

`\di` The *differential* operator should be typeset in upright shape, not in italics, to distinguish it from variables (as mentioned in [1] [2] [4] [33]). To achieve this, we provide the `\di` command. Take a look at the following examples (notice the thin spaces before the d, just like with classic function's names):

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

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

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

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

The command `\di` can also represent the *distance*, hence its name:

$$\mathrm{d}(u, \mathcal{H}) = \frac{|\langle u, v\rangle|}{\|v\|}.$$

`\P` `\E` To refer to probability[12] and expectation the proper use is to typeset the capital letters P, E in roman just like any standard function identifier. This can be achieved with `\P` and `\E` commands.

`\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 the following table), but some authors prefer to use V, which can be produced using `\V`.

[12] LaTeX provides also `\Pr` which gives Pr.

`\MathProba` `\MathNormal` As for e, i or j, you can use `\MathUp{P}`, `\MathUp{E}` or `\MathUp{V}` to avoid typing many `\P`, `\E` or `\V`. However you can also achieve this in a single command with `\MathProba`, for example `\MathProba{PE}`. We get the inverse toggle with `\MathIt` for any individual letter or `\MathNormal` for a list.

`\probastyle` Some authors use “outline” font shape 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}`[13] brings the double-struck letters. The `\mathbb` command is provided by amsfonts package (which needs to be called in the preamble), but also by other complete math font packages such as mathdesign, kpfonts, fourier, unicode-math…

The following standard operator names are defined in mismath:

`\adj`	adj	`\End`	End	`\Res`	Res
`\Aut`	Aut	`\erf`	erf	`\rot`	$\overrightarrow{\mathrm{rot}}$
`\codim`	codim	`\grad`	$\overrightarrow{\mathrm{grad}}$	`\sgn`	sgn
`\coker`	coker	`\id`	id	`\sinc`	sinc
`\Conv`	Conv	`\Id`	Id	`\spa`	span
`\Cov`	Cov	`\im`	im	`\tr`	tr
`\cov`	cov	`\lb`	lb	`\var`	var
`\curl`	$\overrightarrow{\mathrm{curl}}$	`\lcm`	lcm	`\Var`	Var
`\divg`	div	`\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 both are very common. Please note that `\div` already exists (÷) and `\span` is a TeX primitive; they haven't been redefined. Therefore the provided macros are called `\divg` (divergence) and `\spa` (span of a set of vectors). Furthermore `\Z` is used to denote the set of integers (see 2.4), which is why we propose `\Zu`, to designate the center of a group: $\mathrm{Z}(G)$ (from German Zentrum).

The mismath package also provides some (inverse) circular or hyperbolic functions, that are missing in LaTeX:

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

`[nofunction]` Some may find that the definition of all these operators and functions is not useful for their needs. So, the above definitions (on this page) can be disabled with the `nofunction` option.

`\Re` `\Im` The `\Re` and `\Im` macros refer to real and imaginary part of a complex number. They have been redefined to produce Re and Im, in place of outdated symbols $\Re$ and $\Im$. Nevertheless, it is still possible to obtain the old symbols with `\oldRe` and `\oldIm`.

`[otherReIm]` `[classicReIm]` An alternative notation $\mathcal{R}e$, $\mathcal{I}m$ is provided by invoking the `otherReIm` package option, whereas the `classicReIm` option deactivates these redefinitions.

`\bigO` `\bigo` `\lito` Asymptotic comparison operators (in Bachmann-Landau notation) are obtained with `\bigO` or `\bigo` and `\lito` commands. The first one uses the `\cmmathcal` alphabet and the last two compose the letters 'O' and 'o' in roman, as for any operator:

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

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**, , **Z**, **N**, **Q**, whereas double-struck letters ($\mathbb{R}, \mathbb{C}, \mathbb{Z}, \mathbb{N}, \mathbb{Q}$) are reserved for writing at the blackboard [33]. Similarly, to designate a field we use **F** or **K** (Körper in German). We obtain these symbols with the following macros:

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

`\mathset` The `\mathset` command enables you to change the behavior of all these macros in a global way. By default, `\mathset` is an alias for `\mathbf`, but if you prefer outline letters, you can simply use `\renewcommand\mathset{\mathbb}` (with local effect when inside an environment or a pair of curly braces).

`\onlymathC` The macro `\onlymathC` is designed for cases when `\C` is already defined, but only in text mode (usually with the Russian language). Then you get the message: "`Command \C invalid in math mode`". This macro preserves the original definition for text mode and allows you to use `\C` for the complex number set in math mode. Simply call `\onlymathC` once in the preamble or anywhere in the document.

`\ds` The `\displaystyle` command is very common, so 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 placed under or above the symbol whereas for in-line math mode, they are placed on the right, as a 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 can be forced with `\displaystyle` or its alias `\ds`. However, when using these commands, all the rest of the current mathematical environment will be set in displaymath mode (as shown in the previous example, where the fraction will be expanded). To limit the display style effect to the affected symbol only, similar the the amsmath command `\dfrac`, we can use the following macros: `\dlim`, `\dsum`, `\dprod`, `\dcup`, `\dcap`. So

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

\lbar
\hlbar

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

$$\overline{z+z'} = \overline{z} + \overline{z'}, \text{ obtained with } \texttt{\textbackslash hlbar\{z\}}, \text{ looks better than } \overline{z+z'} = \bar{z} + \overline{z'}.$$

\eqdef
\eqdef*

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

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

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

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

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

\unbr

`\unbr` is an alias for `\underbrace`[14], 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 just like the standard macro `\iff` does it with $\Longleftrightarrow$. Similarly, the `\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 as addition or subtraction operators, whereas division obtained with / is closer to its operands. This actually hides the priority of multiplication over + and −. That's why we provide the `\mul` macro, behaving like / (ordinary symbol) and leaving less space around than `\times`:

$$\lambda + \alpha{\times}b - \beta{\times}c, \text{ obtained with } \texttt{\textbackslash mul}, \text{ looks better than } \lambda + \alpha \times b - \beta \times c.$$

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 ensure the same amount of space on both sides, you can use thin negative spaces `\!` or enclose the function or structure with braces:

[13] The effect of this redefinition is global or local to the container environment in which it is used.

[14] The mathtools package by Morten Høgholm and Lars Madsen [6] provides a new and improved version of the `\underbrace` command, along with many other useful macros. It is loaded by mismath.

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

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

The negative thin space after the function name is not related to `\mul`, but to the "excessive" spaces around the `\left... \right` structure.

`\pow` In the same way, when typesetting an exponent after a closing *big* parenthesis produced by `\right)`, the exponent appears to be a little to far from the parenthesis. To address this issue, the `\pow{`⟨*expr*⟩`}{`⟨*pow*⟩`}` command is provided[15], which sets the positioning of the exponent ⟨*pow*⟩ slightly closer to the right parenthesis. Compare:

$$\mathrm{e}^a = \lim_{n\to+\infty}\left(1+\frac{a}{n}\right)^{\!n} \quad \text{obtained with \pow, rather than} \quad \mathrm{e}^a = \lim_{n\to+\infty}\left(1+\frac{a}{n}\right)^n.$$

`\abs` The correct typesetting of absolute value (or modular for a complex number) is achieved using `\lvert`...`\rvert`, rather than |, as the latter doesn't maintain proper spacing in some situations (when a sign follows the open delimiter). For bars whose height has to adapt to the content, we can use `\left\vert`...`\right\vert` or, more simply, the `\abs{`...`}` command, which is equivalent[16].

`\lfrac` The `\lfrac` macro behaves like `\frac` but with thicker spaces around the arguments, making the corresponding fraction bar slightly longer. See the following examples:

$$\overline{Z} = \frac{\;\overline{z_1 - z_2}\;}{\;\overline{z_1 + z_2}\;} \qquad u(x) = \frac{\;\frac{1-2x}{5}\;}{\;x^2+1\;} \qquad y' + xy = \frac{\,1\,}{\,\sqrt{x}\,}$$

This macro has an optional parameter `\lfrac[`⟨*space*⟩`]{`⟨*num*⟩`}{`⟨*denom*⟩`}` to adjust the length of the fraction bar. The optional ⟨*space*⟩ argument must be given in *math units* (`mu`); the default value is `7mu` (equivalent to `\:\,`), `4mu` in the last example.

`[ibrackets]` Open intervals are commonly represented with parenthesis, e.g. $(0,+\infty)$, but sometimes square brackets are used, especially in French mathematics: $]0,+\infty[$. In that specific case, the space around the square brackets is often inappropriate, as in the expression $x\in]0,+\infty[$. To address this issue, we have redefined the brackets in the ibrackets package [24]. This one can be optionally loaded by mismath using the `ibrackets` package option. Thus `$x\in ]-\pi,0[ \cup ]2\pi,3\pi[$`

yields $x \in \mathopen]-\pi,0\mathclose[\cup\mathopen]2\pi,3\pi\mathclose[$ with ibrackets,
instead of $x \in]-\pi,0[\cup]2\pi,3\pi[$ without ibrackets.

In our code, the symbols [and] are set as 'active' characters, behaving like ordinary characters and not as delimiters in most cases. Therefore, a line break could occur between the two brackets, but it is always possible to transform them into delimiters using `\left` and `\right`.

However, when a bracket is *immediately* followed by a + or - character, it becomes an open delimiter. Therefore, when the left bound contains an operator

[15] This macro gives bad results with *normal-sized* parenthesis.
[16] We could also define `\abs` using `\DeclarePairedDelimiter` from the mathtools package [6].

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. For example if you write `$x \in ] -\infty, 0]$`, it yields $x \in]-\infty, 0]$ instead of $x \in]{-\infty}, 0]$. Conversely, when dealing with algebraic expressions involving intervals, *you must leave a blank space between the second bracket and the +/- operation*. For instance `$[a,b] +[c,d]$` yields $[a,b]+[c,d]$ but `$[a,b]+ [c,d]$` yields $[a,b]{+}[c,d]$.

Note that there are other ways to proceed, for example with `\interval`, from the interval package [25], or with `\DeclarePairedDelimiter`[17] from mathtools [6].

`[decimalcomma]` In many countries, except notably in English-speaking countries, the comma is used as a decimal separator for numbers. However, in the math mode of LaTeX, the comma is always, by default, treated as a punctuation symbol and therefore is followed by a space. This is appropriate in intervals: `$[a,b]$` results in $[a,b]$, but is not appropriate for numbers where the comma represents the decimal separator. For example, `$12,5$` is displayed as $12, 5$ instead of 12,5.

Two very convenient packages allow handling the decimal comma in math mode: icomma by Walter Schmidt [26] and ncccomma by Alexander I. Rozhenko [27]. The second package takes a more generic approach, however it poses several compatibility issues, in particular when running through LuaLaTeX, using unicode-math and calling `\setmathfont`. Therefore we propose the decimalcomma package [28], functionally identical to that of ncccomma but without the aforementioned incompatibility. It can be loaded by mismath using the `decimalcomma` package option.

2.6 Environments for systems of equations and small matrices

`system` The `system` environment, defined in the mismath package, is used to represent a system of equations:

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

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

`\systemsep` This first example could also have been achieved using the `cases` environment from the amsmath package, although `cases` places mathematical expressions closer to the bracket. The `\systemsep` length allows you to adjust the gap between the bracket and the expressions. By default, the gap is set to `\medspace`. You can reduce this gap by redefining the command, e.g.: `\renewcommand{\systemsep}{\thinspace}`. Alternatively you can increase the gap using `\thickspace` and the same spacing as of the `cases` environment is obtained with `\renewcommand\systemsep}{}`. The `\systemsep` command allows for greater flexibility in adjusting the spacing within the `system` environment.

`system[⟨coldef⟩]` By default, a system is written like an `array` environment with only one column, left aligned. However the `system` environment has an optional argument that allows to create systems with multiple columns, specifying their alignment using the same

[17]You cannot use `\DeclarePairedDelimiter` with square brackets when ibrackets is loaded.

syntax as the `array` environment in LaTeX. For instance, using `\begin{system}[cl]` will produce a two-column system, with the first column centered and the second column left-aligned, as shown 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` The default spacing between the lines of a `system` environment has been slightly enlarged compared to the one used in `array` environments (using a factor of 1.2). This can be adjusted by using `\renewcommand{\systemstretch}{`⟨*stretch*⟩`}`, where ⟨*stretch*⟩ is the desired value for the spacing. You can place this command inside the current mathematical environment for a local change, or outside for a global change. The default value for is 1.2. Furthermore you can also use the end of the line with a spacing option, as demonstrated above with `\\[1ex]`, to control the spacing between specific lines in the system.

Another example with `\begin{system}[rl@{\quad}l]`[18]:

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

Let's also mention the systeme package [29] which provides a lighter syntax and automatic alignments for linear systems. Additionally, there is the spalign package [30], which offers a convenient and easy syntax for systems and matrices with visually appealing alignments.

`spmatrix` The amsmath package offers several environments to typeset matrices : For example, the `pmatrix` environment surrounds the matrix with parenthesis, and the `smallmatrix` environment creates a smaller matrix suitable for insertion within a text line. We provide a combination of the these both functionalities with the `spmatrix` environment: `$\vec{u}\begin{spmatrix}-1\\2\end{spmatrix}$` yielding $\vec{u}\left(\begin{smallmatrix}-1\\2\end{smallmatrix}\right)$.

The mathtools package enhances the amsmath matrix environments and also provides a small matrix environment with parenthesis: `psmallmatrix`. Moreover, with the starred version `\begin{psmallmatrix*}[`⟨*col*⟩`]`, you can choose the alignment inside the columns (`c`, `l` or `r`). However, the space before the left parenthesis is unfortunately too narrow compared to the space inside the parenthesis. To illustrate this, consider the following comparison: $\vec{u}\left(\begin{smallmatrix}-1\\2\end{smallmatrix}\right)$ (using mismath's `spmatrix`) vs. $\vec{u}\!\left(\begin{smallmatrix}-1\\2\end{smallmatrix}\right)$ (using mathtools `psmallmatrix`).

For typesetting various kinds of matrices, let's mention the excellent nicematrix package by François Pantigny [31].

2.7 Displaymath in double columns

`mathcols` The `mathcols` environment allows you to arrange lengthy calculations with short ex-

[18] `@{...}` sets inter-column space.

pressions across two columns separated by a vertical line, as shown in the following example. However, to use this feature, the multicol package must be loaded in the preamble. The `mathcols` environment activates mathematical mode in display style and uses 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 is used to switch to the next column, and alignments within the columns is done using the classic delimiters `&`, to separate entries, and `\\`, to start a new row.

```
\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 Summary of the package options

The following table summarizes the possible package options. You can add to them any option you want to pass to amsmath or mathtools. The hyperlinks (in blue) redirect to the paragraphs in the documentation where these options are described.

Option	Effect
`nofunction`	don't load some additional function definitions
`otherReIm`	typesets `\Re` and `\Im` as $\mathcal{R}e$ and $\mathcal{I}m$
`classicReIm`	preserves `\Re` and `\Im` as $\Re$ and $\Im$
`ibrackets`	loads the ibrackets package
`decimalcomma`	loads the decimalcomma package

3 Implementation

We load certain packages conditionally to avoid 'option clash' errors in cases where these packages have been previously loaded with other options. The amsmath package is loaded by mathtools.

```
\newif\ifmm@ibrackets % initialized to false
```

```
\DeclareOption{ibrackets}{\mm@ibracketstrue}
\newif\ifmm@decimalcomma
\DeclareOption{decimalcomma}{\mm@decimalcommatrue}
\newif\ifmm@nofunction
\DeclareOption{nofunction}{\mm@nofunctiontrue}
\newif\ifmm@otherReIm
\DeclareOption{otherReIm}{\mm@otherReImtrue}
\newif\ifmm@classicReIm
\DeclareOption{classicReIm}{\mm@classicReImtrue}
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{mathtools}}
\ProcessOptions \relax
%\@ifpackageloaded{amsmath}{}{\RequirePackage{amsmath}}
\@ifpackageloaded{mathtools}{}{\RequirePackage{mathtools}}
\@ifpackageloaded{esvect}{}{\RequirePackage[b]{esvect}}
\RequirePackage{ifthen}
\RequirePackage{xparse} % provides \NewDocumentCommand, now in LaTeX3
\RequirePackage{xspace} % for \iif command
\RequirePackage{iftex}
\RequirePackage{etoolbox} % provides \AtEndPreamble
\RequirePackage{xkeyval} % for \pinumber options

```

The package unicode-math causes some compatibility issues with ibrackets and decimalcomma: theses packages must be loaded *after* unicode-math, but mismath (like amsmath) should be loaded *before* unicode-math. And to complicate matters, unicode-math defines all its commands by `\AtBeginDocument`. Therefore we used the command `\AtEndPreamble`, from the etoolbox package, which makes the job (because both ibrackets and decimalcomma work also in `\AtBeginDocument`).

Moreover the command `\mathbfsfit` (used for tensors) is already defined in unicode-math and will not be redefined if unicode-math is loaded.

```
\@ifpackageloaded{unicode-math}{
    \PackageWarningNoLine{mismath}{The package unicode-math\MessageBreak
        should be loaded after mismath}
}{}
\newif\ifmm@unicodemath
\newif\ifmm@multicol
\AtEndPreamble{% necessary to work with unicode-math
    \ifmm@decimalcomma\RequirePackage{decimalcomma}\fi
    \ifmm@ibrackets\RequirePackage{ibrackets}\fi
    \@ifpackageloaded{multicol}{\mm@multicoltrue}{}
    \@ifpackageloaded{unicode-math}{\mm@unicodemathtrue}{
        \DeclareMathAlphabet{\mathbfsfit}{\encodingdefault}%
            {\sfdefault}{bx}{it}}
}

```

`\bslash` The `\bslash` macro originates from Frank Mittelbach's doc.sty package. It can be employed in other documents as an alternative to `\textbackslash`, especially in situations where `\textbackslash` does not work correctly, such as inside warning messages.

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

```

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

The next three internal macros serve as meta commands for conditionally defining macros while providing a warning message if the macro already exists. These macros can be useful in other packages as well.

```
\newcommand\mm@warning[1]{
    \PackageWarningNoLine{mismath}{Command \bslash #1 already exist
        \MessageBreak 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 rather `\mathup` (based on `\operatorfont` from the amsopn package). This command also works fine with other sans serif fonts like cmbright.

Moreover for beamer, which changes the default font family (to 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.

```
\@ifundefined{mathup}{
    \providecommand*{\mathup}[1]{{\operatorfont #1}}
    }{\mm@warning{mathup} } % also in kpfonts and unicode-math
\mm@macro{e}{\mathup{e}}
\AtBeginDocument{\let\oldi\i \let\oldj\j
    \renewcommand{\i}{\TextOrMath{\oldi}{\mathup{i}}}
    \renewcommand{\j}{\TextOrMath{\oldj}{\mathup{j}}} }

```

\MathFamily

The following macros `\MathUp` and `\MathIt` are switches that transform any chosen letter in math mode to roman or italic style. These switches can be used anywhere in the document or preamble. They are based on the generic macro `\MathFamily`. To obtain a letter in roman style instead of italic, we need to change the mathcode digit that represents the font family: 1 to 0.

For example, except for LuaLaTeX, mathcode of the 'e' letter is: 'e="7165 (decimal 29029), with the second digit '1' indicating "italic" style. To get a roman 'e', we need to change its mathcode to "7065.

When used in the preamble, we call `\MathFamily` by `\AtBeginDocument` for working with the beamer package. Let's notice that `\MathFamily` has an erratic behavior when unicode-math is loaded, but fortunately, in that case, the `\DeclareMathSymbol` can be used instead, even outside the preamble.

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

\newcommand*\MathFamily[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*\MathUp[1]{%
    \ifx\@onlypreamble\@notprerr % not in preamble
        \ifmm@unicodemath
            \DeclareMathSymbol{#1}{\mathalpha}{operators}{`#1}
        \else
            \MathFamily{#1}{0}
        \fi
    \else % in preamble
      \AtBeginDocument{
        \ifmm@unicodemath
            \DeclareMathSymbol{#1}{\mathalpha}{operators}{`#1}
        \else
```

```
            \MathFamily{#1}{0}
        \fi
      }
    \fi
}

\newcommand*\MathIt[1]{%
    \ifx\@onlypreamble\@notprerr % not in preamble
        \ifmm@unicodemath
            \DeclareMathSymbol{#1}{\mathalpha}{letters}{`#1}
        \else
            \MathFamily{#1}{1}
        \fi
    \else % in preamble
      \AtBeginDocument{
        \ifmm@unicodemath
            \DeclareMathSymbol{#1}{\mathalpha}{letters}{`#1}
        \else
            \MathFamily{#1}{1}
        \fi
      }
    \fi
}

```

With a similar approach we could also create additional macros to set any letter in bold or sans serif. However, there is no default family number associated with these typefaces. The family number depends on the font package being loaded and may vary depending on specific `\DeclareSymbolFont` used. Therefore, setting letters in bold or sans serif requires additional consideration and may not have a straightforward solution.

In addition to `\MathUp` and `\MathIt`, we also offer the following command to set a group of letters, among 'e, i, j', in roman typeface.

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

```

`\apply` With the inverse switch `\MathNormal`, you can apply the normal (italic) style on any comma-separated list of characters. This is achieved using the `\apply` macro, e.g. `\apply\macro{arg1,arg2}` expands to `\macro{arg1}\macro{arg2}`. Thus `\apply\MathUp{e,i,j}` is equivalent to `\MathUp{e}\MathUp{i}\MathUp{j}`. I discovered this powerfull macro on `iterate190.rssing.com` by searching for "TeX How to iterate over a comma separated list". The answer was posted under the pseudonym 'wipet' on 2021/02/26. Let its author, Petr Olšák, be thanked. This macro allows to accomplish tasks that usual loop instructions like `\@for` or `\foreach` cannot achieve due to errors like "! Improper alphabetic constant". For instance, if you try `\def\letter{A} \MathUp{\letter}` it will fail because the control sequence `\letter` is not strictly equivalent here to 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 were used until version 2.2 but still work. They were intended to set some letters in upright shape in math mode, but only worked in the preamble. This is now managed by the more powerful `\MathUp` command, and the old commands are maintained but as alias for `\MathUp`.

```
\newcommand{\enumber}{\MathUp{e}}
\newcommand{\inumber}{\MathUp{i}}
\newcommand{\jnumber}{\MathUp{j}}

```

Obtaining an upright Greek letter π must be handled differently. The switches are called `\pinumber` and `\pinormal` and can be used anywhere in the document.

But `\pinumber` must be called first in the preamble with an optional argument. This argument can be a valid command name that produces an upright pi letter (after having loading an appropriate package). Compatibility with unicode-math is a bit tricky! When given without an argument in the preamble, `\pinumber` uses an LGR font encoding called `lmr`. A new feature (v2.11) is to use `\pinumber` with a keyval option to use many other Greek pi letters without loading a whole package, thus without altering the other (italic) Greek letters. We achieve this with `\DeclareSymbolFont` and `\DeclareMathSymbol`. We just have to know the "name" of the desired symbol font.

```
\newif\ifmm@lgr
\define@cmdkey{pifonts}[mm@]{lgrmath}[lmr]{\mm@lgrtrue}
\newif\ifmm@upgreek
\define@choicekey{pifonts}{upgreek}[\mm@upgreek@option]%
    {Euler,Symbol,Symbolsmallscale}[Symbol]{\mm@upgreektrue}
\newif\ifmm@mathdesign
\define@choicekey{pifonts}{mathdesign}[\mm@mathdesign@option]%
    {Utopia,Garamond,Charter}[Charter]{\mm@mathdesigntrue}
\newif\ifmm@kpfonts
\define@choicekey{pifonts}{kpfonts}[\mm@kp@option]%
    {normal,light}[normal]{\mm@kpfontstrue}
\define@boolkeys{pifonts}[mm@]{fourier,pxfonts,txfonts}[true]
\newif\ifmm@fontspec
\define@cmdkey{pifonts}[mm@]{fontspec}[GFS Didot]{\mm@fontspectrue}

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

```
\setkeys{pifonts}{#1}
\let\pi\relax

\ifmm@lgr
    \DeclareFontEncoding{LGR}{}{}
    \DeclareSymbolFont{mmupgr}{LGR}{\mm@lgrmath}{m}{n}
    % may work with bold (b) instead of m
    \DeclareMathSymbol{\pi}{\mathord}{mmupgr}{112}

\else\ifmm@fontspec
    \@ifpackageloaded{fontspec}{}{
        \PackageError{mismath}{\string\pinumber\space with
            the ‘fontspec’ option\MessageBreak
            needs the fontspec package,\MessageBreak
            which must be run with LuaLaTeX or XeLaTeX}{}
    }
    \newfontfamily\mismathgreekfont{\mm@fontspec}[NFSSFamily=mgr]
    \DeclareSymbolFont{mmupgr}{TU}{mgr}{m}{n}
    \Umathchardef\pi="7 \symmmupgr "03C0

\else\ifmm@upgreek
    \ifdefstring{\mm@upgreek@option}{Euler}{
        \DeclareFontFamily{U}{eur}{\skewchar\font’177}
        \DeclareFontShape{U}{eur}{m}{n}{%
          <-6> eurm5 <6-8> eurm7 <8-> eurm10}{}
        \DeclareSymbolFont{mmupgr}{U}{eur}{m}{n}
        \DeclareMathSymbol{\pi}{\mathord}{mmupgr}{"19}
    }{
    \ifdefstring{\mm@upgreek@option}{Symbol}{
        \DeclareSymbolFont{mmupgr}{U}{psy}{m}{n}
        \DeclareMathSymbol{\pi}{\mathord}{mmupgr}{‘p}
    }{
    \ifdefstring{\mm@upgreek@option}{Symbolsmallscale}{
        \DeclareFontFamily{U}{fsy}{}
        \DeclareFontShape{U}{fsy}{m}{n}{<->s*[.9]psyr}{}
        \DeclareSymbolFont{mmupgr}{U}{fsy}{m}{n}
        \DeclareMathSymbol{\pi}{\mathord}{mmupgr}{‘p}
    }{}}}

\else\ifmm@mathdesign
    \ifdefstring{\mm@mathdesign@option}{Utopia}{
        \DeclareSymbolFont{mmupgr}{OML}{mdput}{m}{n}
    }{
    \ifdefstring{\mm@mathdesign@option}{Garamond}{
        \DeclareSymbolFont{mmupgr}{OML}{mdugm}{m}{n}
    }{
    \ifdefstring{\mm@mathdesign@option}{Charter}{
        \DeclareSymbolFont{mmupgr}{OML}{mdbch}{m}{n}
    }{}}}

    \else\ifmm@fourier
        \DeclareFontEncoding{FML}{}{}
        \DeclareSymbolFont{mmupgr}{FML}{futm}{m}{it}

    \else\ifmm@kpfonts
        \ifdefstring{\mm@kp@option}{normal}{
            \DeclareSymbolFont{mmupgr}{U}{jkpmia}{m}{it}
        }{
        \ifdefstring{\mm@kp@option}{light}{
            \DeclareSymbolFont{mmupgr}{U}{jkplmia}{m}{it}
        }{}}

    \else\ifmm@pxfonts
        \DeclareSymbolFont{mmupgr}{U}{pxmia}{m}{it}

    \else\ifmm@txfonts
        \DeclareSymbolFont{mmupgr}{U}{txmia}{m}{it}

    \fi\fi\fi\fi\fi
        \DeclareMathSymbol{\pi}{\mathord}{mmupgr}{"19}
    \fi\fi\fi
}

\newcommand*\pinumber[1][]{%
    \ifthenelse{\equal{#1}{}}{% no argument given
        \ifx\@onlypreamble\@notprerr % not in preamble
            \@ifundefined{savedpi}{
                \PackageWarning{mismath}{%
                    \string\pinumber\space
                    must be used in the preamble first}
            }{\let\pi\savedpi}
        \else % in the preamble
            \AtBeginDocument{
                \let\itpi\pi
                \pifonts{lgrmath}
                \let\savedpi\pi
            }
        \fi
    }{% command name or keyval options, necessarily in the preamble
        \AtBeginDocument{% must be here with unicode-math
            \let\itpi\pi
            \@ifundefined{#1}{%
                \pifonts{#1}
            }{
                \ifmm@unicodemath
                    \ifthenelse{\equal{#1}{uppi}}{% or "1D70B
                        \renewcommand\pi{\symup{\symbol{"03C0}}}
                        \renewcommand\itpi{\symit{\symbol{"03C0}}}
                    }{\renewcommand{\pi}{\csname #1\endcsname}}
                \else
```

```
                    \renewcommand{\pi}{\csname #1\endcsname}
                \fi
            }
            \let\savedpi\pi}
      }
}

\newcommand{\pinormal}{%
  \@ifundefined{itpi}{
    \PackageWarning{mismath}{Command \string\itpi\space undefined,
    \MessageBreak
    use \string\pinumber\space in the preamble first}
  }{
    \ifmm@unicodemath
      \@ifundefined{savedpi}{
        \PackageError{mismath}{Before using \string \pinormal,
        \MessageBreak
        you must call \string\pinumber\space in the preamble}{}}
    \fi
    \let\pi\itpi
  }
}

```

When `\pinumber[`⟨*keyval*⟩`]` has been called, you can also get some other mathematical constants using Greek letters, e.g. γ, the Euler-Mascheroni constant:

```
\let\gamma\relax
\DeclareMathSymbol{\gamma}{\mathord}{mmupgr}{"0D}
```

If unicode-math is used, you must put these commands in `\AtBeginDocument`. The hexadecimal code `"0D` depends on the option passed to `\pinumber` in the preamble (see the command `\pifonts` above and search in package docs).
And to get the golden ratio φ:

```
\let\phi\relax
\DeclareMathSymbol{\varphi}{\mathord}{mmupgr}{"27}
```

To preserve the original γ or φ, you can define `\upgamma` or `\upvarphi` instead.

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{A}#1}}
\newcommand*{\hvec}[1]{\vec{\vphantom{A}#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

\newcommand{\tensor}{\mathbfsfit} % isomath uses \mathsfbfit

\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}}}

\newcommand*\MathProba[1]{%
    \in@{P}{#1} \ifin@ \MathUp{P} \fi
    \in@{E}{#1} \ifin@ \MathUp{E} \fi
    \in@{V}{#1} \ifin@ \MathUp{V} \fi
}

```

Classic identifiers are presented below. They will be defined only if the option `nofunction` has not been activated.

```
\ifmm@nofunction\else
    \mm@operator{\adj}{adj}
    \mm@operator{\Aut}{Aut}
    \mm@operator{\codim}{codim}
    \mm@operator{\coker}{coker}
    \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}
    \mm@operator{\lb}{lb}
    \mm@operator{\lcm}{lcm}
    \mm@operator{\rank}{rank}

    \mm@operator{\Res}{Res}
    \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}
\fi

```

The `\mathcal` alphabet, from the original Computer Modern font family, is used here to produce $\mathcal{R}e$, $\mathcal{I}m$ and $\mathcal{O}$. Several font packages redefines this alphabet producing glyphs that may seem less suitable for the commands below. We have therefore retained the original `\cmmathcal` math alphabet, which can be used for other letters. If unicode-math is called, it will redefine the commands `\Re` and `\Im` in `\AtBeginDocument`, hence the use of '`\AtEndPreamble{\AtBeginDocument{`' to ensure that the mismath redefinition occur after the actions of unicode-math.

```
\DeclareFontFamily{U}{cmsy}{\skewchar\font48 }
\DeclareFontShape{U}{cmsy}{m}{n}{% from mathalpha
    <-5.5> cmsy5%
    <5.5-6.5> cmsy6%
    <6.5-7.5> cmsy7%
    <7.5-8.5> cmsy8%
    <8.5-9.5> cmsy9%
    <9.5-> cmsy10}{}
\DeclareMathAlphabet{\cmmathcal}{U}{cmsy}{m}{n}

\AtEndPreamble{\AtBeginDocument{
    \ifmm@classicReIm\else
        \let\oldRe\Re
        \let\oldIm\Im
    \ifmm@otherReIm
```

```
        \renewcommand{\Re}{\cmmathcal{R}\mathit{e}}
        \renewcommand{\Im}{\cmmathcal{I}\mathit{m}}
    \else
        \renewcommand{\Re}{\operatorname{Re}}
        \renewcommand{\Im}{\operatorname{Im}}
    \fi\fi
}}

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

```

And finally we present the remaining macros.

With Cyrillic languages, the command `\C` may already be defined but only for text mode. Thus, it will not be redefined by mismath. However, when activating `\onlymathC`, you can to use our `\C` macro only for math mode, without interfering the definition of the text `\C` that is already defined.

When using XƎLaTEX or LuaTEX engines with the hyperref package, `\C` will be already defined and you get the message "`Command \C unavailable in encoding TU`". Therefore `\onlymathC` is automatically called in that case.

```
\mm@macro{mathset}{\mathbf}
\mm@macro{R}{\mathset{R}}
\AtBeginDocument{
    \@ifpackageloaded{hyperref}%
        {\iftutex\onlymathC\fi}% LuaTex or XeTeX engines
        {\mm@macro{C}{\mathset{C}}}
}
\providecommand\onlymathC{\let\oldC\C
    \renewcommand{\C}{\TextOrMath{\oldC}{\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{A}#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}

```

Above, we have used `\mathrm` before `\Delta` in case of defining capital Greek letters in italics (for example with the fixmath package).

The use of `\mbox{}` ensures that the space produced by `\` in the `\then` macro is not 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}[3][7mu]{%
        \frac{\mkern#1#2\mkern#1}{\mkern#1#3\mkern#1}} }{
    \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
  \ifmm@multicol
    \renewcommand{\columnseprule}{0.1pt}
    \begin{multicols}{2}
        \par\noindent\hfill
        \begin{math}\begin{aligned}\displaystyle
  \else
    \PackageError{mismath}{The mathcols environment
        needs the multicol package}{Call the package multicol
        in your preamble.}
  \fi
}{%
```

```
        \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] *The mathtools package*, Morten Høgholm, Lars Madsen, CTAN, v1.29 2022/06/29.

[7] *amsmath – AMS mathmatical facilities for LaTeX*, Frank Mittelbach, Rainer Schöpf, Michael Downes, Davis M. Jones, David Carlisle, CTAN, v2.17n 2022/04/08.

[8] *Experimental Unicode mathematical typesetting: The unicode-math package*, Will Robertson, Philipp Stephani, Joseph Wright, Khaled Hosny, and others, CTAN, v0.8r 2023/08/13.

[9] *The fixmath package for LaTeX 2ε*, Walter Schmidt, CTAN, v0.9 2000/04/11.

[10] *isomath – Mathematical style for science and technology*, Günter Milde, CTAN, v0.6.1 2012/09/04.

[11] *PM-ISOmath, The Poor Man ISO math bundle*, the pm-isomath package by Claudio Beccari, CTAN, v1.2.00 2021/08/04.

[12] *The mathgreeks package*, Antoine Missier, CTAN, v1.1 2024/05/04.

[13] *The upgreek package for LaTeX 2ε*, Walter Schmidt, CTAN, v2.0 2003/02/12.

[14] *The mathdesign package*, Paul Pichaureau, CTAN, v2.31 2013/08/29.

[15] *Kp-Fonts – The Johannes Kepler project*, Christophe Caignaert, CTAN, v3.34 20/09/2022.

[16] Fourier-GUT*enberg*, Michel Bovani, CTAN, v1.3 2005/01/30.

[17] *PX Fonts – Palatino-like fonts in support of mathematics*, Young Ryu, CTAN, 2000/12/14.

[18] *TX Fonts – Times-like fonts in support of mathematics*, Young Ryu, CTAN, 2000/12/15.

[19] *The LibertinusT1 Math Package*, Michael Sharpe, CTAN, v2.0.4 2024/01/14.

[20] *The lgrmath package*, Jean-François B., CTAN, v1.0 2022/11/16.

[21] *New TX font package*, Micahel Sharpe, CTAN, v1.735 2024/03/01.

[22] *The textalpha package* (part of the greek-fontenc bundle), Günter Milde, CTAN, v2.1 2022/06/14.

[23] *The spacingtricks package*, Antoine Missier, CTAN, v1.8 2023/12/06.

[24] *Intelligent brackets – The ibrackets package*, Antoine Missier, CTAN, v1.2, 2023/07/26.

[25] *The interval package*, Lars Madsen, CTAN, v0.4 2019/03/06.

[26] *The icomma package for LaTeX 2ε*, Walter Schmidt, CTAN, v2.0 2002/03/10.

[27] *The ncccomma package*, Alexander I. Rozhenko, CTAN, v1.0 2005/02/10.

[28] *The decimalcomma package*, Antoine Missier, CTAN, v1.4 2023/12/30.

[29] *L'extension pour TeX et LaTeX systeme*, Christian Tellechea, CTAN, v0.32 2019/01/13.

[30] *The spalign package*, Joseph Rabinoff, CTAN, 2016/10/05.

[31] *The package nicematrix*, François Pantigny, CTAN, v6.14 2023/02/18.

[32] *L'extension frenchmath*, Antoine Missier, CTAN, v3.0 2024/05/04.

[33] *The Not So Short Introduction to LaTeX 2ε*, 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`.

[34] *The LaTeX Companion*, Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, Chris Rowley, 2nd edition, Pearson Education, 2004.