The **mathastext** package

JEAN-FRANÇOIS BURNOL
contact: jfbu (at) free (dot) fr

Package version: 1.4e (2024/10/26)

The **mathastext** package changes the fonts which are used in math mode for letters, digits and a few other punctuation and symbol signs to replace them with the font as used for the document text. Thus, the package makes it possible to use a quite arbitrary font without worrying too much that it does not have specially designed accompanying math fonts. Also, **mathastext** provides a simple mechanism in order to use more than one math-as-text font in the same document.

```
`mathastext' is a LaTeX package

    \usepackage{mathastext}

The document will use in math mode the text font
as configured at package loading time, for these
characters:

    abcdefghijklmnopqrstuvwxyz
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    0123456789
    !?,.:;+-=()[]/#$%&<>|{}\

The command \MTsetmathskips allows to set up extra
spacings around each given letter.

Use multiple \Mathastext[name]'s to define in the
preamble various math versions using each a given
text font, to be later activated in the document
body via the command \MTversion{name}.

With the subdued option, mathastext will be active
only inside such math versions distinct from the
normal and bold.

Main options: italic, frenchmath, defaultmathsizes,
              subdued, asterisk, LGRgreek.
```

Documentation generated from the source file with Time-stamp: <26-10-2024 at 17:38:46 CEST>.

Contents

1 What mathastext does

If you have used the package before please make sure to check first section 4 where all changes across releases are recorded.

All blue colored words, such as `\Mathastext` or `italic`, are hyperlinked to their offical descriptions located either in the section 2 (Package commands) or subsection 3.2 (Complete list of options).

1.1 Aim of this package and basic usage

The initial ideology of mathastext was to produce mathematical texts with a very uniform look, not separating math from text as strongly as is usually done.

mathastext's basic aim is thus to have the same font for text and mathematics. With hundreds of free text fonts packaged for LaTeX and only a handful of math ones, chances are your favorite text font does not mix so well with the available math ones; mathastext may then help. Note that mathastext was initially developed for the traditional TeX fonts and engines, and that compatibility with Unicode engines and OpenType fonts is partial.

Here is a minimal example of what may go into the preamble:

```
\usepackage[T1]{fontenc}
\usepackage{times}
\usepackage[italic]{mathastext}
```

The package records which font is set up for text, at the time it is loaded, and then arranges things in order for this text font to be used in math mode as well. So, with the preamble as above all letters, digits, and punctuation signs inside math mode will then be typeset in Times.[1] The exact list of characters concerned by **`mathastext`** is a subset of the basic `ASCII` set:

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
! ? * , . : ; + − = ( ) [ ] / # $ % & < > | { } and ****

As one can see, this is a very limited list! Some possibilities exist regarding Greek letters and will be described later.

1.2 Examples

Here is another simple example:

```
\usepackage{libertinus-type1}
\usepackage[italic,LGRgreek,defaultmathsizes]{mathastext}
```

The `LGRgreek` option is there to take advantage that the libertinus-type1 package[2] also provides Greek letters in LGR encoding, which can thus be used by **`mathastext`** in math mode. And we do here as if we did not know about the existence of the libertinust1math package![3] This would have been the obvious choice, but then one wouldn't need **`mathastext`** and I couldn't even start this documentation.

More sophisticated preambles will use multiple times the `\Mathastext` command in the preamble with its optional argument [⟨*math_version*⟩] in order to define *math versions* corresponding to a given font configuration. These **`mathastext`**-enriched math versions are then activated in the document body via the `\MTversion`{⟨*math_version*⟩} command, which modifies *both* the text font and the math font.

---

[1] let's do as if we did not know the excellent txfonts package which employs Times for text and has a very complete math support, including many additional mathematical glyphs in comparison to the CM fonts. *This was written many years ago, nowadays,* newtx *is the successor of* txfonts*.*

[2] Bob TENNENT, *Support for using Libertinus fonts with LaTeX/pdfLaTeX*, `https://ctan.org/pkg/libertinus-type1`.

[3] Michael SHARPE, *A Type 1 font and LaTeX support for Libertinus Math*, `https://ctan.org/pkg/libertinust1math`.

We now give some examples with a verbatim copy of the preamble code corresponding to them, as can be found in the source of this documentation. The detailed option and command descriptions will be given later.

First of all, the package was loaded using this:

```
\usepackage[subdued,%
            asterisk,%
            defaultmathsizes,%
            symbolmisc,symbolre,%
            LGRgreek]{mathastext}
```

In the definitions of the **mathastext**-enriched *math versions* we keep commands which may have been redundant in the original preamble, because they were issued earlier for a previous math version definition.

Let's start with Latin Modern typewriter proportional. Its usage was configured in the preamble using this:

```
\MTlettershape{n}
\MTupgreek
\MTgreekfont{cmtt}
\MTfamily{lmvtt}
\Mathastext[lmvtt]
```

Its usage is triggered using

```
\MTversion{lmvtt}
```

in the document. Here is an example:

> Let (X, Y) be two functions of a variable a. If they obey the differential system $(VI_{\nu,n})$:
>
> $$a\frac{d}{da}X = \nu X - (1 - X^2)\frac{2na}{1 - a^2}\frac{aX + Y}{1 + aXY}$$
> $$a\frac{d}{da}Y = -(\nu + 1)Y + (1 - Y^2)\frac{2na}{1 - a^2}\frac{X + aY}{1 + aXY}$$
>
> then the quantity $q = a\frac{aX+Y}{X+aY}$ satisfies as function of $b = a^2$ the P_{VI} differential equation:
>
> $$\frac{d^2q}{db^2} = \frac{1}{2}\left\{\frac{1}{q} + \frac{1}{q-1} + \frac{1}{q-b}\right\}\left(\frac{dq}{db}\right)^2 - \left\{\frac{1}{b} + \frac{1}{b-1} + \frac{1}{q-b}\right\}\frac{dq}{db}$$
> $$+\frac{q(q-1)(q-b)}{b^2(b-1)^2}\left\{\alpha + \frac{\beta b}{q^2} + \frac{\gamma(b-1)}{(q-1)^2} + \frac{\delta b(b-1)}{(q-b)^2}\right\}$$
>
> with parameters $(\alpha, \beta, \gamma, \delta) = (\frac{(\nu+n)^2}{2}, \frac{-(\nu+n+1)^2}{2}, \frac{n^2}{2}, \frac{1-n^2}{2})$.
>
> Test of uppercase Greek in math: $AB\Gamma\Delta\Xi\Omega$.

Both the Latin and Greek letters are upright, in conformity to the way the `lmvtt` version was defined.

Now with the fonts from the `libertinus-type1` distribution[4]. The preamble code is:

```
\MTfamily{LibertinusSerif-TLF}
\MTlettershape{n}
\MTseries{m}
\MTgreekfont{LibertinusSerif-TLF}
\MTupgreek
\Mathastext[libertinus]
\MTseries{sb}
\Mathastext[libertinussemibold]
```

Its usage in the document body for the example below is triggered via

```
\MTversion[libertinus]{libertinussemibold}
```

This syntax modifies the text fonts to be those which were defined to hold for the **`mathastext`**-math version passed as optional argument, and sets the math fonts according to the mandatory argument. Hence the math mode uses semibold font but the text font uses the normal weight.

Let (X, Y) be two functions of a variable a. If they obey the differential system $(VI_{\nu,n})$:

$$a\frac{d}{da}X = \nu X - (1 - X^2)\frac{2na}{1-a^2}\frac{aX+Y}{1+aXY}$$

$$a\frac{d}{da}Y = -(\nu+1)Y + (1 - Y^2)\frac{2na}{1-a^2}\frac{X+aY}{1+aXY}$$

then the quantity $q = a\frac{aX+Y}{X+aY}$ satisfies as function of $b = a^2$ the P_{VI} differential equation:

$$\frac{d^2q}{db^2} = \frac{1}{2}\left\{\frac{1}{q} + \frac{1}{q-1} + \frac{1}{q-b}\right\}\left(\frac{dq}{db}\right)^2 - \left\{\frac{1}{b} + \frac{1}{b-1} + \frac{1}{q-b}\right\}\frac{dq}{db}$$
$$+ \frac{q(q-1)(q-b)}{b^2(b-1)^2}\left\{\alpha + \frac{\beta b}{q^2} + \frac{\gamma(b-1)}{(q-1)^2} + \frac{\delta b(b-1)}{(q-b)^2}\right\}$$

with parameters $(\alpha, \beta, \gamma, \delta) = (\frac{(\nu+n)^2}{2}, \frac{-(\nu+n+1)^2}{2}, \frac{n^2}{2}, \frac{1-n^2}{2})$.

Test of uppercase Greek in math: $\mathrm{AB}\Gamma\Delta\Xi\Omega$.

Now with a Times clone. We will configure Latin letters to be in italic shape, and Greek letters to be italic for lowercase and upright for uppercase:

```
\usepackage{times}% it modifies the \{rm,sf,tt}default's
\MTfamily{\rmdefault}
\MTlettershape{it}
\MTseries{m}
```

---

[4] Bob Tennent, *Support for using Libertinus fonts with LaTeX/pdfLaTeX*, https://ctan.org/pkg/libertinus-type1.

```
\MTgreekfont{txr}
\MTitgreek\MTupGreek
\Mathastext[times]
% \MTversion{times} will change not only math but also text, so it
% will re-enact the \rmdefault, \sfdefault, \ttdefault from loading times.sty
```

We now use this in the document body via

`\MTversion{times}`

Let (X, Y) be two functions of a variable a. If they obey the differential system $(VI_{\nu,n})$:

$$
\begin{aligned}
a\frac{d}{da}X &= \nu X - (1 - X^2)\frac{2na}{1-a^2}\frac{aX+Y}{1+aXY}\\
a\frac{d}{da}Y &= -(\nu+1)Y + (1 - Y^2)\frac{2na}{1-a^2}\frac{X+aY}{1+aXY}
\end{aligned}
$$

then the quantity $q = a\frac{aX+Y}{X+aY}$ satisfies as function of $b = a^2$ the P_{VI} differential equation:

$$
\begin{aligned}
\frac{d^2q}{db^2} = \frac{1}{2}\left\{\frac{1}{q} + \frac{1}{q-1} + \frac{1}{q-b}\right\}\left(\frac{dq}{db}\right)^2 - \left\{\frac{1}{b} + \frac{1}{b-1} + \frac{1}{q-b}\right\}\frac{dq}{db}\\
+\frac{q(q-1)(q-b)}{b^2(b-1)^2}\left\{\alpha + \frac{\beta b}{q^2} + \frac{\gamma(b-1)}{(q-1)^2} + \frac{\delta b(b-1)}{(q-b)^2}\right\}
\end{aligned}
$$

with parameters $(\alpha, \beta, \gamma, \delta) = (\frac{(\nu+n)^2}{2}, \frac{-(\nu+n+1)^2}{2}, \frac{n^2}{2}, \frac{1-n^2}{2})$.

Test of uppercase Greek in math: $\mathrm{AB}\Gamma\Delta\Xi\Omega$.

Let us be a bit more original and have our mathematics with italic letters from the sans serif font Helvetica, while the letters in text use New Century Schoolbook. Also we want Greek letters (both lowercase and uppercase) to be in italic shape. The preamble code was:

```
\usepackage{newcent}% attention that it modifies all three of \rmdefault,
                    % \sfdefault and \ttdefault
\MTfamily{\rmdefault}
\MTlettershape{it}
% \MTitgreek\MTupGreek % our demo does not use newcent for math anyway
\Mathastext[newcent]

\usepackage[scaled]{helvet}
\MTfamily{\sfdefault}
\MTlettershape{it}  % redundant here
\MTseries{m}
\MTitgreek          % make both lowercase and uppercase Greek italic
\MTgreekfont{cmss}
\Mathastext[helvet]
```

And the next demo is configured in the document body via

`\MTversion[newcent]{helvet}`

Let (X, Y) be two functions of a variable a. If they obey the differential system $(VI_{\nu,n})$:

$$
\begin{aligned}
a\frac{d}{da}X &= \nu X - (1 - X^2)\frac{2na}{1-a^2}\frac{aX+Y}{1+aXY}\\
a\frac{d}{da}Y &= -(\nu+1)Y + (1 - Y^2)\frac{2na}{1-a^2}\frac{X+aY}{1+aXY}
\end{aligned}
$$

then the quantity $q = a\frac{aX+Y}{X+aY}$ satisfies as function of $b = a^2$ the P_{VI} differential equation:

$$
\begin{aligned}
\frac{d^2q}{db^2} = \frac{1}{2}\left\{\frac{1}{q} + \frac{1}{q-1} + \frac{1}{q-b}\right\}\left(\frac{dq}{db}\right)^2 - \left\{\frac{1}{b} + \frac{1}{b-1} + \frac{1}{q-b}\right\}\frac{dq}{db}\\
+\frac{q(q-1)(q-b)}{b^2(b-1)^2}\left\{\alpha + \frac{\beta b}{q^2} + \frac{\gamma(b-1)}{(q-1)^2} + \frac{\delta b(b-1)}{(q-b)^2}\right\}
\end{aligned}
$$

with parameters $(\alpha, \beta, \gamma, \delta) = (\frac{(\nu+n)^2}{2}, \frac{-(\nu+n+1)^2}{2}, \frac{n^2}{2}, \frac{1-n^2}{2})$.

Test of uppercase Greek in math: $AB\Gamma\Delta\Xi\Omega$.

And after all that, we may wish to return to the default math typesetting (let's shorten the extract here in case the reader makes an indigestion ...). This is easy because all previous usages were enclosed in braces `{...}` so as to limit the scope. As **`mathastext`** was loaded with option **`subdued`** the default rendering (i.e. in the *normal* and *bold* math versions) is (almost) as if the package was not loaded at all, and it simply matches the document font configuration. Here it thus matches the

`\usepackage{mlmodern}`

which was included in the document preamble prior to loading **`mathastext`**.

Let (X, Y) be two functions of a variable a. If they obey the differential system $(VI_{\nu,n})$:

$$
\begin{aligned}
a\frac{d}{da}X &= \nu X - (1 - X^2)\frac{2na}{1-a^2}\frac{aX+Y}{1+aXY}\\
a\frac{d}{da}Y &= -(\nu+1)Y + (1 - Y^2)\frac{2na}{1-a^2}\frac{X+aY}{1+aXY}
\end{aligned}
$$

then the quantity $q = a\frac{aX+Y}{X+aY}$ satisfies as function of $b = a^2$ the P_{VI} differential equation with parameters $(\alpha, \beta, \gamma, \delta) = (\frac{(\nu+n)^2}{2}, \frac{-(\nu+n+1)^2}{2}, \frac{n^2}{2}, \frac{1-n^2}{2})$.

Test of uppercase Greek in math: $\Gamma\Delta\Xi\Omega$ (no \Alpha, no \Beta).

If the scope of our earlier examples using **`mathastext`**-enriched math versions had not been limited we would have issued

```
\MTversion{normal}
```

to return to the normal (almost not influenced by **`mathastext`**) math version.

The Greek letters varied across our examples thanks to the `LGRgreek` option which made the `\MTgreekfont` command active for configuration of the math versions.[5]

Since `1.3y` this documentation uses globally the `mlmodern`[6] font package and has added an example using the Libertinus font in type-1 format[7] although there is an existing accompanying math font[8].

1.3 Main options

1.3.1 The `italic` option

In the initial version `1.0`, the Latin letters in mathematical mode assumed the exact same shape as in text mode, and this meant, generally speaking, that they would turn up upright. Doing this gives a very uniform look to the document, so that one has to make an effort and read it with attention, and this was one of the design goals of **`mathastext`**.

Nevertheless, soon after I posted the initial version of the package to CTAN, I was overwhelmed by numerous[9] questions[10] on how to have the letters be in italic shape.

The default is still, as in version `1.0`, for everything to be in upright shape, but it suffices to pass to the package the option `italic` to let the Latin letters in math (1.1)
mode be in italic shape.[11]

---

[5] The document used the `cmtt`, `cmss`, `txr`, as well as `LibertinusSerif-TLF` font families in LGR encoding. The first two are available (with no need to load explicitly any package in the document) if the LaTeX installation provides the cbfonts (or cbgreek-complete) & babel packages, and the LGR encoded `txr` font (again no package loading is necessary) is part of the files of the txfontsb package. For `LibertinusSerif-TLF`, the files of the libertinus-type1 package must be present.

[6] Daniel Benjamin MILLER, *A blacker Type 1 version of Computer Modern, with multilingual support*, `https://ctan.org/pkg/mlmodern`. I have added to the preamble

```
\DeclareEncodingSubset{TS1}{mlmtt}{0}
```

to circumvent some LaTeX complaints about `\textasciigrave` (this is a widespread problem when not using default fonts) related to occurrences of the backtick character in verbatim displays.

[7] Bob TENNENT, *Support for using Libertinus fonts with LaTeX/pdfLaTeX*, `https://ctan.org/pkg/libertinus-type1`.

[8] Michael SHARPE, *A Type 1 font and LaTeX support for Libertinus Math*, `https://ctan.org/pkg/libertinust1math`. Note that it is then highly advantageous to use `latex+dvipdfmx` and not `pdflatex` for reasons of PDF file size.

[9] this means "more than one."

[10] I thank in particular Tariq PERWEZ and Kevin KLEMENT for their kind remarks (chronological order).

[11] more precisely stated, the value of `\itdefault` is used.

1.3.2 The `frenchmath` option

It is a variant of the `italic` option which keeps the uppercase Latin letters in upright shape[12]. Also lets the Greek letters, if the latter are under **`mathastext`** influence, be all upright, lowercase as uppercase.

1.3.3 The `defaultmathsizes` option

The default sizes give for subscripts of subscripts barely legible glyphs (author's opinion!). So **`mathastext`** makes more reasonable choices. It also redefines `\Huge` and defines a `\HUGE` size, copied from the moresize package. To cancel all of this use option `defaultmathsizes`.

1.3.4 The `subdued` option

This option was introduced in `v1.15`. It provides a manner to switch on the **`mathastext`**-*ification* only for limited portions of the document, with the help of the mechanism of math versions. Without the `subdued` option, the *mathastextification* applies by default to the whole of the document (and one may also define additional math versions in the preamble); with the `subdued` option the *mathastextification* is done only in *math versions* distinct from the standard and bold ones. `(1.15)`

Despite some limitations I will now partially describe, the `subdued` option has its utility, as I think is illustrated enough by the examples given at the start of this document and it works reasonably well.

`mathastext` was not written initially in order to allow its action to be completely canceled. It does not store (all) mathcodes nor does it set them (all) when changing math versions; only that would allow a perfect subdued mode (and LaTeX is rather obstinate in making that tricky or at least uneasy if sticking to its official interface to math mode, as it is almost entirely preamble only).

Releases `1.3t` and `1.3u` do this kind of things to maintain usability across multiple **`mathastext`**-ified math versions of characters which are obviously font encoding dependent such as the minus sign as en-dash (or unicode minus), the dotless i, the `\hbar`, the text accents.

But this should be extended to all **`mathastext`**-ified characters which basically would amount to an extensive rewrite of large legacy portions of the code. Currently the support for the `subdued` mode and to multiple math versions amounts to some kind of a kludge, added to an initial design which handled a single unique text font.

To get the displayed math (almost) as if **`mathastext`** had not been loaded, one must also use the option `defaultmathsizes`. But this does not quite suffice, as, for

---

[12] more precisely stated, the value of `\shapedefault` is used.

example, the colon, the dot, and the minus sign belong in the default LaTeX math mode set-up to three distinct fonts whereas `mathastext` will pick (even subdued) the three of them in the same font,[13] and although it will make a reasonable choice of this font, this is not an exact re-installement of the previously prevailing situation. And then other packages could have done arbitrary things regarding character mathcodes, so to be on the safe side one needs the `basic` option which limits the mathastextification to letters and digits.[14] [15] [16] Even then, in some circumstances, this may not suffice: for example the euler package declares the digits to be picked from the same font as the Latin letters, but the subdued `mathastext` "normal" math version will pick them from the same font as used for operator names, which here with the euler package is the document body default text font.

The `frenchmath` option effect applies *also* to the `subdued` "normal" and "bold" math versions.

1.3.5 The `LGRgreek` option

There is the issue of Greek letters. Sometimes the text font has Greek glyphs, in `LGR` encoding[17] (this should be mentioned in the documentation of the font package). Then option `LGRgreek` tells `mathastext` to pick up these Greek letters.

It is naturally possible to leave the responsability to set up Greek letters to some other packages loaded previously to `mathastext`. And even if `mathastext` has been loaded with one of its Greek related options the command `\MTstandardgreek` will locally cancel its customization of Greek letters. The command `\MTcustomgreek` reenables the customization done by `mathastext`, if it was loaded with the `LGRgreek` or one of the other Greek related options.

Release `1.3y` has added important new aspects to the handling of Greek letters via the `LGRgreek` option. Make sure to read the subsubsection 1.7.3.

1.4 Miscellanea

Please note that most material to be found in this section was written many years ago (except the two subsections on `frenchmath` on one hand and intervals on the

---

[13] The minus sign is now perfectly subdued, because its original mathcode is stored and restored; this was only way to handle the case with Unicode engines where the math operator font is in a classic TeX encoding, but the minus sign is configured by `mathastext` to use a Unicode en-dash or minus character in non-subdued math versions. `(1.3t)`

[14] The `subdued` mode does extinguish in the normal and bold math versions the action of options `selfGreek`, `eulergreek`, and `symbolgreek` (previously only `LGRgreek` was subdue-able). `(1.3d)`

[15] The `\imath` and `\jmath` now obey the subdued regime. `(1.3t)`

[16] Also `\hbar` and the math accents (see `mathaccents` option) obey the subdued regime. `(1.3u)`

[17] For example the default CM and its replacement Latin Modern for european languages are (transparently to the user) extended with `LGR` encoded fonts from the `cbfonts` (cbgreek-complete) TeXLive package.

other hand). But it should still be valid!

Ultimately most information here should be moved into the reference sections section 2 and subsection 3.2, and only some generalities should be kept here.

1.4.1 Load **`mathastext`** always last

The "large" math symbols are not modified in any way by **`mathastext`**. Only loading some math font packages such as fourier, kpfonts, mathabx, mathdesign, txfonts, newtxmath, libertinust1math, others… will change them. Think of loading these packages before **`mathastext`**, else they might undo what **`mathastext`** did.

More generally any package (such as amsmath) dealing with math mode should be loaded *before* **`mathastext`**.

1.4.2 Avoid `OT1` encoding

The default `OT1` does not have the $<>|\{\}$ and $\backslash$ glyphs. If **`mathastext`** detects `OT1` as the default encoding it will leave these characters to their defaults from the math fonts.[18]

If **`mathastext`** detects the obsolete `OT1` encoding it does not do anything with $<$, $>$, $|$, $\{$, and $\}$ which (except for monospace fonts) are not available in that encoding. To fully benefit from **`mathastext`** it is recommended to use some other encoding having these glyphs such as `T1` or `LY1`.

1.4.3 Derivative, minus, asterisk

The text characters ' and - are not used, and the asterisk is done only optionally:

- the derivative sign $'$ is left to its default as the text font glyph ' is not, as a rule, a satisfying alternative.[19]
- for the minus sign **`mathastext`** uses the endash character –, if available, and not the hyphen character -. With an OpenType font, **`mathastext`** uses per default the EN DASH U+2013 (see `unicodeminus`).
- the `asterisk` option is required for **`mathastext`** to use the text font for the binary infix math operator $*$ and the control sequence `\ast`. They will use then the text asterisk * suitably lowered, and with the correct spaces around it as binary operator.

---

[18] The `subdued` option, described next, acts a bit otherwise, it forces, contrarily to its usual low-key character, the replacement of `OT1` by `T1` for the fonts ultimately used with letters and digits in math mode.

[19] `v1.2` adds a customizable tiny space before $'$ to separate it from the previous letter, this is really needed when using upright letters in math mode with the CM derivative glyph.Compare f' with f'.

Attention that with this option, inputs such as `$R^*$` or `$R^\ast$` raise errors and *must* be replaced by `$R^{*}$`, respectively `$R^{\ast}$`.

1.4.4 Large symbols and delimiters

Nothing is changed to the "large" math symbols, except for $\prod$ and $\sum$ in inline math which, like here: $\prod\sum$, will be taken from the Symbol Font if option `symbolmisc` was used.

The left and right delimiters are taken from the text font only for the base size: any one of `\big`, `\bigl`, `\bigr`, etc... will trigger the use of the original math symbols.

1.4.5 amsmath

The behavior of the `\DeclareMathOperator` command of `amsmath` is modified by **`mathastext`** for it to use the correct font. Additionally, release `1.3n` of **`mathastext`** at long last also handles an extra operation done by `amsmath` for `'.:/-*` to be used in operator names without the extra math spacing.[20] This customization is suppressed in `subdued` mode for the `normal` and `bold` math versions. (1.3n)

1.4.6 hbar

The default LaTeX definition of `\hbar` would in our context make use of the `h` of the current math font (so for us, it is also the text font, perhaps in italic shape), but with a bar accross the `h` from the original default math font for letters (usually `cmmi`). We redefine `\hbar` to use the text font macron accent (`\=`) as a mock math accent (this takes into account the `italic` option and is compatible with subscripts and superscripts).

Since `1.12` **`mathastext`** when dealing with a Unicode font sets the `\hbar` to be the character from the font having hexadecimal codepoint `U+0127`.

Since `1.3u` the general 8bits font encoding is supported (see discussion of the `mathaccents` option at end of this list for the shared limitations). Brief testing with various usual TeX fonts shows that the vertical positioning of the bar isn't satisfying. It is planned to either add a parameter to adjust it or to modify altogether the mode of construction of the `\hbar`. (1.3u)

Use `nohbar` to tell **`mathastext`** not do provide its own `\hbar`.

---

[20] To the experts: there is a long story here that `\newmcodes@` hardcodes the font, that it was not compatible with Unicode engines, that during some time (2013-2016) `lualatex-math` fixed that and very recently `amsopn.sty 2016/03/08 v2.02` also, so now `lualatex-math 1.6` does nothing as it is already fixed "upstream" in `amsopn.sty`, but anyhow in both cases, this still hardcoded the font, so finally **`mathastext`** does the right thing from its point of view. See the code comments for more, there is an issue here with LuaLaTeX not applying the curly right quote contrarily to XeLaTeX.

1.4.7 Dotless i and j

By default the package redefines `\imath` and `\jmath` to give (in math mode) the dotless i and j (if it exists at all) from the text font.

1.4.8 fontspec

fontspec has to be loaded with the option `no-math`, *and before* **`mathastext`**.

1.4.9 vec accent

The default `\vec` accent is not appropriate for upright letters, so **`mathastext`** provides under option `fouriervec` a math accent control sequence `\fouriervec` which takes its glyph in a Fourier font. A poorman Ersatz `\pmvec` is always available ; it is reasonably good looking on upright letters and works with the `\rightarrow` glyph.

1.4.10 math accents

If option `mathaccents` is used then **`mathastext`** attempts to let the math accents `\acute`, `\grave`, etc... use the suitable glyphs from the text font.

The `\vec` math accent is not handled here, as it is not available in the usual 8bits font encodings. See the `fouriervec` option or the `\pmvec` command.

The math accents obey the `subdued` option and will change in sync with the **`mathastext`**-ified text font used in each non subdued math version. (1.3u)

(Very) brief testing during `1.3u` development with XeTeX and LuaTeX let the author conclude that usage with the `\Umathaccent` primitive of an OpenType accent glyph slot (which in the text font is for usage as a postpended combining character) gives definitely bad horizontal placements for both engines (each in its own way). Thus, the redefinitions of accents for a **`mathastext`** declared math version with an OpenType font is by default canceled.[21] Use `unimathaccents` to force usage of the OpenType font text accents glyph slots with the `\Umathaccent` primitive. Expert users are invited to check out the code and to contribute suggestions if some extras can improve it.

1.4.11 Sans serif in math

The following set-up often gives esthetically pleasing results: it is to use the sans-serif member of the font family for math, and the serif for text.

```
\renewcommand\familydefault\sfdefault
\usepackage{mathastext}
\renewcommand\familydefault\rmdefault
\begin{document}
```

---

[21] I.e., the `\grave` etc... control sequences will, in math versions with an OpenType **`mathastext`**-ified font, expand to macros holding their initial meanings, unmodified by **`mathastext`**, which was in force at the `\begin{document}`.

1.4.12 `mathastext` with beamer

Starting with release `3.34` of beamer[22], **`mathastext`** is recognized as a "math font package".

Only with **earlier** beamer versions *is it necessary to issue*

`\usefonttheme{professionalfonts}`

in the preamble. Example:

```
\documentclass{beamer}
%\usefonttheme{professionalfonts}% obsolete for mathastext since beamer 3.34
\usepackage{newcent}
\usepackage[scaled=.9]{helvet}
\renewcommand{\familydefault}{\rmdefault}
\usepackage[defaultmathsizes,symbolgreek]{mathastext}
\renewcommand{\familydefault}{\sfdefault}
\begin{document}
\begin{frame}
  This is some text and next comes some math: $E=mc^2$
  \[
  E=mc^2=a^n+b^n-c^n=\alpha\beta\gamma
  \]
  \begin{align}
    E&=mc^2\\
    E&=h\nu
  \end{align}
  And again some text.
\end{frame}
\end{document}
```

1.4.13 `mathastext` with frenchmath

To use **`mathastext`** concurrently with the frenchmath package[23] [24]of Antoine MISSIER:

- load frenchmath with its option `capsit`,
- and load **`mathastext`** afterwards (with possibly some font packages loaded in-between), passing it the option **`frenchmath*`**.

Limited testing indicated that the combination of the two packages (using the options as indicated above) works satisfactorily. There may be some minor adjustments to do, as the **`mathastext`**-ified math font may cause issues to some of the

---

[22] Till TANTAU, Joseph WRIGHT, Vedran MILETIĆ, *A LaTeX class for producing presentations and slides*, `https://ctan.org/pkg/beamer`.

[23] Antoine MISSIER, *Typesetting mathematics according to French rules*, `https://ctan.org/pkg/frenchmath`.

[24] The package mismath also by the Antoine MISSIER may probably be used with **`mathastext`**, but not in a fully inter-operative way, as the two packages conflict on some aspects. Reports welcome, we have not tested this.

frenchmath macros: for exemple `\Oijk` may not work well simply due to the font lacking a dotless `j`, but use then **`defaultimath`**.

You can either use the Greek related options of frenchmath or those of **`mathastext`**. Quite certainly better not to use both at same time, anyhow this has not been tested and is not supported.

1.4.14 Intervals and separators

For appropriate mark-up and typesetting of intervals with conventions about opening and closing delimiters which are not the default TEX ones, one may use the mathtools[25] provided `\DeclarePairedDelimiterX`. For example, here is how one can define an `\Ioo` macro (the letter "o" standing for "open") for typesetting an open (in the mathematical meaning of the word) interval using square brackets:

```
\DeclarePairedDelimiterX\Ioo[2]{]}{[}{#1;#2}
```

Use then `$I = \Ioo{A}{B}$` type mark-up in your source, and the derived variants `\Ioo*` or `\Ioo[\Big]` for example will also work.

Note for very advanced users: if employing **`\MTnonlettersobeymathxx`**, our `\Ioo` must be used as `\Ioo*` or `\Ioo[\Big]` (for example) else it raises an error. Alternatively, replace in the above `]` by `{]}` and `[` by `{[}` and then `\Ioo` works (and also `\Ioo[\Big]`). But `\Ioo*` is broken. This is a known limitation of the **`\MTnonlettersobeymathxx`** functionality, and is one reason why **`mathastext`** does not make it the default behavior.

We used in this example the semi-colon as separator. This is seen sometimes in contexts where the interval extremities are decimal numbers, and the language convention is to use the comma as decimal point. The **`binarysemicolon`** option tells **`mathastext`** to configure the `;` character to use in math mode "binary infix operator" type spacing, matching observed practice in some mathematical contexts. The **`binarysemicolon`** option is executed automatically by **`mathastext`** on receiving either the **`frenchmath*`** or the **`frenchmath+`** options.

On the topic of the decimal point, it is recommended to use the `\np` macro from the numprint[26] package with its `autolanguage` and `np` options. This is the best choice if one may have to also use the same mathematical expression with numerical quantities in another language having different conventions.

For those languages such as French where the convention is to use as decimal separator a colon, you may alternatively pass to **`mathastext`** either the **`decimalcomma`** or the **`ncccomma`** options, to tell it to load the eponymous packages decimalcomma[27] or ncccomma[28] respectively, which make the comma (to some extent) 'intelligent', i.e. decide on the spacing type (ordinary or punctuation) depending on next token.

---

[25] Morten Høgholm, Lars Madsen and the LaTeX3 project, *Mathematical tools to use with amsmath*, `https://ctan.org/pkg/mathtools`. As explained elsewhere in this documentation always load **`mathastext`** after mathtools.

[26] Harald Harders, *Print numbers with separators and exponent if necessary*, `https://ctan.org/pkg/numprint`.

[27] Antoine Missier, *Comma for decimal numbers*, `https://ctan.org/pkg/decimalcomma`.

[28] Alexander I. Rozhenko, *Use comma as decimal separator in mathematics*, `https://ctan.org/pkg/ncccomma`.

Do not load directly the packages but simply use the corresponding option and **`mathastext`** will do the loading and take appropriate needed measures for compatibility. The `decimalcomma` option is included in the `frenchmath*` option, and the (1.3zb)
`ncccomma` option is included in the `frenchmath+` option. (1.3zb)

Let's give another example of usage of mathtools here to define a macro for integer ranges:

```
\DeclarePairedDelimiterX\Iffint[2]{\llbracket}{\rrbracket}{#1,#2}
```

This used control sequences `\llbracket` and `\rrbracket` from the fourier package (and possibly others). A poorman definition might be:

```
\ifdefined\llbracket\else \def\llbracket{{[\![}}\fi
\ifdefined\rrbracket\else \def\rrbracket{{]\!]}}\fi
```

Regarding open intervals in the French notation such as `]a,b[`, an alternative avoiding usage of specific mark-up is provided by the ibrackets[29] package which makes the square brackets mathematically active, in the same spirit as for the (semi) 'intelligent' comma mentioned above. Brief testing indicates this package is compatible with **`mathastext`**, even when using multiple math versions. Read the fine print below for some limitations though.

Note for very advanced users: compatibility is only partial as the effect of ibrackets is canceled after `\MTnonlettersobeymathxx`. This is expected and a special compatibility layer would be needed, of the same type as has been done to support fully the decimalcomma and ncccomma packages via eponymous **`mathastext`** options. There is no plan at this time to add such a patch making the compatibility exhaustive.

It is possible to use the `noparenthesis` option to turn off completely the **`mathastext`** actions on square brackets (and parentheses).

1.4.15 Unicode engines

`mathastext` is minimally Unicode aware since 1.12 and can be used with XƎTEX or LuaTEX. Starting with release 1.3, it needs `luatex` to be at least as recent as the one which was provided with the TL2013 distribution.

However **`mathastext`** applies only to (a subset of) the `32-127` ascii range, and optionally to Greek letters, but for the latter only if provided via "TEX fonts" such as Euler, Symbol or LGR-encoded fonts. It does not know how to use a given Unicode font simultaneously for Latin and Greek letters.

Thus, first consider much better alternatives:

- Since 2018, the package mathfont[30] adapts Unicode text fonts to usage in math mode. It works with both XƎTEX and LuaTEX.
- For XƎTEX only, mathspec[31] also allows usage of arbitrary text fonts in mathematics.

---

[29] Antoine Missier, *Intelligent brackets*, `https://ctan.org/pkg/ibrackets`.

[30] Conrad Kosowsky, *Use TrueType and OpenType fonts in math mode* `https://ctan.org/pkg/mathfont`.

[31] Andrew Gilbert Moschou, *Specify arbitrary fonts for mathematics in XƎTEX* `https://ctan.org/pkg/mathspec`.

- and of course unicode-math[32] is the standard package for using OpenType fonts which are equipped with the needed extra support being used in TEX math mode.

If using any one of the above you probably don't need, don't want, and should not use **`mathastext`**.

Let me insist that **`mathastext`** has not been tested in any systematic manner under the Unicode engines; and that it is expected to be most definitely incompatible with unicode-math, although your mileage may vary and some features may appear to work.

When using **`mathastext`** with either XeTEX or LuaTEX it is recommended to use the fontspec package (see remark below on `\encodingdefault`). Furthermore, if using fontspec it is *necessary* to load it with its `no-math` option, and this *must* happen before loading **`mathastext`**.

- Use fontspec with its *no-math* option, and load it *prior* to **`mathastext`**. As some packages load fontspec themselves (for example `polyglossia`), a

  `\PassOptionsToPackage{no-math}{fontspec}`

  early in the preamble might be needed.
- The amsmath package, if used, *must* be loaded *prior* to **`mathastext`**.
- Under `lualatex` engine, it is recommended to also load the package lualatex-math.

I already mentioned in the subsubsection 1.8.2 the fact that the italic corrections were not available for OpenType fonts under the XeTEX engine and only partially available for the LuaTEX engine, with the result that the spacings in math mode when using for the letters an upright text font will be less satisfying than with the standard PDFTEX engine (the OpenType fonts not being usable with the latter engine, this is not a criterion of choice anyhow).

To define math versions when using unicode fonts, use fontspec's `\setmainfont` before the `\Mathastext[`⟨*version*⟩`]` command, or simply before loading **`mathastext`** for the default math versions.

It is possible to mix usage of Unicode fonts and classical TEX fonts. All used 8bits font encoding must have been passed as options to the fontenc package.

1.4.16 The unicodeminus option

For legacy reason, **`mathastext`** uses by default the `EN DASH U+2013` for the minus sign in math mode, if the font is determined to be a "Unicode" font.

---

[32] Will ROBERTSON, et al., *Unicode mathematics with support for XeTeX and LuaTeX* `https://ctan.org/pkg/unicode-math`.

There is now the `unicodeminus` to use rather `MINUS SIGN U+2212`.[33] Check `its documentation` on page 61. (1.3q)

1.4.17 Two examples with OpenType fonts

I include here two examples which compiled successfully (a long time ago!) with XƎLaTeX and LuaLaTeX, the first one on a Linux machine, the second one on a Mac OS X machine.

```
\documentclass{article}
\usepackage[hscale=0.8]{geometry}
\usepackage{multicol}
\usepackage[no-math]{fontspec}
\usepackage{lmodern}
\usepackage[subdued,italic]{mathastext}
\setmainfont[Color=999999]{Verdana}       \Mathastext[Verdana]
\setmainfont[Color=0000FF]{Arial}         \Mathastext[Arial]
\setmainfont[Color=00FF00]{DejaVu Serif} \Mathastext[DejaVu]
\MTDeclareVersion{times}{T1}{ptm}{m}{n}
\setmainfont[Color=FF0000]{Andale Mono}   \Mathastext[Andale]
\begin{document}
\newcommand\TEST[1]{\MTversion{#1}%
\begin{multicols}{2}
\hbox to\columnwidth{\hbox to\columnwidth{\hfil
                $abcdefghijklmnopqrstuvwxyz$\hfil}\kern-2.5em{#1}}
   \centerline{ $ABCDEFGHIJKLMNOPQRSTUVWXYZ$ }
   \centerline{ $0123456789$ }
   \centerline{ $!\,?\,*\,,\,.\,:\,;\,+\,-\,=\,(\,)\,[\,]\,/\,\#\,%
   \$\,\%\,\&\,<\,>\,|\,\{\,\}\,\backslash$ }
\columnbreak
   \centerline{ abcdefghijklmnopqrstuvwxyz }
   \centerline{ ABCDEFGHIJKLMNOPQRSTUVWXYZ }
   \centerline{ 0123456789}
   \centerline{ !\,?\,*\,,\,.\,:\,;\,+\,-\,=\,(\,)\,[\,]\,/\,\#\,%
   \$\,\%\,\&\,<\,>\,|\,\{\,\}\,\char92 }
\end{multicols}}
\begin{multicols}{2}
   \centerline{\textbf{math mode}}
\columnbreak
   \centerline{ \textbf{text} }
\end{multicols}
\TEST{DejaVu}\TEST{Verdana}\TEST{times}\TEST{Andale}
\TEST{Arial}\TEST{bold}\TEST{normal}
\end{document}
```

And now the same thing with fonts available on Mac OS X:

```
\documentclass{article}
\usepackage[hscale=0.8]{geometry}
\usepackage{multicol}
\usepackage[no-math]{fontspec}
```

---

[33] Thanks to Tobias Brink who asked for this feature.

```
\usepackage{lmodern}
\usepackage[subdued,italic]{mathastext}
\setmainfont[Color=FF0000]{Hoefler Text}  \Mathastext[Hoefler]
\setmainfont[Color=336633]{American Typewriter}\Mathastext[Typewriter]
\setmainfont[Color=0000FF]{Herculanum}    \Mathastext[Herculanum]
\setmainfont[Color=FF00FF]{Didot}         \Mathastext[Didot]
\setmainfont[Color=999999]{Comic Sans MS} \Mathastext[Comic]
\begin{document}
   --- copy here the code from the previous example ---
\TEST{Didot}\TEST{Comic}\TEST{normal}\TEST{Herculanum}
\TEST{Hoefler}\TEST{Typewriter}\TEST{bold}
\end{document}
```

1.4.18 Compatibility with other packages

Regarding the namespace of the package: all internal macros use `\mst@` (or `\ifmst@` for TEX conditionals) as prefix. Almost all user commands have `\MT` prefix, the oldest ones may use `\Mathastext` or variants.[34]

Compatibility issues are often questions of who decides last. They are naturally to be expected with packages dealing with the math setting. The fix is simply to load **mathastext** last. In particular one should *always* load **mathastext** *after* amsmath (this is especially needed with Unicode engines but applies in general as well).

Any definition made in a package loaded before **mathastext** of the font to be used for letters or for the common characters in the `ascii` basic range will be overruled by the loading of **mathastext**. Conversely most of the set-up done by **mathastext** may well be overruled by packages loaded later which do math related things.

Starting with version `1.2`, **mathastext** makes some characters 'mathematically active' to achieve certain effects: automatic insertion of the italic corrections when using an upright text font in math (subsubsection 1.8.2), extended scope of the math alphabet commands which now apply to non-letter symbols (subsubsection 1.8.4; and also to math operator names, but this is much easier to achieve). And the (already mathematically active) right quote is modified to have some extra space added before the derivative glyph $'$ (see `\MTprimeskip`).

This mathematical activation is compatible with the `\label` and `\ref` commands in and outside of math mode.

But a difficulty arises when some other package has made the character 'catcode active' everywhere in the document. If it is detected for such a character that **mathastext** wishes to make 'mathematically active' that it is already 'catcode active', **mathastext** then checks if it is handling a Babel shorthand. If yes, it then hacks into babel support macros for that character to let it do what it desires it to

---

[34] It is only years after initial release that I became aware that package mathtools used `\MT_` prefix (with an underscore as shown) for most internal constructs, and sometimes `\MT@`, and those can not clash with our `\MT[a-z|A-Z]` for public commands and `\mst@` for internal macros. They do have however user commands `\MTFlushSpaceAbove` and `\MTFlushSpaceBelow`, but nothing else hence some retroactive relief.

do in math mode. And it does not make the character mathematically active, to the contrary it makes sure that the character is *not* mathematically active.

This last paragraph applies to the characters ; , : ! ? + − = < > ( ) [ ] * mentioned in subsubsection 1.8.4 as 'hard non-letters'. The right tick (which is already mathematically active per default) is also handled via a similar process, and in particular is tested for being a Babel shorthand (which happens in particular with the Spanish language `activeacute` option).

The reason for avoiding in general for a catcode active character to be at same time mathematically active is illustrated in next shaded box.

In the case of the Spanish active `'`, the advantage of **`mathastext`** intervention to reset its mathcode is that in case of a faulty input as a curly right tick `U+8217` ', which LaTeX will map to `\textquoteright` which itself gives the catcode 12 `'`, it will show as curly quote in output not as prime glyph, as **`mathastext`** removes its mathematical activation. This complements the LaTeX warnings about `\textquoteright` being invalid in math mode.

The ascii letters can not have been made active by Babel interface (see code comments of `\mst@do@activecase`) so if an ascii letter is catcode active when **`mathastext`** examines it, nothing will change to its meaning.

`changed:` At `1.4` these actions are done for ascii letters at loading of the package or via `\MTmathactiveletters`, which is itself done by `\MTversion` when entering non-subdued math version. Check documentation of `\MTmathactiveletters` for some relevant details. `(1.4)`

On matters of *mathematical* versus *catcode* active character tokens, here is some code, not involving **`mathastext`**, that you should **not** try at home:

```
\documentclass{article}
\usepackage[french]{babel}
\usepackage{mathtools}\mathtoolsset{centercolon}
\begin{document}
$:$
\end{document}
```

DO NOT DO THIS AT HOME : it creates (with `pdflatex`) an infinite loop. This is due to the fact that the colon is simultaneously active (this is made by babel-french at begin document) and mathematically active (done by mathtools in the preamble). The interaction gives an infinite loop.

Regarding mathematical activation, an incompatibility of another type arose with amsmath. To fix it, **`mathastext`** now replaces an inner macro of amsmath (`\resetMathstrut@`) with its own version.

Always load amsmath before **`mathastext`**.

Actually this last commandment was already made necessary by the use of the text endash to represent the minus sign in math mode, and, especially for Unicode engines, some aspects of the `\DeclareMathOperator` macro from amsmath.

New with 1.3i: **`mathastext`** patches `\url` of packages url and hyperref, and also `\nolinkurl`, to force them to do automatically `\MTeverymathoff`. Indeed they use math mode, and it is better to turn **`mathastext`** off for their dealings.

See the `\MTeverymathoff` documentation.

1.5 Math alphabets

Let us first recall fundamental facts of life, in the world of traditional PDFTEX engine and TEX fonts, as background for understanding what **`mathastext`** does in this context. People familiar with using Unicode engines and unicode-math, please be aware that the semantics there of the LATEX math alphabet commands are **significantly** modified!

- In the default LATEX set-up all five of `\mathrm`, `\mathbf`, `\mathit`, `\mathsf` and `\mathtt` tell TEX to use for their arguments specific `OT1`-encoded fonts.
- If the document body uses, as will be the case probably with any language other than English or its variants, some other encoding such as `T1` for its fonts, there is no change whatsoever to the math configuration, indeed most font packages ignore it completely.
- It is thus a priori wrong to think of these commands as switching to some body text font, although letters within their scopes will act as in a text font, and in particular obey ligatures (this also applies to operator names defined by `amsmath`'s `\DeclareMathOperator` which are, with some extras not mentioned here, as using `\mathrm`).
- These commands are completely different in spirit from the LATEX `\textrm`, `\textbf`, and others, which change only some font axis; indeed the math alphabet commands inherit from legacy Knuth's `\rm`, `\bf`, and others which are complete font specifiers.
- In particular when nesting, it is the inner-most which wins.
- Only mathematical characters (such as letters) which are declared to TEX as being of "variable family type" react to being in the argument of a math alphabet command.
- Lowercase Greek letters are by default in LATEX immune to math alphabets (so `\mathrm{\pi}` induces no change in output), but the eleven uppercase Greek letters are of "variable family type" because they are picked in the `OT1`-encoded font also used for operator names (the one to which `\mathrm` maps), *and they occupy the exact same slots in the* `OML`*-encoding to which* `\mathnormal` maps! So in default TEX, `\mathnormal{\Gamma}` gives a slanted glyph. The slots occupied in `OML`-encoding by the lowercase Greek letters (to which encoding they are a priori assigned) give completely unrelated glyphs in the `OT1`-encoding, so it makes sense that the default LATEX declares lowercase Greek to not react to math alphabets. Notice

though, that if LaTeX had declared a `\mathnormalbold`, mapping to a bold `OML`-encoded font, it would have made sense to also have `\alpha`, `\beta`, etc... be of "variable family type".

- But of course even if `\pi` is of such "variable family type" then `\mathbf{\pi}` will give garbage because the default `\mathbf` selects an `OT1`-encoded font, where there is no pi glyph whatsoever and in particular not at the slot (which is 25) of π in the `OML` encoding!

Please keep all the above in mind when trying to understand what **mathastext** does with math alphabets. The most significant point described next naturally is that **mathastext** will sync `\mathnormal`, `\mathrm`, `\mathbf`, `\mathit`, `\mathsf` and `\mathtt` to map to the **mathastext**-ified body text fonts.

- `\mathnormal`, `\mathrm`, `\mathbf`, `\mathit`, `\mathsf` and `\mathtt` are modified to use the **mathastext**-ified text fonts; this can be disabled via `defaultalphabets` and related individual options, but the package always provides `\Mathnormal`, `\Mathrm`, etc..., to match the **mathastext** font configuration (prior to `1.3za` `defaultalphabets` and related options also caused the **mathastext** alphabet commands not to be defined). Recall that there may arise a "too many math alphabets" error if too many of these commands are *used* in the document: *declaring* them is not by itself the cause of the error. See the LaTeX news entry of its `2021-11-15` release for the counter `localmathalphabets` (with default value 2) which can be now be used if one hits such a difficulty. **changed:** (1.3za)
- We define a new math alphabet command `\mathnormalbold` which gives direct access to the bold version of the `\mathnormal` alphabet (rather than using either the `\bm` command from the bm package or the `\boldsymbol` command from the amsbsy package). As it does not exist in the default LaTeX math font set-up, this alphabet is *not* subjected to the subdued option action.
- Version `1.2` of **mathastext** has extended the scope of the math alphabets to apply to non-alphabetical characters and to operator names. This respects the automatic white spaces added by TeX around math symbols. See the devoted subsubsection 1.8.4. (1.2)
- The optional extra skips around letters (see subsubsection 1.8.1 and subsubsection 1.8.2) are removed in the scope of the math alphabets. (1.3i)
- Depending on options, further math alphabet commands are defined by the package: `\MathEulerBold`, `\MathEuler`, `\MathPSymbol`, and since `1.3y` under the `LGRgreek` family of options also `\mathgreekup` and `\mathgreekit`. (1.3y)

See subsubsection 1.7.3. And also `\mathgreekupbold` and `\mathgreekitbold` under the `LGRgreek` family of options. (1.3za)

- With the `LGRgreek+` option which enhances Greek letters with a specific behavior when in the arguments of the `\mathrm`, `\mathbf`, etc..., math alphabet commands, this special behavior is not triggered by the `\Mathrm`, `\Mathbf`, et al., which are genuine unmodifed math alphabet commands. See subsubsection 1.7.5. (1.3za)

1.6 Math versions

LaTeX has the concept of *math versions*[35], but most font packages do not define any such version beyond the default normal and bold (the package unicode-math for unicode engines does use this concept).

mathastext extends the concept of math versions in order to allow the math mode fonts (for letters, digits, punctuation and a few other ascii symbols) used in the different parts of the document to be kept in sync with the text fonts.

Most math symbols (sums, products, integrals, logical signs, etc. . .) are kept the same throughout the document though as it is not in **mathastext** power to modify them.

For examples see the earlier subsection 1.2. The interface to define a `math version` includes the commands `\Mathastext` and `\MTDeclareVersion`.

Once such a `math versions` has been defined in the preamble, `\MTversion{name_of_version}`, or equivalently `\Mathastextversion{name_of_version}`, enacts the font switches when encountered in the body of the document. As is usual with LaTeX one can limit the scope to inside a group, or also switch back to the main set-up via `\Mathastextversion{normal}`.

When `\Mathastext` is used in the preamble, it records the current text font defaults (`\familydefault` et al. or what has been configured by `\MTfamily` and similar commands) and (except for the `normal` and `bold` versions if in `subdued` regime) sets up *both* the math font and the text font in the defined **mathastext**-math version to be this text font. It is still possible to switch on via `\MTversion` in the document body distinct fonts for text and math: an optional argument (the name of another **mathastext**-declared math version) to `\MTversion` is allowed (such as for example `\MTversion[newcent]{helvet}` for one of the examples of the subsection 1.2). It instructs to use as text font the font which was configured to be used in this second **mathastext**-math version.[36]

---

[35] `math versions` are discussed in the document `fntguide.pdf` from your TeX distribution.

[36] When not using math versions at all (so not using `subdued` mode either) another way to achieve distinct fonts in text and math is naturally to modify the document text font *after* having loaded **mathastext** (or after last usage of `\Mathastext` without optional argument). Another way is to use `\MTfamily`, `\MTencoding`, `\MTseries`, `\MTshape`, `\MTlettershape` in the preamble before a call to `\Mathastext` which will configure math fonts without having modified the document text fonts. However if one does `\MTversion{normal}` in the document then the text font will be reset to what was recorded as math font by the `\Mathastext` call in the preamble (as said above, when

The native LaTeX command `\mathversion{⟨version_name⟩}` would change only the fonts for the math mode, not the text mode fonts. It is important to use rather the package command `\MTversion` (or one of its synonyms `\mathastextversion`, `\Mathastextversion`, `\MTVersion`), with its mandatory argument `{⟨version_name⟩}`, as it does additional actions:

- it sets the font for math mode (letters, math operator names, digits, punctuations, some other symbols) according to the version name given as mandatory argument,
- it resets the text font of the document and the `\(family,rm,sf,...)defaults` to their values as registered at the time of definition of the version. *Use the starred variant in case this is not desired.* As explained above tt is possible to specify within brackets an extra optional version name, and the text font will be set according to it.

For all math versions if not using the **subdued** option, or only for the non-*normal* and non-*bold* math versions if using the **subdued** option, `\MTversion` does further additional tasks:

- it resets the `\hbar`, `\imath` (see `\inodot`), `\jmath`, math accents (see option **mathaccents**) and minus sign as en dash according to the used font encoding for the **mathastext**-ified text font, (1.3u)
- (see subsubsection 1.8.1 and subsubsection 1.8.2) it re-issues the command `\MTmathactiveletters` to let a to z, A to Z, be mathematically active in order to automatically insert the skips as defined by the user with `\MTsetmathskips`, and the italic corrections (if the font is not italic or slanted),
- (see subsubsection 1.8.3) it resets the extra spaces after the symbols $\exists$, $\forall$ and before the derivative $'$ to the values as decided by the user in the preamble on a *per version* basis,
- (see subsubsection 1.8.4) it re-issues the commands `\MTmathoperatorsobeymathxx` and `\MTeasynonlettersobeymathxx` to let the math operator names and (‘easy’) non letter characters obey the math alphabets,
- in case of option **asterisk**, it re-issues `\MTactiveasterisk`,
- it does the additional set-up for Greek letters in case of the package received one of the Greek related options.

The scope is limited to the current LaTeX environment or group.

It is sometimes not compatible with **mathastext** to load a font package after it, as the font package may contain instructions which will modify the math set-up.

---

not using subdued option).

This may be a bit hidden to the user: for example the epigrafica package loads `pxfonts`. Hence it will interfere with **`mathastext`** if it is loaded after it.[37] But one can use instead `\renewcommand{\rmdefault}{epigrafica}`,[38] followed with `\Mathastext`, or also `\MTfamily{epigrafica}\Mathastext` which will only change the font in math.

To use `epigrafica` for Greek in math mode one can use the package option `LGRgreek` and the command `\MTgreekfont{epigrafica}\Mathastext`. Or `\usepackage{epigrafica}` followed with `\usepackage[LGRgreek]{mathastext}`.

1.7 Greek letters

1.7.1 The Greek-related options

The Computer Modern fonts are very light and thin in comparison to many text fonts, and as a result rarely mix well with them (particularly if the Latin letters in math mode are upright). The following options are provided by **`mathastext`**:

no option: nothing is done by the package, Greek letters are the default Computer Modern ones or have been set-up by other packages; for example by the fourier package with option 'upright', which gives upright Greek letters or by the author lgrmath package.

`LGRgreek`: (this was substantially updated at `1.3y`, make sure to read the new documentation at subsubsection 1.7.3) this option is for fonts which additionally to Latin letters also provide Greek letters in `LGR` encoding. Here is a list from a 2012 standard TeX installation: the Computer Modern, Latin Modern, and the CM-LGC fonts; the Greek Font Society fonts (such as GFS Didot), the epigrafica and kerkis packages, the txfontsb package which extends the txfonts package with LGR-encoded Greek letters; the Droid fonts, the DejaVu fonts, the Comfortaa font, and the Open Sans font. The `LGR` encoded `CM/LM` fonts (in serif, sans-serif and typewriter family) give the nice Greek letters in upright shape from the cbfonts package. To get these letters in your **`mathastext`** math mode, you can do the following:

```
% instructions to load the document fonts:
\usepackage{nice_font}
% and then the following:
\renewcommand{\familydefault}{cmr} % or cmss or cmtt for sans resp. mono
\usepackage[LGRgreek]{mathastext}
\renewcommand{\familydefault}{\rmdefault}
\Mathastext % this re-initializes mathastext with the nice_font,
```

---

[37] may typically give a 'too many math alphabets' error message.

[38] sometimes one needs to look in the `.sty` file of the font package to figure out the font name (it is rarely as here with epigrafica, the same as the package name), and, if one does not know the arcanes of finding `.fd` files in one's TeX distribution, one should look at the log file of a test document to see if for example `T1` is available for that font; for `epigrafica` it is not, only `OT1` and `LGR` are possible.

```
% without changing the LGR font cmr/cmss/cmtt used for Greek letters
% in math mode.
\begin{document}
```

If you use the `italic` option note that the italic Greek letters from the `cbfonts` are not the same glyphs as the default Greek letters from the `OML` encoded font `cmmi`.

`LGRgreek+`: extends `LGRgreek` to allow abusive usage of `\mathrm` and alike commands with Greek letters. This is very much not in the spirit (especially with traditional "8bit" TEX fonts) of the LATEX kernel concept of math alphabet commands. Check subsubsection 1.7.5 for relevant information. (1.3za)

`eulergreek`: the Greek letters will be taken from the Euler font (the document does not have to load the eulervm package, **`mathastext`** directly uses some file included in this package, as it provides a mechanism to scale by an arbitrary factor the Euler font.) The letters are upright.

`symbolgreek`: the Greek letters will be taken from the (Adobe Postscript) Symbol font. A command is provided so that the user can scale the Symbol font to let it better fit with the text font. The letters are upright.

`selfGreek`: this option concerns only the eleven Greek capitals from the `OT1`-encoding. It does nothing for the lowercase Greek letters. The encoding used in the document does not have to be `OT1`.

There is also `LGRgreeks` (and `LGRgreeks+`) which tells **`mathastext`** to pick up in each math version the letters from the `LGR` encoded font used in that version, and `selfGreeks` to tell **`mathastext`** to do as for `selfGreek` but separately in all math versions.

Under the `subdued` option the Greek letters in the normal and bold math versions are kept to their defaults as found at the time of loading the package.

The commands `\MTstandardgreek` allow at any point in the document to turn inactive any Greek related option passed to **`mathastext`**. And conversely `\MTcustomgreek` reactivates it.

1.7.2 Shape of Greek letters

Classic TEX uses in math mode italic lowercase and upright uppercase Greek letters. French typography uses upright shape for both lowercase and uppercase. And the ISO standard is to use italic shape for both lowercase and uppercase.

The Euler and Symbol fonts not being available in other than their default upright shape, this question of shapes for Greek letters raises issues only in the case of the options `LGRgreek` and `selfGreek`.

The options `frenchmath`, `itgreek`, `upgreek`, `itGreek` and `upGreek` modify the Greek letter shapes according to the following rules, listed from the lowest to the highest priority:

no option: the lowercase Greek letters are in the same shape as Latin letters, and the uppercase in the same shape as is applied to digits and operator names,

frenchmath: both lowercase and uppercase are in the same shape as the digits and operator names (most of the time this means "upright shape", but it can be otherwise),

itgreek : says that Greek letters (both lowercase and uppercase) will be in '`it`'
changed: shape. More precisely the expansion of `\MTgreekitdefault` is used. (1.3y)

This was changed at 1.3y, formerly the value of `\itdefault` which was in force at the time of using `\Mathastext` (or at time of loading the package if no use is made of `\Mathastext`) was used.

upgreek : says that Greek letters (both lowercase and uppercase) will be in '`n`'
changed: shape. More precisely the expansion of `\MTgreekupdefault` is used. (1.3y)

This was changed at 1.3y, formerly the value of `\updefault` which was in force at the time of using `\Mathastext` (or at time of loading the package if no use is made of `\Mathastext`) was used. But since LaTeX 2020-02-02 this caused many Font Warnings in the log because `\updefault` is now '`up`', not '`n`' as formerly.

itGreek, upGreek: same but they apply only to the uppercase Greek letters. Their effect is computed after having taken into account either `itgreek` or `upgreek` presence.

So, the default gives the classic TeX behavior when option `italic` was passed.

As mentioned already the package allows to define various "math versions". There are commands to be used inside the preamble to influence the shapes, and even the font, used for Greek letters in each given **mathastext**-declared math version: `\MTitgreek`, `\MTupgreek`, `\MTitGreek`, `\MTupGreek` and `\MTgreekfont{name_of_font}`.

Their effect is as the options of the alike name, except that the effect applies only to **mathastext**-math versions declared *next* in the preamble (be it via `\Mathastext` or `\MTDeclareVersion`).

To use `\MTgreekfont` you need to know the name of a suitable font family available in `LGR` encoding: for example `lmr`, `txr` (needs txfontsb package on your system), `DejaVuSerif-TLF` (needs dejavu package on your system), etc...

`\MTitgreek`, `\MTupgreek`, `\MTitGreek`, `\MTupGreek` have some effect only if one of the `LGRgreek`, `LGRgreeks`, `selfGreek` or `selfGreeks` options was passed to the package.

Once any of these commands has been made use of, changes in the shape configuration of the Latin letters will stop having any influence on the shape of the Greek letters.

`\MTgreekfont` has an effect only for `LGRgreek` and `selfGreek`. It is without any effect with `LGRgreeks` and `selfGreeks`.

1.7.3 Control sequences to access directly upright or italic shape for Greek under `LGRgreek` option

Some changes were made at `1.3y` to enhance the `LGRgreek` (and `LGRgreeks`) options with new features. Everything which will be explained here applies only to these two options.

First of all the package now makes available control sequences to access either the upright or italic shape of the Greek letters: `\alphaup`, `\alphait`, etc...[39] Which shape is meant by 'up' or 'it' is configured via defining `\MTgreekupdefault` and `\MTgreekitdefault` respectively prior a `\Mathastext` command in the preamble (possibly with `[`⟨*version_name*⟩`]` optional argument). Their default definitions are to expand to '`n`' and '`it`' respectively. They can also be defined prior to loading **`mathastext`**.

See the Table 1 and Table 2 for illustrations (using here the Libertinus Serif font). (`1.3y`)

\Alphaup Α	\Xiup Ξ	\alphaup α	\xiup ξ
\Betaup Β	\Omicronup Ο	\betaup β	\omicronup ο
\Gammaup Γ	\Piup Π	\gammaup γ	\piup π
\Deltaup Δ	\Rhoup Ρ	\deltaup δ	\rhoup ρ
\Epsilonup Ε	\Sigmaup Σ	\epsilonup ε	\sigmaup σ
\Zetaup Ζ	\Tauup Τ	\zetaup ζ	\tauup τ
\Etaup Η	\Upsilonup Υ	\etaup η	\upsilonup υ
\Thetaup Θ	\Phiup Φ	\thetaup θ	\phiup ϕ
\Iotaup Ι	\Chiup Χ	\iotaup ι	\chiup χ
\Kappaup Κ	\Psiup Ψ	\kappaup κ	\psiup ψ
\Lambdaup Λ	\Omegaup Ω	\lambdaup λ	\omegaup ω
\Muup Μ	\Digammaup Ϝ	\muup μ	\digammaup ϝ
\Nuup Ν		\nuup ν	\varsigmaup ς

Table 1: Greek letters via 'up' control sequences (math mode only)

The regular control sequences without '`up`' or '`it`' postfix will map to either one of the two versions according to how the shapes were configured, i.e. in almost all

---

[39] No check is done of pre-existing such math symbol, they will be replaced by the **`mathastext`** definition with no warning. If they happen to be pre-defined as LaTeX commands, not as math symbols, errors will happen during the loading of **`mathastext`**.

\Alphait A
\Betait B
\Gammait $\varGamma$
\Deltait $\varDelta$
\Epsilonit E
\Zetait Z
\Etait H
\Thetait $\varTheta$
\Iotait I
\Kappait K
\Lambdait $\varLambda$
\Muit M
\Nuit N

\Xiit $\varXi$
\Omicronit O
\Piit $\varPi$
\Rhoit P
\Sigmait $\varSigma$
\Tauit T
\Upsilonit $\varUpsilon$
\Phiit $\varPhi$
\Chiit X
\Psiit $\varPsi$
\Omegait $\varOmega$
\Digammait F

\alphait α
\betait β
\gammait γ
\deltait δ
\epsilonit ε
\zetait ζ
\etait η
\thetait θ
\iotait ι
\kappait κ
\lambdait λ
\muit μ
\nuit ν

\xiit ξ
\omicronit o
\piit π
\rhoit ρ
\sigmait σ
\tauit τ
\upsilonit υ
\phiit ϕ
\chiit χ
\psiit ψ
\omegait ω
\digammait $\digamma$
\varsigmait ς

Table 2: Greek letters via ‘it’ control sequences (math mode only)

cases via usage of either the `itgreek`, `upgreek`, etc... options or `\MTitgreek` et al. commands. This is on a per **`mathastext`**-enriched math version basis, depending only on how the options or commands were used in the preamble.

Furthermore two math alphabets are provided (1.3y)

`\mathgreekup`
`\mathgreekit`

which can be used to map a letter to the corresponding Greek fonts:

`$\mathgreekup{a}=\mathgreekup{\alpha}=\mathgreekup{\alphait}=\alphaup$`

$$\upalpha = \upalpha = \upalpha = \upalpha$$

`$\mathgreekup{G}=\mathgreekup{\Gamma}=\mathgreekup{\Gammait}=\Gammaup$`

$$\Gamma = \Gamma = \Gamma = \Gamma$$

`$\mathgreekit{z}=\mathgreekit{\zeta}=\mathgreekit{\zetaup}=\zetait$`

$$\zeta = \zeta = \zeta = \zeta$$

`$\mathgreekit{W}=\mathgreekit{\Omega}=\mathgreekit{\Omegaup}=\Omegait$`

$$\varOmega = \varOmega = \varOmega = \varOmega$$

Some refactoring[40] was required to achieve this at `1.3y` and it is not 100% backwards compatible: if none of the `itgreek` etc... things was used, the Greek letters formerly would follow the shape of Latin letters (for lowercase Greek) and of operator names (for uppercase Greek). Now, some check is made for each of these two shapes whether it is ‘`it`’ or ‘`sl`’ and then the ‘italic’ shape, i.e. `\MTgreekitdefault` which by default is ‘`it`’ (without the quotes) is used, else the ‘upright’ shape, i.e. `\MTgreekupdefault` which by default expands to ‘`n`’ (without the quotes) is used. (1.3y)

---

[40] Technically, formerly two symbol fonts were declared, one for the lowercase Greek letters and one for the uppercase Greek letters; now those are dropped and replaced by two symbol fonts, one for ‘italic’ Greek letters, the other for ‘upright’ Greek letters.

Naturally these checks are done on a per mathastext-math version basis, if multiple math versions are used.

So for example those who used some adventurous '`sc`' for the main shape (the one used per default for operator names) and used the option `LGRgreek` but none of the `itgreek` et al. options, and none of the `\MTitgreek` et al. commands, now will need to adjust `\MTgreekupdefault` to expand to '`sc`' prior to some `\Mathastext` or `\Mathastext`[⟨*version_name*⟩] or `\MTDeclareVersion` in the preamble depending on context.

It is hoped most documents, even those using multiple math versions, which made use of the `LGRgreek` (or `LGRgreeks`) option will simply produce unmodified output. Please report to the author unexpected results not fitting the above attempted description of the only partial backwards compatibility.

1.7.4 `\mathgreekupbold` and `\mathgreekitbold`

Again this applies only to `LGRgreek` and `LGRgreeks` options (and the `1.3za` added `LGRgreek+` and `LGRgreeks+`).

See the Table 3 and Table 4 for illustration of usage (in math mode only) of code such as (1.3za)

```
    \mathgreekupbold{\alpha}
or  \mathgreekitbold{\alpha}
```

Note that all three of `\alpha`, `\alphaup` and `\alphait` would give the same output. These two tables again use the Libertinus Serif font via an mathastext math version which was configured in the preamble using this set-up (and the package `LGRgreek` option):

```
\MTfamily{LibertinusSerif-TLF}
\MTlettershape{n}
\MTseries{m}
\MTgreekfont{LibertinusSerif-TLF}
\MTupgreek
\Mathastext[libertinus]
```

Some examples here to illustrate the effet of the math alphabet commands on Latin letters also:

`$\mathgreekupbold{a}=\mathgreekupbold{\alpha}$`

$$\boldsymbol{\alpha} = \boldsymbol{\alpha}$$

`$\mathgreekupbold{G}=\mathgreekupbold{\Gamma}$`

$$\mathbf{\Gamma} = \mathbf{\Gamma}$$

`$\mathgreekitbold{z}=\mathgreekitbold{\zeta}$`

$$\boldsymbol{\zeta} = \boldsymbol{\zeta}$$

`$\mathgreekitbold{W}=\mathgreekitbold{\Omega}$`

$$\boldsymbol{\Omega} = \boldsymbol{\Omega}$$

\Alpha → **Α**
\Beta → **Β**
\Gamma → **Γ**
\Delta → **Δ**
\Epsilon → **Ε**
\Zeta → **Ζ**
\Eta → **Η**
\Theta → **Θ**
\Iota → **Ι**
\Kappa → **Κ**
\Lambda → **Λ**
\Mu → **Μ**
\Nu → **Ν**

\Xi → **Ξ**
\Omicron → **Ο**
\Pi → **Π**
\Rho → **Ρ**
\Sigma → **Σ**
\Tau → **Τ**
\Upsilon → **Υ**
\Phi → **Φ**
\Chi → **Χ**
\Psi → **Ψ**
\Omega → **Ω**
\Digamma → **Ϝ**

\alpha → **α**
\beta → **β**
\gamma → **γ**
\delta → **δ**
\epsilon → **ε**
\zeta → **ζ**
\eta → **η**
\theta → **θ**
\iota → **ι**
\kappa → **κ**
\lambda → **λ**
\mu → **μ**
\nu → **ν**

\xi → **ξ**
\omicron → **ο**
\pi → **π**
\rho → **ρ**
\sigma → **σ**
\tau → **τ**
\upsilon → **υ**
\phi → **φ**
\chi → **χ**
\psi → **ψ**
\omega → **ω**
\digamma → **ϝ**
\varsigma → **ς**

Table 3: Greek control sequences in the argument of `\mathgreekupbold`.

\Alpha → ***Α***
\Beta → ***Β***
\Gamma → ***Γ***
\Delta → ***Δ***
\Epsilon → ***Ε***
\Zeta → ***Ζ***
\Eta → ***Η***
\Theta → ***Θ***
\Iota → ***Ι***
\Kappa → ***Κ***
\Lambda → ***Λ***
\Mu → ***Μ***
\Nu → ***Ν***

\Xi → ***Ξ***
\Omicron → ***Ο***
\Pi → ***Π***
\Rho → ***Ρ***
\Sigma → ***Σ***
\Tau → ***Τ***
\Upsilon → ***Υ***
\Phi → ***Φ***
\Chi → ***Χ***
\Psi → ***Ψ***
\Omega → ***Ω***
\Digamma → N/A

\alpha → ***α***
\beta → ***β***
\gamma → ***γ***
\delta → ***δ***
\epsilon → ***ε***
\zeta → ***ζ***
\eta → ***η***
\theta → ***θ***
\iota → ***ι***
\kappa → ***κ***
\lambda → ***λ***
\mu → ***μ***
\nu → ***ν***

\xi → ***ξ***
\omicron → ***ο***
\pi → ***π***
\rho → ***ρ***
\sigma → ***σ***
\tau → ***τ***
\upsilon → ***υ***
\phi → ***φ***
\chi → ***χ***
\psi → ***ψ***
\omega → ***ω***
\digamma → N/A
\varsigma → ***ς***

Table 4: Greek control sequences in the argument of the `\mathgreekitbold` command. This font has no bold italic Digamma nor digamma (last tested 2023/12/19).

1.7.5 Special behavior of `\mathrm`, `\mathbf`, `\mathit` with Greek letters via the `LGRgreek+` option

With option `LGRgreek+` or `LGRgreeks+`, **`mathastext`** makes Greek letters control (1.3za) sequences `\alpha`, `\beta`, ... (but not `\alphaup` or `\betait` and the others) react in a special manner within the scope of `\mathnormal`, `\mathrm`, `\mathit`, `\mathbf`, and `\mathnormalbold`, but not further math alphabet commands, and not when using the **`mathastext`** defined commands named with an uppercased initial.

Here is an example

$$\begin{array}{rc}
 & abCD\alpha\pi\Delta\Gamma \\
\mathit{mathnormal} & abCD\alpha\pi\Delta\Gamma \\
\mathit{mathrm} & \mathrm{abCD}\text{απΔΓ} \\
\mathit{mathit} & \mathit{abCD}\textit{απΔΓ} \\
\mathit{mathbf} & \mathbf{abCD}\textbf{απΔΓ} \\
\mathit{mathnormalbold} & \boldsymbol{abCD\alpha\pi}\mathbf{\Delta\Gamma} \\
\mathit{mathgreekup} & \text{αβ῾ΔαπΔΓ} \\
\mathit{mathgreekit} & \textit{αβ῾ΔαπΔΓ} \\
\mathit{mathgreekupbold} & \textbf{αβ῾ΔαπΔΓ} \\
\mathit{mathgreekitbold} & \textbf{\textit{αβ῾ΔαπΔΓ}}
\end{array}$$

It used this source:

```
\[\def\zzz{abCD\alpha\pi\Delta\Gamma}
  \begin{array}{rc}
    &\zzz\\
    mathnormal& \mathnormal{\zzz}\\
    mathrm& \mathrm{\zzz}\\% \mathrm on Greek is like \mathgreekup
    mathit& \mathit{\zzz}\\% \mathit on Greek is like \mathgreekit
    mathbf& \mathbf{\zzz}\\% \mathbf on Greek is like \mathgreekupbold
    mathnormalbold& \mathnormalbold{\zzz}\\
    mathgreekup& \mathgreekup{\zzz}\\%
    mathgreekit& \mathgreekit{\zzz}\\%
    mathgreekupbold& \mathgreekupbold{\zzz}\\%
    mathgreekitbold& \mathgreekitbold{\zzz}%
  \end{array}
\]
```

This was typeset here using a “libertinustexstyle” math version which (differently from the one used in an earlier section) has the default TEX settings for the shape of Latin and Greek letters: i.e. italic Latin and lowercase Greek, upright uppercase Greek. Its preamble definition was something like this:

```
\MTfamily{LibertinusSerif-TLF}
\MTgreekfont{LibertinusSerif-TLF}
\MTlettershape{it}% not needed with italic option if nothing was changed prior
\MTitgreek\MTupGreek% this is also the default configuration
\Mathastext[libertinustexstyle]
```

The difference with using only `LGRgreek` option is that with the latter the Latin math alphabets such as `\mathrm`, `\mathit`, `\mathbf` produce a Latin letter when acting on a Greek control sequence, as the latter are defined by **mathastext** under `LGRgreek` to be of “variable family type” for usage with `\mathgreekup` and `\mathgreekit`. With `LGRgreek+`, the Greek control sequences are not mathchar tokens anymore but macros with conditionals detecting some flag set by custom `\mathnormal`, `\mathnormalbold`, `\mathrm`, `\mathit`, and `\mathbf`.

mathastext has no logical way to sync shape of Latin and Greek letters once usage has been made of Greek related commands. Hence `\mathnormal` is currently configured to do nothing on Greek letters. This may change, please consider this behavior unstable.

Remark: this `LGRgreek+` functionality is considered by its author an abuse of the concept of a math alphabet command and required accomodating a serious deviation from internal logical design of **mathastext**. I don’t know if it is because LaTeX documentations are deficient or misleading on such matters but it appears many LaTeX users are surprised when `$\mathrm{\pi}$` does not give an upright pi letter but this is completely to be expected in a world with fonts having only 128 or 256 glyphs, and from the fact that `\mathrm` and `\mathbf` originate in Plain TeX `\rm` and `\bf` and are still quite akin to it, they are font switching commands nothing more or less.

1.8 Advanced capacities

Some such capacities are on per default (but if with `subdued` option will be turned off in the *normal* and *bold* math versions), others require an action from the user for activation.

1.8.1 Extra spaces around letters

This is a new feature[41] added with release `1.3`: the command `\MTsetmathskips` allows the user to set up some spaces (more precisely, ‘mu glue’; but stretch and shrink are discarded) to be automatically inserted around the letters in math mode. Some (very) unrealistic uses:

```
% this may be anywhere in the document (also within a math group):
\MTsetmathskips{x}{20.33mu}{15.66mu}% 20.33mu before all x's and 15.66mu after.
\MTsetmathskips{y}{\thickmuskip}{\thickmuskip}%
\MTsetmathskips{z}{10mu}{5mu}% stretch and shrink are anyhow without effect.
\MTsetmathskips{A}{\muexpr \thickmuskip*2}{\muexpr \medmuskip-\thinmuskip/2}%
```

Here is what `$wxtytz^{wxtytz}=BAC^{BAC}$` then gives using the Times font: $w\quad x\quad t\;y\;t\;z^{w\quad x\quad t\;y\;t\;z} = B\quad A\,C^{B\quad A\,C}$. Any TeX group or LaTeX environment limits as usual the scope of this command. Furthermore the command `\MTunsetmathskips` cancels previous use of `\MTsetmathskips` for a given letter.

---

[41] It was new in 2013 indeed. Not so much new now, but it is never too late to try it out.

The implementation relies on the 'mathematical activation' of letters, which is done by default by the package since release `1.2b`. Should this cause compatibility problems, the command `\MTmathstandardletters` cancels it entirely. To reactivate it, there is `\MTmathactiveletters`. Note that `\MTmathactiveletters` is done automatically (as part of `\MTicinmath`) by **`mathastext`** when loaded (if not with `subdued` option), and also each time the package enhanced math-version-switch command `\MTversion` is used, except for the normal and bold math versions under the `subdued` option.

The extra skips are set at natural width; they do not contribute to the overall stretchability or shrinkability of the math formula and do not create break points.

Changed with `1.3i`: they are *not* applied within the scope of math alphabet commands.

1.8.2 Background on italic corrections in math mode

Note: this is somewhat technical discussion which may well be skipped in its entirety on first reading.

With the `italic` option the letters in math will be generally in italic shape (and, normally, upright in operator names).

For the built-in placement routines of TeX in math mode to work as well as they usually do, the characters from the math italic font obviously should have their bounding boxes wide enough for the glyphs not to collide with other symbols. A letter from a text italic font such as *f* extends way out of its declared bounding box; let us compare the bounding boxes[42] for the letter f in the math italic font to the one from the text italic font: $\boxed{f}$ vs. $\boxed{\textit{f}}$.

This could make us think that attempting to use in math a text italic font will lead to disaster. Well, surprisingly the situation is not that bad. Sure `$f(x)$` is wider with the standard math italic $\boxed{f(x)}$ (`21.31474pt`) than it is with the text italic font used in math:[43] $\boxed{\mathit{f(x)}}$ (`19.74986pt`) but we should be surprised that our text italic *f* did not end up even closer to the opening parenthesis. Why is it so?

The explanation is that TeX uses in such a situation the *italic correction* for the letter *f*. The italic correction also exists and is used for the math italic font, it was inserted in `$f$` without us having to ask anything. Its value is `1.17865pt` for the math italic f and `1.8919pt` for the text italic *f*.[44] With the italic corrections included our bounding boxes are indeed more alike: $\boxed{f}$ vs $\boxed{\textit{f}}$.

Without the italic corrections[45] it is $\boxed{f}$ vs $\boxed{\textit{f}}$. I said that `$f$` included the italic

---

[42] let's be honest, we are lying here about what exactly the first of these is bounding; this is explained later!

[43] we used simply `$\mathit{f(x)}$`.

[44] these values are for the Latin Modern fonts of course.

[45] here we give correctly the bounding box for the math italic f… without its italic correction!

correction automatically, but if we tell TeX to use the text italic in math, and typeset the alphabet, we obtain something exactly identical to typing the letters in text, hence without any italic correction:

abcdefghijklmnopqrstuvwxyz	`text italic in text`
abcdefghijklmnopqrstuvwxyz	`text italic in math`
$abcdefghijklmnopqrstuvwxyz$	`math italic in math`
abcdefghijklmnopqrstuvwxyz	`math italic in text`

Where are our italic corrections gone? the last line was done with `\usefont{OML}{mlmm}{m}{it}` and the line before that using math mode is longer and confirms that italic corrections have been used for the math italic in math mode.

Turning to the TeXbook (and its Appendix G) we learn that in such circumstances, for the italic corrections to be put in from the font, one of its parameters, the interword space (aka `\fontdimen2`), should be zero. It is indeed zero for the math italic font, not for the text italic.

It is possible to make TeX believe it is. Doing so, we obtain in math mode with the text italic:

$abcdefghijklmnopqrstuvwxyz$	`text italic in math`
$abcdefghijklmnopqrstuvwxyz$	`math italic in math`

We saw that the italic correction was taken into acount automatically (independently of the value of the interword space font parameter) in expressions such as `$f(x)$`. Another clever thing done by TeX is to use it for the placement of superscripts; the next examples systematically use the text italic in math. We see that f^j is very different from f^j... where the latter was coded with `$\hbox{\itshape f}^j$`. The inputs `$\mathit{\hbox{\itshape f\/}^j}$` and `$\mathit{f^j}$` give almost identical results: $\boxed{f^j}$ vs. $\boxed{f^j}$. Close examination reveals that the horizontal spacing is exactly identical, however the exponent in the second case is a bit lower. Anyway, the point is that in the second case the italic correction for *f* was indeed used.

Subscripts are another matter: they do *not* take into account the italic correction. For example `$\mathit{f_i}$` gives the same horizontal positions as `$\mathit{\hbox{\itshape f}_i}$`: f_i vs. f_i. Printing them one on another gives f_i and reveals (use the zoom of your viewer!) that only the vertical placement was affected, not the horizontal placement.

We learn in Appendix G of the TeXbook that the italic correction is used for the horizontal shift of the superscript with respect to the position of the subscript: f_i^j, or, going back now to the standard math italics f_i^j. In the next paragraphs we use f_i^i for more accurate comparison of the positioning of the sub- and superscript.

If we try something like this: `${f\/}_i^i$` we obtain f_i^i. Our overlapping game with `\rlap{$f_i^i$}${f\/}_i^i$` gives f_i^i. We discover that the effect of the explicit italic correction has mainly been to translate the subscript horizontally to be positioned exactly below the superscript![46] We most probably do *not* want this to happen for our indices and exponents in math mode. So perhaps we can rejoice

---

[46] there are also some tiny vertical displacements of the sub- and superscripts.

in how astute TEX has been in judiciously using the italic correction data, and there seems to be no need into fiddling with this algorithm which seems to work well even when applied to a text italic font. Actually we may even be of the opinion that the text italic version f_i^i is a bit better-looking than the true math italic f_i^i . . .

But wait... **mathastext** was initially developed to easily use in math mode the document text font not in its italic variant, but as is, so, usually, upright. And upright TEX fonts may also have italic correction data! And what I just said about the shift of the superscript with respect to the subscript apply equally well to such a font, if TEX has been told to use it. Let's try Latin Modern Upright for letters in math: `$f_i^i$` now gives[47] $\mathrm{f}_\mathrm{i}^\mathrm{i}$. We see the italic correction in action for the positioning of the superscript! Compare with `$\mathrm{\hbox{f}_i^i}$`: $\mathrm{f}_\mathrm{i}^\mathrm{i}$. Overlapping with `\rlap{$\mathrm{f_i^i}$}$\mathrm{\hbox{f}_i^i}$` gives $\mathrm{f}_\mathrm{i}^\mathrm{i}$ and shows that the upright f has an italic correction which was used to shift the superscript to the right (and it is now in a slightly lower position). Let's now do `$\mathrm{{f\/}_i^i}$`: this gives $\mathrm{f}_\mathrm{i}^\mathrm{i}$ and the subscript is shifted to the right, and is now on the same vertical axis as the superscript. There are also some slight vertical displacements, `\rlap{$\mathrm{f_i^i}$}$\mathrm{{f\/}_i^i}$` gives $\mathrm{f}_\mathrm{i}^\mathrm{i}$.

People will tell me crazy, but if we decide for using upright fonts in math, wouldn't it be satisfying to have the subscript and superscript positioned on the same vertical axis? the letter has no slant, why should the indices display one?

We end up in this strange situation that it is attractive to systematically incorporate the italic corrections after the upright Latin letters in math! But we don't want to do this inside the arguments to math alphabets as this would make impossible the formation of ligatures (the standard `$\mathrm{ff}$`, `$\mathit{ff}$`, `$\mathbf{ff}$`, `$\mathsf{ff}$` all give ligatures ﬀ, *ﬀ*, **ﬀ**, and ﬀ and we would like to preserve this behavior).

Starting with version `v1.2b`, **mathastext** adds the italic correction automatically after each letter of the Latin alphabet in math mode, *except* when these letters are italic or slanted.[48]

These italic corrections are canceled inside the arguments to the math alphabet commands, to allow the formation of ligatures as is expected in the standard default TEX font set-up in math.

The feature-implementing commands `\MTicinmath`, `\MTnoicinmath`, `\MTicalsoinmathxx` are described in subsubsection 2.2.2.

---

[47] we just use `$\mathrm{f_i^i}$`.

[48] the situation is rather ironical! by the way, the warnings in subsubsection 1.8.4 with `$x^?$` or similar are less of an issue here, because the letter is only *followed* by `\/` and anyhow the whole is put by **mathastext** within group braces, so no surprises with `$x^y$` or `$\mathbin x$`. Nevertheless it is still true that (in math mode only) the letters `a-z`, `A-Z`, expand to composite objects, something which could surprise other packages. The command `\MTmathstandardletters` cancels this mechanism.

Note: *from brief testing on 2012/12/28, XeTeX seems not to obey in math mode italic corrections for OpenType fonts. Hence the TeX placement algorithms for math mode described in this section do not work well when an OpenType (text) font is used for the letters in math mode, and the document is compiled with the XeTeX engine. On the other hand LuaLaTeX seems to implement the italic corrections when using OpenType fonts, but only with italic fonts (as far as I could tell). Try the following (which will use the OpenType Latin Modern font) on a recent TeX installation and compare the output of both engines:*

```
\documentclass{article}
\usepackage{fontspec}
\begin{document}
\Huge
$\mathit{f_i^i}$\par $\mathrm{f_i^i}$
\end{document}
```

Comment out the `fontspec` *line and use pdfLaTeX. All three outputs are different on my TeX installation. XeTeX does not have the italic corrections. LuaLaTeX does, but only for the italic font. pdfLaTeX has them for both the italic and the upright font.*[49]

1.8.3 Extra glue after `\exists`, `\forall`, and before the prime glyph

`\MTforallskip`, `\MTexistsskip`, and `\MTprimeskip` are three commands with each a mandatory argument like for example `3mu plus 1mu minus 1mu` or just `2.5mu`. They are especially useful when using an upright font in math mode. The `mu` is a unit length used in math mode ('math unit', 1/18th of the 'quad' value of the symbol font in the current style). Its value is relative to the current math style. Its use is mandatory in the commands described here.

- compare $\forall\mathrm{B}$ with $\forall\mkern2mu\mathrm{B}$, typeset after `\MTforallskip{2mu}`,
- compare $\exists\mathrm{N}$ with $\exists\mkern2mu\mathrm{N}$, typeset after `\MTexistsskip{2mu}`,
- and finally compare f' with $\mathrm{f}\mkern2mu'$, typeset after `\MTprimeskip{2mu}`.

These three commands may be used throughout the document, or also in the preamble, in which case the declared math versions will record the then current values of the skips. **`mathastext`** applies the following (small) default skips: `0.6667mu` for the skip after $\forall$, `1mu` for the skip after $\exists$, and `0.5mu` for the skip before the prime. The examples above become $\forall\mathrm{B}$, $\exists\mathrm{N}$ and f'.[50]

With the `italic` option the defaults are set to zero. Indeed $\forall B$, $\exists N$ and f' look fine without additional skips. If the document decides then to declare in the preamble a math version with an upright font it is thus recommended to use the

---

[49] 2016/11/04: the situation hasn't changed, at least on current TL2016.
2022/10/29: no change with current TL2022.

[50] the derivative glyph from the txfonts math symbols adapts itself better to an upright letter, no skip seems to be needed then.

commands in the preamble before the `\Mathastext[`⟨*version_name*⟩`]` (or `\MTDeclareVersion`) command defining the version. They will be remembered when this math version is entered in the document. The commands may also be used directly in the document body.

Under the `subdued` option, the *normal* math version (at the start of the document body, or after `\MTversion{normal}`) and the *bold* math version (either at the start of the document body after `\boldmath`, or after `\MTversion{bold}`) do not have any extra skip inserted (even one of zero width) after ∀, ∃, or before the ′.

1.8.4 Extended scope of the math alphabets commands

Ever since the initial version of the package, some characters usually unaffected by the math alphabet commands `\mathbf`, `\mathtt`, `\mathsf`... are declared to be of 'variable family type', in order for them to obey these commands: for example the hash sign # gives **#** if input as `$\mathbf{\#}$` (**`mathastext`**, especially in its beginnings, wanted as many characters as possible to be picked up from the text font and to behave similarly to letters and digits).

So it was especially frustrating that mathematical characters such as +, or <, or] could not be declared of 'variable family' (in addition to being picked up in the text font) as this would, for reasons of the inner workings of TeX, not be compatible with the automatically inserted spaces around them.

A revolutionary ;-) novelty is introduced with version `1.2` of the package: (1.2)

1. the pre-declared or user-declared (using the `amsmath` `\DeclareMathOperator` or equivalent) operator names obey the math alphabet commands,[51]

2. as well as all non alphabetical characters treated by **`mathastext`**.

The non-letters handled by **`mathastext`** (if not disabled by options) fall into two groups:

- Those to which TeX associates some specific spacings:

  ! ? , : ; + − = ( ) [ ] < > { } ∗

  We call them the "hard" ones.
- Those for which TeX uses so-called ordinary spacings:

  . / | \ # $ % &

  We call them the "easy" ones.

The "easy" non-letters are handled *easily* by **`mathastext`**, simply by declaring them to be of "variable family type". This will be done automatically.[52]

---

[51] contrarily to the next feature, this one is not likely to create incompatibilities with other packages, so it is activated by default.

[52] # $ % & obey the math alphabets since the initial version of **`mathastext`**; the dot ., the slash /, the vertical bar | and the backslash \ do not have specific spacings inserted by TeX around them, and the procedure is then activated by default since `1.2` for these characters as they are 'easy non-letters'. But for `\mid` and `\setminus` which are | and \ with special spacing (of type `\mathrel` and `\mathbin` resp.) the activation requires `\MTnonlettersobeymathxx`.

The “hard” non-letters require a more complex approach using a concept called “mathematical activation”. For reasons explained next, this is not done automatically and requires the user to employ the command `\MTnonlettersobeymathxx` (or, for the specific case of the asterisk the option `asterisk`).

> It is a fundamental feature that the spacing added by TEX before and after each such `mathastext`-ified non-letter is in no way modified.

Let us compare, for example, the new behavior of `\mathtt` and `\mathbf` when `\MTnonlettersobeymathxx` has been used

$$\textit{new}:\quad \mathtt{(sin(n!) < cos(m - p)?)} \qquad \mathbf{[sin(x+y)=cos(z-t)]}$$

with the traditional default behavior one observes without `mathastext` or in `subdued` normal or bold math version:

$$\textit{standard}:\quad (\sin(\mathtt{n}!) < \cos(\mathtt{m}-\mathtt{p})?) \qquad [\sin(\mathbf{x}+\mathbf{y}) = \cos(\mathbf{z}-\mathbf{t})]$$

The commands for *deactivation* are:
`\MTmathoperatorsdonotobeymathxx`,
`\MTeasynonlettersdonotobeymathxx`,
`\MTnonlettersdonotobeymathxx`,
and those for *activation*:
`\MTmathoperatorsobeymathxx` (done by default),
`\MTeasynonlettersobeymathxx` (done by default),
`\MTnonlettersobeymathxx` is *not* done by default (see explanations why in the shaded box next) and applies to the “hard” non-letters mentioned above and also to `\mid` and `\setminus`. Furthermore, it applies to the braces { } only if `\MTexplicitbracesobeymathxx` is also used.

> **Important:** the package does not execute on its own `\MTnonlettersobeymathxx`. The reason is that the mechanism in effet replaces the original “hard” characters such as `?`, `[`, `<` by (in math mode only) a more complex structure, which ceases looking to TEX as only one token, and as a consequence `$x^?$`, `$R^+$` or `$\mathopen<A\mathclose>$` raise an error, the workaround being to employ additional braces: `$x^{?}$`, `$R^{+}$` and `$\mathopen{<}A\mathclose{>}$`.
>
> Similarly `$R^*$` does not work anymore under option `asterisk`, the user is supposed to know that with this option `$R^{*}$` is the mark-up to use.
>
> Thus, if one adds
>
> `\usepackage{mathastext}\MTnonlettersobeymathxx`
>
> to a pre-existing document, it is needed to check if the mark-up satisfies to the

above guidelines. For this reason the mechanism is by default not activated and the user has to execute:

`\MTnonlettersobeymathxx`

This can be done in the preamble and will trigger actual modifications only at time of `\begin{document}`.

Some TEXnical notes:[53]

- The asterisk $*$ is associated with its own option `asterisk`, because independently of the matter of its behavior in the scope of math alphabets commands, its handling by **`mathastext`** (via the `\MTlowerast` associated configuration) always requires it to be made mathematically active and to expand to a more complex structure. So once this option is received by the package, it goes full steam and also adds by default the responsiveness to math alphabet commands (in non-`subdued` math versions).
- The legacy situation which is a bit late to change is that the responsiveness of the asterisk to math alphabets being on by default under option `asterisk`, it was decided that this responsiveness would be turned off by `\MTeasynonlettersdonotobeymathxx` and back on by `\MTeasynonlettersobeymathxx`, although the $*$ is part of the "hard" non-letters. One can use `\MTnormalasterisk` to stop the $*$ to obey math alphabet commands, but it also cancels the `\MTlowerast` mechanism.
- changed: At `1.4` "easy" non-letters found out to be (at `\begin{document}` or with `\MTversion` except for `subdued` *normal* or *bold*) mathematically active characters will not get overridden by **`mathastext`**. This applies for example with the dot "." under babel with Spanish language. Its special behavior to transform into a comma "," if before a digit is now kept. As an amusing side note, if `\MTnonlettersobeymathxx` is used, this comma does obey math alphabet commands. (1.4)
- An "easy" non-letter which (at `\begin{document}` or with `\MTversion` except for `subdued` *normal* or *bold*) is catcode active will still be set by **`mathastext`** to be of "variable family type", if it is not mathematically active. The catcode status is not checked nor modified, only the mathcode as per previous item. Now whether the **`mathastext`**-ification has any effect depends on how the user has configured the catcode active character to behave if in math mode.
- When **`mathastext`** wants to employ mathematical activation for a non-letter character among the "hard" cases (inclusive of the asterisk), which is done at `\begin{document}` if not `subdued` or with every `\MTversion` except for `subdued` *normal* or *bold*, it first checks if this character is currently catcode active. If this is the case, it then checks if the character is a babel shorthand. If yes, it then hooks into babel internals to modify the way this shorthand acts in math mode, so that now the character will respond to math alphabet commands. If however the catcode active character does not appear to be a babel shorthand, then **`mathastext`** does not do anything at all (beyond its general business at package loading time to set-up the font used for the non-active token). In relation to this context **`mathastext`** should always be loaded *after* babel. And also *after* the amsmath package.
- The braces `\{` and `\}` remain unresponsive to the alphabet changing commands even after `\MTnonlettersobeymathxx`. The user must employ for this

  `\MTexplicitbracesobeymathxx`

  This has the disadvantage that `\{` and `\}` become then unusable as variable-size delimiters: `\big\{` or `\big\}` create errors and one must make use of `\big\lbrace` and `\big\rbrace`.

---

[53] The complete truth is only to be found in source code. Here are some extra details:
The dollar sign and the curly braces are not tested for mathematical or catcode activation.
changed: The $*$ is tested as are the "hard" non-letters, except under option `everymath`. On the other hand `\ast` will be modified independently of what is decided for $*$. Comprenne qui pourra. (1.4)

But one can now enjoy $\mathtt{\{a, a > b\}}$, $\mathbf{\{a, a > b\}}$, $\mathsf{\{a, a > b\}}$, or even $\boldsymbol{\{a, a > b\}}$.[54][55]

- Even with `\MTnonlettersobeymathxx`, the parentheses-like symbols (,), [,], $<$ and $>$ and the slashes /, \, *if used as left/right delimiters* (i.e. with `\left/\right`) will not respond to math alphabet commands. This is mainly explained by the fact that the text font will not contain suitable glyphs, hence no attempt was made by `mathastext` to make the delimiters pick up their glyphs there.
  But `mathastext` does try to pick up most of the 'small variants' of the delimiters from the text font: `$\left<x\right>$` gives $<x>$ but `$\left<b\right>$` gives $\langle b\rangle$. Notice that this differs from standard LaTeX for which `$\left< x\right>$` gives $\langle x\rangle$. As it is perhaps a bit strange to have $<x>$ and $\langle y\rangle$ not use the same bracketing glyphs, there is an option `nosmalldelims`: with this option the small-sized variants of the delimiters are not modified by `mathastext` (option `nosmalldelims` has the side effect that, for the non-delimiter uses of `\{, \}` to be `mathastext`-ified it is necessary to issue `\MTnonlettersobeymathxx` and `\MTexplicitbracesobeymathxx`.)
- At any rate, as said above, whether 'small' or not, delimiters will remain unresponsive to math alphabet commands, due to technical aspects of TeX, and the way `mathastext` handles these things. Examples:
  - `\mathbf{<a,b>}` gives $\mathbf{< a, b >}$: no use of `\left/\right`, hence brackets do obey the math alphabets — as we issued `\MTnonlettersobeymathxx` a bit earlier,
  - `\mathbf{\left<a,b\right>}` gives $\langle \mathbf{a, b}\rangle$: delimiters used with `\left/\right` will not obey the math alphabets,
  - `\mathbf{\mathopen{<}a,b \mathclose{>}}` gives $\mathbf{<a, b>}$: no `\left/\right`, so the brackets do obey the math alphabets due to `\MTnonlettersobeymathxx`.
  - to compare, the LaTeX standard for `\mathbf{\mathopen{<}a,b\mathclose{>}}` is to produce $<\mathbf{a}, \mathbf{b}>$: neither brackets nor the comma are responding to `\mathbf`.

1.8.5 Hacking letters (and even digits) for special tasks

For some ascii characters, i.e. the "hard" non-letters considered in the previous section, and even more spectacularly for all ascii letters, `mathastext` achieves the capabilities describes in earlier sections via a TeXnique known as "mathematically active characters". For letters this is turned on by default, but for the non-letter characters it is the command `\MTnonlettersobeymathxx` which triggers this mathematical activation.

It is possible for daring LaTeX users to hook into this architecture. Release `1.4` has made this especially easy regarding ascii letters (and even for digits if using the package option `activedigits`).

Except if told otherwise, `mathastext` will make all Latin ascii letters mathematically active, and when TeX encounters e.g. 'e' in math mode it replaces it with the macro having name `\MTcommandlettere` (mind the ending `e`). Its default definition inserts optional extra skips and/or an italic correction.

You can redefine this `\MTcommandlettere`, or any `\MTcommandletter`⟨*ascii-letter*⟩, to achieve (globally or locally) custom goals.[56] Attention, this redefinition should

---

[54] This last example uses the `\mathnormalbold` additional alphabet defined by `mathastext`.

[55] Let me recall that braces will anyhow not be handled at all by `mathastext` if the document font encoding is `OT1`, except under option `alldelims`.

changed: [56] Prior to `1.4`, there was no equivalent to `\MTcommandletter`⟨*letter*⟩. An internal macro `\mst@`⟨*letter*⟩ (1.4) stood for what is denoted now `\MTmathcharletter`⟨*letter*⟩, but required a `\protected` if redefined as it was submitted to an `\edef` during processing.

not use ‘e’ itself as a “naked” character to typeset in math mode, else an infinite loop will arise at time of use. To access a symbol equivalent to the “naked” ‘e’, use `\MTmathcharlettere` (again mind the ending `e`). For example:

```
\renewcommand\MTcommandlettere{\mathsf{\MTmathcharlettere}}% not \mathsf{e}!
```

In this case we could do the simpler approach with `\textsf`, which works as it will then typeset the `e` in text mode, not math mode.

```
\renewcommand\MTcommandlettere{\textsf{e}}%                 not \mathsf{e}!
```

This gives a priori the same result in our **mathastext** context, which picks the font for `\mathsf` from the one used by `\textsf`, assuming here default configuration in the preamble following the loading of **mathastext**.[57]

It is recommended in general to add an extra pair of braces to avoid problems when used with `_` and `^`. Let’s give an other example and demonstrate its output:

```
\[
\renewcommand\MTcommandlettere{{\mathcolor{blue}{\mathsf{\MTmathcharlettere}}}}%
abcde^efgh
\]
```

$$abcd\mathsf{e}^{\mathsf{e}}fgh$$

As this document is in **subdued** mode, we had, to show the effect, to switch temporarily to some math version activating **mathastext**, and we used here a `times` math version with Times and Helvetica clones.

If you redefine `\MTcommandlettere` as described in this section the optional extra math skips added before or after via `\MTsetmathskips` will be lost, as well as the automatic italic correction `\/` added by **mathastext** for an upright font, and it is up to the redefinition of `\MTcommandlettere` to do the job.

The explanations above apply to any (ascii) Latin letters ⟨*ascii-letter*⟩ with associed macros `\MTcommandletter`⟨*ascii-letter*⟩ and “naked” symbol command `\MTmathcharletter`⟨*ascii-letter*⟩.

With option **activedigits** you can even extend the game to digits. The “raw” digit symbol is kept as a math symbol control sequence `\MTmathchardigit`⟨*y*⟩ where ⟨*y*⟩ stands for the Roman version of the digit: empty for 0, then `i`, `ii`, etc... until `ix`. The macro to redefine for special effect is `\MTcommanddigit`⟨*y*⟩. Example:

```
\[
\renewcommand\MTcommanddigit{{\MTmathchardigit^{\mathcolor{blue}{1}}}
```

---

[57] This example simply reassigns `e` to another font, and one could use also the LaTeX command `\DeclareMathSymbol` for this. But few LaTeX users are familiar with its interface, and such an approach could cost adding an extra math symbol font, depending on what one wants to do. Besides `\DeclareMathSymbol` is a preamble-only command, which limits considerably its usability, forcing basically the change to apply to the whole document. Using the **mathastext** interface via mathematically active characters opens up the possibility of arbitrary replacements, of local scope in the document body.

```
                                             _{\mathcolor{red}{2}}}}%
\renewcommand\MTcommanddigiti{{\boxed{\mathbf{\MTmathchardigiti}}}}
0^0_0
\]
```

$$0_{2}^{\boxed{\mathbf{1}}}{}^{0_{2}^{\boxed{\mathbf{1}}}}_{0_{2}^{\boxed{\mathbf{1}}}}$$

(`\boxed` is from amsmath) Let's hope you find better usage... recall that you can't use digit 0 in its redefinition but must use `\MTcommanddigit`. But you can of course use other digits... except if their definitions use the digit 0 rather than the non-active symbol control sequence `\MTmathchardigit`.

See also `\MTmathactiveletters` and `\MTmathactivedigits`.

2 Package commands

2.1 Commands for regular usage

2.1.1 Preamble-only commands

These commands mainly facilitate the definition of math versions, in a **mathastext** extended sense. It is not necessary to use them to activate the package basic functionalities, as loading **mathastext** is enough (except with the `subdued` option).

- `\Mathastext` (or `\mathastext`) reinitializes **mathastext**: it sets the fonts used in math mode (in versions `normal` and `bold`) for letters, digits and a few ascii symbols to the *current* defaults of encoding, family, series and shape.[58] Both the normal and bold math version are modified by this action of `\Mathastext`.

- `\Mathastext`[⟨*version_name*⟩] rather than redefining the fonts for math mode, `\Mathastext` declares a new *math version*, and it is this math version which will use the then current text font in math mode.[59]

- `\Mathastext`[⟨*version_name*⟩][⟨*parent_name*⟩] declares ⟨*version_name*⟩ and configures it to inherit from ⟨*parent_name*⟩ all which is not under the scope of **mathastext**, such as large symbols. The main use will be with `[bold]` in order for the symbols and large symbols to be typeset as in the `bold` math version. For example, this document has in its preamble:
  ```
  \usepackage{newcent}% this package makes New Century the roman font
  \Mathastext[newcent]% this math version will use New Century
  ```

---

[58] `\Mathastext` updates also the font and shapes for the Greek letters (`LGRgreek` option), and the skips to be inserted after the symbols ∀ and ∃, see *infra.*

[59] The allowed version names are as for the LaTeX `\DeclareMathVersion` macro. *Do not use* `\Mathastext[foo]` *with* `foo` *equal to "normal" or "bold";* this is already taken care of by the initial loading of the package or a later command `\Mathastext` without any optional argument. And it will be rejected.

```
\MTseries{b}        % next \Mathastext will use a bold font
\Mathastext[boldnewcent][bold]% large symbols, etc, will be bold too
```

We can check that it does work:

`\MTversion{newcent}`: $abcde \oint \bigvee \biguplus \bigotimes \bigoplus$

`\MTversion{boldnewcent}`: $\boldsymbol{abcde} \oint \bigvee \biguplus \bigotimes \bigoplus$

Naturally, for this one needs an initial math font setup with some nice bold fonts also for large symbols. This is the case with the excellent txfonts package of Young RYU. As the present document must use many fonts and declares many math alphabets, we did not load the full package and fonts but only the **`largesymbols`**:

```
\DeclareSymbolFont{largesymbols}{OMX}{txex}{m}{n}
\SetSymbolFont{largesymbols}{bold}{OMX}{txex}{bx}{n}
\DeclareFontSubstitution{OMX}{txex}{m}{n}
```

- `\MTencoding`{⟨*enc*⟩}, `\MTfamily`{⟨*fam*⟩}, `\MTseries`{⟨*ser*⟩}, `\MTshape`{⟨*sh*⟩}, and `\MTlettershape`{⟨*sh*⟩}.[60] For example valid respective arguments are, respectively, ⟨*T1*⟩, ⟨*phv*⟩, ⟨*m*⟩, ⟨*n*⟩, and ⟨*it*⟩: this is the Helvetica font in T1-encoding, regular (medium) series, upright shape, and the letters will be in italic shape. Once used their effect applies to all succeeding calls to `\Mathastext`, and can only be undone by using them again with other settings, again followed by a call to `\Mathastext`.

  NOTE: *only* if `\Mathastext` is used next (possibly with a version name as optional argument) will these commands have any real effect.

- `\MTWillUse`[⟨*ltsh*⟩]{⟨*enc*⟩}{⟨*fam*⟩}{⟨*ser*⟩}{⟨*sh*⟩} tells **`mathastext`** to use the font with the specified encoding, family, series, and shape for the letters and digits (and all other afflicted characters) in math mode. The optional argument ⟨*ltsh*⟩ specifies a shape for the letters, for example `\itdefault`, or directly ⟨*it*⟩ or ⟨*sc*⟩.

- `\MTDeclareVersion`[⟨*ltsh*⟩]{⟨*name*⟩}{⟨*enc*⟩}{⟨*fam*⟩}{⟨*ser*⟩}{⟨*sh*⟩}[⟨*other_version*⟩]: declares that the document will have access to the font with the specified characteristics, under the math version name ⟨*name*⟩. For example:

  `\MTDeclareVersion[sc]{palatino}{T1}{ppl}{b}{sl}`

  declares under the name **`palatino`** a version where mathematics will be typeset using the Palatino font in T1-encoding, bold, slanted, and the letters will in fact be in caps and small caps (and bold).[61] When the initial optional argument is absent, and **`mathastext`** was loaded with the **`italic`** option, then the default letter shape will be **`it`**,[62] else letters will have the same shape as used for digits

---

[60] These commands exist also with long names: `\Mathastextencoding`, etc... The same applies to the other commands mentioned in this section.

[61] I do not especially recommend to use this in real life!

[62] more precisely, the shape is the latest value passed in one of the previously used package commands to specify the shape of letters, or the `\itdefault` of the time of loading the package.

and operator-names.

Another optional argument may be used as last argument. Similarly as its use with `\Mathastext` this makes the declared math version inherit, for things not modified by **mathastext** like large symbols, the font set up of the math version whose name was passed as optional argument (typical use will be with `[bold]`). (1.3c)

- `\MTboldvariant{⟨var⟩}`: when used before `\Mathastext`, specifies which bold (`b`, `sb`, `bx`, ...) to be used by `\mathbf` (and `\boldmath`). Default is the `\bfdefault` at the time of loading **mathastext**. When used before the declaration of a version, decides the way `\mathbf` will act in this version.
- `\MTEulerScale{⟨factor⟩}`: scales the Euler font by ⟨*factor*⟩.
- `\MTSymbolScale{⟨factor⟩}`: scales the Symbol font by ⟨*factor*⟩.
- `\MTitgreek`, `\MTupgreek`, `\MTitGreek`, `\MTupGreek`: these commands are active in case the `LGRgreek` option was used; they act as the options of the similar names `itgreek`, `upgreek`, `itGreek`, `upGreek`, but only for the Greek letters in the versions yet to be defined. Their effect become recorded only when the version is declared via `\Mathastext` or `\MTDeclareVersion`.
- `\MTgreekfont{⟨fontfamily⟩}`: a command with a mandatory argument which specifies the font family for Greek letters in all **mathastext** math versions declared afterwards via `\Mathastext` or `\MTDeclareVersion`. Only effective if `LGRgreek` (or `LGRgreek+`) or `selfGreek` option was passed to the package.

  Check the `LGRgreek` documentation for some relevant information.

2.1.2 Commands for body or math

- `\MTversion[⟨nametext⟩]{⟨namemath⟩}`, `\MTversion*{⟨namemath⟩}`, also known as `\Mathastextversion` (and as `\MTVersion`, and `\mathastextversion`):
  - the non-starred version changes *both* the document text fonts and the math fonts (for those characters treated by **mathastext**): the mandatory argument is the math version to be used for math; the optional argument is the name of (another) **mathastext**-declared math version, the font which was chosen during its declaration will be set as document text font (and `\familydefault` etc...also are redefined). In the absence of the optional argument, the mandatory one is used. The versions *must* be either `normal`, or `bold`, or previously declared ones via `\Mathastext` or `\MTDeclareVersion`.
  - the starred variant does the math set-up, but changes *nothing* to the text fonts (see subsection 1.6 for a description of the math set-up, which summarizes what is done additionally to only using LaTeX's `\mathversion`). (1.3c)

`\MTversion[⟨nametext⟩]{⟨namemath⟩}` does `\MTeverymathdefault` (except for `\MTversion{normal}` and `\MTversion{bold}` under package option `subdued`), which in particular activates the insertion of skips around letters specified by `\MTsetmathskips` and also, if the font used is not oblique the insertion of italic corrections (for better positioning of subscripts; see the discussion in subsubsection 1.8.2). Under the `frenchmath` option the package checks separately the letter shape for lowercase and uppercase.

`\MTversion` also does `\MTexistsdoesskip`, `\MTforalldoesskip`, and also `\MTprimedoesskip`, `\MTmathoperatorsobeymathxx`, except under the `subdued` option for *normal* and *bold*, in which case it does the opposite actions. (1.3j)

- `\hbar`: this macro is by default redefined (in a way compatible with the `italic` option) combining the `h` letter and the ¯ accent from the `mathastext` font. Note that `\mathrm{\hbar}` and `\mathbf{\hbar}` will work and that `\hbar` does scale in subscripts and exponents. Since `1.3u`, this is a priori compatible with all 8bits text font encodings supporting the `\=` text accent in the LaTeX way.[63] [64] (1.3u)
- `\fouriervec`: this is a `\vec` accent taken from the Fourier font; the fourier package need not be loaded. Active only if option `fouriervec`.
- `\pmvec`: this provides a poor man `\vec` accent command, for upright letters. It uses the right arrow. Does not change size in subscripts and exponents.
- new description: `\Mathnormal`, `\Mathrm`, `\Mathbf`, `\Mathit`, `\Mathsf`, `\Mathtt`: they use the `mathastext`-ified fonts. By default, `\mathnormal`, `\mathrm`, `\mathbf`, `\mathit`, `\mathsf`, `\mathtt` are redefined to map to these new commands using the `mathastext` fonts. The option `defaultalphabets` tells to keep them with their original meanings. Alternatively the original commands can be saved under other names before loading `mathastext`: the underlying architecture is not deleted by the package, and aliases defined before loading `mathastext` will work as expected. (1.3za)
- `\mathnormalbold`: a bold version of `\mathnormal`, i.e. picks up the math alphabet used for ascii letters as mathematical variables, but in a bold weight. When the package typesets such letters in the same shape as for operator names (i.e. neither `italic` option nor the `\MTlettershape` command have been used) the output is as the one of `\mathbf`.

  This command is also made available under `subdued` option in the “normal” and “bold” math versions, as LaTeX does not define it a priori, contrarily to `\mathbf` and other math alphabet commands.
- `\mathgreekup`: math alphabet, only available under `LGRgreek` (or `LGRgreeks`) option, which gives access to ‘upright’ Greek letters (picked up from a font available in `LGR`-encoding). Note that the package also defines `\alphaup`, . . . , `\piup`, (1.3y)

---

[63] The horizontal skips for letter `h` from `\MTsetmathskips` are ignored for `\hbar`. (1.3u)

[64] The `\hbar` redefinition is canceled in normal and bold math versions under the `subdued` option. (1.3u)

… mathematical character tokens, see subsubsection 1.7.3. What “`up`” shape really means may be math version dependent. It is configurable in the preamble via re-defining `\MTgreekupdefault` and then declaring the math version via `\Mathastext` (with optional argument if for a math version other than the “normal” one), or `\MTDeclareVersion`. The font used is also math version dependent: it is the one which was similarly configured via usage of `\MTgreekfont` prior to the `\Mathastext` or `\MTDeclareVersion` step. In absence of any such configuration in the preamble, it will be (in all math versions) the family default at time of loading the package (which thus has then to be available in `LGR` encoding; it is not a problem if the family default has no `LGR` support as long as suitable usage of `\MTgreekfont` later on configures a suitable font).

Also `\mathgreekupbold`. (1.3za)

These math alphabets are also available under `subdued` option in the “normal” and “bold” math versions, as LaTeX does not (a priori) define analog ones, so `mathastext` has no reason not to leave them live. Note though that `\mathgreekup{\pi}` will work only if the original `\pi` is of “variable family type” which is not the case except if some math package handling Greek was used, but then why load `mathastext` with option `LGRgreek`?

But you can use `\mathgreekup{p}` as the slot number of `p` in the LaTeX font for mathematical letters is the same as the slot number of π in `LGR` encoding.

Or, use rather `\piup` because it is not undefined by `mathastext` in `subdued` normal mode, as LaTeX has no a priori definition for it. Or use (but why?) `\mathgreekup{\piit}`.

The `LGR` font family used will be the latest one configured by `\MTgreekfont` usage followed by `\Mathastext` (*without* optional argument) in the preamble which is what is needed to modify the non-subdued aspects of subdued “normal” math; if no such configuration was done, the font family will be the family default found at time of loading the package.

Worse: Δ is per LaTeX default of variable family type but its slot number in its assigned font is not at all the one of the `LGR` encoding, so `\mathgreekupbold{\Delta}` will give some unrelated glyph. This is because `mathastext` restores the pristine `\Delta` in `subdued` normal mode to its original meaning. But it keeps its own defined `\Deltaup` and `\Deltait`, so you can use `\mathgreekupbold{\Deltaup}` for example. Or `\mathgreekupbold{D}` as the mathematical letter `D` slot number in LaTeX is also the one of Δ in `LGR` encoding.

I am sorry for such lengthy explanations, but this is to comment on why `mathastext` keeps also in `subdued` normal math some of its Greek related functionality, if option `LGRgreek` was used. Most `mathastext` users will not use the `subdued` option anyhow.

- `\mathgreekit`: math alphabet, only available under `LGRgreek` (or `LGRgreeks`) option, which gives access to ‘italic’ Greek letters (picked up from a font available in `LGR`-encoding). The actual shape is configurable via re-defining `\MTgreekitdefault` and then redeclaring the math version via `\Mathastext` (with optional argument if for a math version other than the “normal” one), or `\MTDeclareVersion`. (1.3y)

  Also `\mathgreekitbold`. (1.3za)

  See the discussion of `\mathgreekup` for some TeX hacker level information on what happens with `subdued` option in the “normal” (or “bold”) math version.

- `\inodot`, `\jnodot`: the corresponding glyphs in the `mathastext`-ified font for use in math mode. By default, `\imath` and `\jmath` are redefined to use them. The `defaultimath` option prevents that, but `\inodot` and `\jnodot` are always defined (although they may not render correctly if the used text font is missing the glyphs). Since `1.3t`, these macros obey the `subdued` regime.
- `\MathEuler`, `\MathEulerBold`: math alphabets to access all the glyphs of the Euler font, if option `eulergreek` (or `eulerdigits` was passed to the package.
- `\MathPSymbol`: math alphabet to access the Symbol font.
- when one of the options `symbolgreek`, `eulergreek`, or `selfGreek` is passed to the package the capital Greek letters which look like their Latin counterparts acquire names: `\Digamma`, `\Alpha`, `\Beta`, `\Epsilon`, `\Zeta`, `\Eta`, `\Iota`, `\Kappa`, `\Mu`, `\Nu`, `\Omicron`, `\Rho`, `\Tau`, `\Chi` (no `\Digamma` for Symbol). Also an `\omicron` control sequence is provided.
- `LGR` Greek and 'var'-letters: only the `\varsigma` is available in this encoding, so using for example `\varphi` will load the previous default math font. It might thus be suitable when recompiling already written LaTeX sources to add to the preamble `\let\varphi=\phi`, `\let\varepsilon=\epsilon`, etc..., in case only the 'variant' form of the letter was used in the documents.
- Miscellaneous mathematical symbols from the postscript Symbol font are made available (or replaced) by option `symbolmisc`.[65] They are `\prod` $\prod$ `\sum` $\sum$ `\implies` $\Rightarrow$ `\impliedby` $\Leftarrow$ `\iff` $\Longleftrightarrow$ `\shortiff` $\Leftrightarrow$ `\to` $\to$ `\longto` $\longrightarrow$ `\mapsto` $\mapsto$ `\longmapsto` $\longmapsto$ `\aleph` $\aleph$ `\inftypsy` ∞ `\emptyset` $\emptyset$ `\surd` $\surd$ `\nabla` ∇ `\angle` $\angle$ `\forall` $\forall$ `\exists` $\exists$ `\neg` $\neg$ `\clubsuit` $\clubsuit$ `\diamondsuit` $\blacklozenge$ `\heartsuit` $\heartsuit$ `\spadesuit` $\spadesuit$ `\smallint` $\smallint$ `\wedge` $\wedge$ `\vee` $\vee$ `\cap` $\cap$ `\cup` $\cup$ `\bullet` $\bullet$ `\div` $\div$ `\otimes` $\otimes$ `\oplus` $\oplus$ `\pm` $\pm$ `\ast` $\ast$ `\times` $\times$ `\proptopsy` $\propto$ `\mid` $\mid$ `\leq` $\leq$ `\geq` $\geq$ `\approx` $\approx$ `\supset` $\supset$ `\subset` $\subset$ `\supseteq` $\supseteq$ `\subseteq` $\subseteq$ `\in` $\in$ `\sim` $\sim$ `\cong` $\cong$ `\perp` $\perp$ `\equiv` $\equiv$ `\notin` $\notin$ `\langle` $\langle$ `\rangle` $\rangle$. And a `\DotTriangle` $\therefore$ is made available by option `symbolre` (which overwrites `\Re` and `\Im`: $\Re, \Im$). The `\inftypsy` and `\proptopsy` have these names to leave up to the user the choice to replace (or no) the original (larger) `\infty` ∞ and `\propto` $\propto$.

  Regarding the `\prod` and `\sum` commands: they will use the Symbol glyphs $\prod\sum$ in inline math, and in display math the Computer Modern ones (or whatever is set up by other packages; here we have the symbols from `txfonts`):

$$\prod\sum$$

---

[65] option `asterisk` is also required to treat the $*$. Recall from subsubsection 1.8.4 that the asterisk in math mode (also when using the control sequence `\ast`) appears then to TeX to be a composite object.

The package provides `\prodpsy` and `\sumpsy`: if one really wants in all situations the Symbol glyphs, one can do `\let\prod\prodpsy` and `\let\sum\sumpsy`. Also `\MToriginalprod` and `\MToriginalsum` will refer to the `\prod` and `\sum` before redefinition by the package: this is to allow constructs such as `$\displaystyle\MToriginalprod$` or `\[\textstyle\MToriginalprod\]`, because they would not work with the `\prod` and `\sum` as re-defined by the package.

2.2 Commands for expert usage

A few preliminary comments, mainly destined to advanced users aware of some TEX innards (more extensive explanations are to be found in the code comments).

The timing for actions of **`mathastext`** falls into three cases:

1. things done by **`mathastext`** itself during its loading (some are delayed to `\begin{document}`),
2. things done as the result of user commands, either in the preamble or in the body of the document, (but note that some commands if used in preamble have a real effect only at the time of `\begin{document}`),
3. things done everytime math mode is entered.

changed: At `1.4` a very significant change took place: the last category, the one of things done everytime math mode is entered, has become *empty*.[66] (1.4)

The `everymath` option added at `1.4` re-establishes the legacy behavior. Except for ascii letters: they will not even then be made mathematically active at each entrance in to math mode, but only once at time of package loading (if not `subdued`) and also when using `\MTversion` (for non-`subdued` math versions).

Under `everymath` option some commands described here as being usable everywhere have in fact an effect only if used externally of math mode. The documentation is only accurate for the default `1.4` configuration, not for the legacy one as re-enacted by `everymath`.

2.2.1 Expert commands which are preamble-only

- `\MTgreekupdefault`: a command with no argument whose expansion specifies, under `LGRgreek` regime, the shape for the ‘`up`’ Greek control sequences (and for the no-postfix Greek control sequences under `upgreek` option) in all **`mathastext`** math versions declared *afterwards* via `\Mathastext` or `\MTDeclareVersion`. The a priori default for this shape is ‘`n`’ (without the quotes). See subsubsection 1.7.3. (1.3y)

---

[66] There ia always an exception to a good rule, and here it is: only with LuaLATEX engine, a certain command is executed as part of `\everymath`. For details see `\MTfixfonts`.

This command can also be defined *prior* to loading the package, as the package itself only does:

```
\providecommand*\MTgreekupdefault{n}
```

- **\MTgreekitdefault**: a command with no argument whose expansion specifies, under `LGRgreek` regime, the shape for the '`it`' Greek control sequences (and for the no-postfix Greek control sequences under `itgreek` option) in all **mathastext** math versions declared *afterwards* via `\Mathastext` or `\MTDeclareVersion`. The a priori default for this shape is '`it`' (without the quotes). See subsubsection 1.7.3. (1.3y)

  This command can also be defined *prior* to loading the package, as the package itself only does:

  ```
  \providecommand*\MTgreekitdefault{it}
  ```

2.2.2 Expert commands usable everywhere

- **\MTcustomgreek**: in case **mathastext** has been loaded with one of its Greek related options, this activates the corresponding customization of Greek letters in math mode. It is issued automatically by the package in the preamble (except if loaded with `subdued` option) and at each switch of math version via `\MTversion` or `\MTversion*` (except for the normal and bold math versions in subdued mode). Also available as `\Mathastextcustomgreek`. May be used even inside of math mode. (1.3d)

- **\MTstandardgreek**: in case **mathastext** was loaded with one of the Greek related options this command reverts the customization, it resets the Greek letters to their definitions in force at package loading time. Can be used in the preamble, but is mainly for the document body (may even be used inside math mode ...). Done automatically under the `subdued` option when switching to the normal or bold math version. Also available as `\Mathastextstandardgreek`. (1.3d)

- **\MTsetmathskips**`{`⟨*a-z/A-Z*⟩`}{`⟨*muglue_before*⟩`}{`⟨*muglue_after*⟩`}`: is used to specify extra skips (or rather mu glue) to be inserted in math mode, before and after a letter. The rationale is that standard text fonts used in math mode may sometimes cause glyph (near-) collisions with math symbols, as TEX has some implicit expectations on the design of fonts for math letters. (1.3a)

  These extra skips around letters are set at their natural width and do not add any stretchability or shrinkability to the math formula as a whole, nor do they result in extra potential break points.

  Random (silly) examples:

  ```
  \MTsetmathskips{x}{\medmuskip}{\thickmuskip}
  \MTsetmathskips{A}{.5mu}{2.3mu}
  ```

  and the effect: $vw\,x\;yz\,A\,BC^{vw\,x\;yz\,A\,BC}$. The effect obeys the usual LATEX scoping rules.

The first argument of `\MTsetmathskips` may be any expandable code giving a letter; this facilitates use of `\MTsetmathskip` in `\@for` loops such as this one:

```
\makeatletter
\@for\@tempa:=a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z\do{%
              \MTsetmathskips{\@tempa}{2mu}{2mu}}%
\makeatother
```

Starting with v1.3i: the extra skips are *not* applied to the letters within the scope of math alphabet commands, or the letters from operator names (predefined or user declared).

Note that contrarily to the `\MTexistsskip`, `\MTforallskip`, and `\MTprimeskip` commands described next, these extra skips (which may be specified in the preamble) are not recorded in the definition of the math version (as defined via `\Mathastext` with its optional argument or via `\MTDeclareVersion`). The declared skips hold thoughout the document until modified or canceled, independently of math versions (of course, `mathastext` cancels the skips in the normal and bold math versions if package option `subdued` was used).

- `\MTunsetmathskips{`⟨*a-z/A-Z*⟩`}`: cancels the skips for that letter (they are not set to `0mu` but completely removed).

  The argument may be a macro (or any expandable code) expanding to a letter.

- `\MTnormalexists`, `\MTexistsdoesskip`: the latter (done by default if not **`subdued`**, and also on each use of `\MTversion` in the body of the document) makes it so that ∃ takes into account the math glue as specified by `\MTexistsskip`. The former is its opposite. (1.3j)

- `\MTexistsskip{`⟨*math glue*⟩`}`: specifies the amount of skip or more generally glue to put after each ∃ math symbol. Indeed, upright letters (or digits for that matter) often appear to be positioned a bit too close to the quantifier: ∃B. The package default is to add a `1mu` skip (this default is set to zero in the case of `italic`): ∃B. One can change the default with the following syntax: `\MTexistsskip{2mu plus 1mu minus 1mu}`, which if used in the preamble and followed with a `\Mathastext` command (or `\MTDeclareVersion`), will be recorded in the definition of this math version (and subsequent ones). One may also use the command at any time in the document. In the case of the option **`subdued`**, the skip is canceled in the *normal* and *bold* math versions. In the case of the option `italic`, the default skip is set to zero.

- `\MTnormalforall`, `\MTforalldoesskip`: the latter (done by default if not **`subdued`**, and also on each use of `\MTversion` in the body of the document) makes it so that ∀ takes into account the math glue as specified by `\MTforallskip`. The former is its opposite. (1.3j)

- `\MTforallskip{`⟨*math glue*⟩`}`: the default is to add a `.6667mu` math skip after each ∀ (except with the option `italic` for which the default skip is set to zero).

Compare ∀F (has the skip) with ∀F (has no skip). Use this command in the preamble to set up the skip or glue to be used in the *next to be declared* math versions. In the case of the option `subdued`, the skip is canceled in the *normal* and *bold* math versions. In the case of the option `italic`, the default skip is zero for all math versions. One may use the command at any location in the document.

- `\MTnormalprime`, `\MTprimedoesskip`: the latter (done by default if not `subdued`, and also on each use of `\MTversion` in the body of the document except for the `subdued` *normal* and *bold* math version) makes it so that ′ takes into account the math glue as specified by `\MTprimeskip`. The former is its opposite. In all cases the right quote `'` is a mathematically active character throughout the document producing ′ as is the default in TEX, it is only its meaning which changes to include or not an extra skip. (1.3j)

  Even though `\MTprimedoesskip` is not done in the `subdued` case, it is *not* a no-op even then in the preamble or in the *normal* and *bold* math versions.

- `\MTprimeskip{⟨math glue⟩}`: specifies the amount of math skip to add before the derivative glyph. The default iniitial value is `0.5mu`, except with the `italic` option where it is `0mu`. In the case of the option `subdued`, the skip is canceled in the *normal* and *bold* math versions.

- `\MTnormalasterisk`, `\MTactiveasterisk`: the latter will use for `*` and `\ast` the text font asterisk, suitably lowered; the former if used in the preamble tells `mathastext` to not modify the non-`mathastext` situation, or if used in the body to revert to it. Both are no-op's in absence of option `asterisk`.

  A legacy feature is that `\MTactiveasterisk` is *not* a no-op in `subdued` *normal* or *bold* math versions, and does let the asterisk obey the `\MTlowerast` configuration, if used explicitly by user (even in the preamble).

  changed: At 1.4 `mathastext` checks at `\begin{document}` (or each time `\MTactiveasterisk` is made use of) if `*` is a Babel shorthand (which I far as I know is the case of no language) or is already mathematically active. In the latter case the `*` is not modified in math mode; and it is not modified either if found to be catcode active but not a babel shorthand. In contrast, the `\ast` will always be set to obey `\MTlowerast` configuration after `\MTactiveasterisk`. (1.4)

- `\MTlowerast{⟨dimen⟩}`: under option `asterisk` a `\raisebox` command is used to lower the text asterisk to produce a reasonable math asterisk. The package uses this command initially with argument `0.3\height`, this will have to be fine-tuned for each given text font but worked out ok with the fonts we tried. The dimension argument will be used also in sub-scripts and sub-sub-scripts, so it is best not to use an absolute dimension. The dimension specification is for text it can not be with `mu` unit.

- `\MTmathoperatorsobeymathxx`, `\MTmathoperatorsdonotobeymathxx`: the former is done by default, it makes operator names obey math alphabets. See also subsubsection 1.8.4. This functionality *does* **not** *rely* on “math active characters”. Automatically issued by each `\MTversion`, except under option **`subdued`** when switching to *normal* or *bold*.

- `\MTmathactiveletters`: ‘math activation’ of all ascii Latin letters. This is done by the package automatically except under the **`subdued`** option. (1.3)

changed: At `1.4` its behavior was modified significantly: instead of setting some toggle obeyed only at entrance of math mode, it acts on the spot immediately. (1.4)

Formerly, this command was only a configuration toggle with no immediate impact (and was not usable from inside math mode). Indeed, the **`mathastext`** work of mathematical activation was done (or not done) *each time* math mode was entered (if not in **`subdued`**), and the ascii letters kept standard mathcodes outside of math mode.

The command now does the mathematical activation on the spot and the meaning of the active shape of the ascii letters — a priori undefined because typesetting a document with an active ascii letter is near impossible, as no LaTeX command name used in the document can contain it — is modified here and then; but of course the catcodes are not modified only the mathcodes are. Except if **`mathastext`** is loaded with the **`subdued`** option, this mathematical activation is executed already at package loading time.

If with the **`subdued`** option, the command is inactive in the preamble, as well as in the *normal* and *bold* math versions. In all cases it gets (re-)executed automatically when `\MTversion` is used for activating a non-subdued math version.

The new situation may be dangerous if the document author makes a letter, say **`A`**, catcode active at some point, with a definition of the active state using `\string A`. As **`mathastext`** has (earlier, in the preamble) set **`A`** to be mathematically active as well, and as the `\string` changes nothing to that, an infinite loop will be triggered by such an **`A`** in math mode.

But, having an active ascii letter can only be in very localized portions of a document, and only be authored by experts. The experts should carefully make sure the mathcode of the letter is not active if they set the catcode to active and let the active meaning use `\string` on the letter itself, thus we can trust that the mathematical activation done earlier by **`mathastext`** is undone and there will be no problems.

This `1.4` change may thus need to be followed by some adjustments in some quite special circumstances caused by some expert interventions.

See subsubsection 1.8.1 and subsubsection 1.8.2 for the rationale of this mathematical activation, and subsubsection 1.8.5 for ways to use it for other goals.

If you do want mathematically active letters also in subdued *normal* or *bold* math versions (to apply tricks such as those of subsubsection 1.8.5) there is `\mst@mathactivateletters` which does not check the **`subdued`** status.

- `\MTmathstandardletters`: turns off the mathematical active ascii letters, i.e. reverts their mathcodes to package font configuration. Here “standard” only (1.3) means that the letters will not be mathematically active, but they are still under the influence of **mathastext** regarding the typeface they use, so they are only "standard" from **mathastext** point of view.

changed: At `1.4` its behavior was modified: instead of setting some toggle obeyed only at entrance of math mode, it acts on the spot immediately. (1.4)

`\MTmathstandardletters` will not try to restore the meaning associated to the catcode active variant of the letter token which may have been in place at the time of `\MTmathactiveletters`. This meaning is not saved for later reset.

Except if the *letter* was an active Babel-shorthand at the time of `\MTmathactiveletters`, which is in fact almost impossible to achieve via the babel interface.

And if the letter was catcode-active due to user action, `\MTmathactiveletters` will not have made it mathematically active, nor will it have modified its active meaning. So the active meaning may get overwritten only for a letter having a normal catcode at time of `\MTmathactiveletters` but which for some reason has some meaning associated to its catcode active variant.

new: - `\MTmathactivedigits`: is a no-op except under option `activedigits`. It then makes all digits mathematically active and is executed during package loading, except if `subdued`. It is again executed when entering any non-`subdued` math version in the document body. See subsubsection 1.8.5 for an example of use. (1.4)

  If you do want mathematically active digits also in subdued *normal* or *bold* math versions (to apply tricks such as those of subsubsection 1.8.5) there is `\mst@mathactivatedigits` which does not check the `subdued` status.

new: - `\MTmathstandarddigits`: a no-op except under option `activedigits`. Under this option it resets the digits to their normal mathcodes as configured by the package. (1.4)

- `\MTicinmath`: this command is executed by default by **mathastext** except in case of option `subdued` or if the user chosen letter shape is oblique (`it` or `sl`). It tells **mathastext** to add italic corrections after all letters in math mode, except within the scope of math alphabets.

changed: This command and the next ones in this item can be used anywhere in the document and even from inside math mode. In case of `subdued` option, using the command from within the preamble remains without effect, as the document body will start in the subdued normal math version anyhow. Each `\MTversion` in the body reemits `\MTicinmath` (in case of non-oblique letter shape), except if the `subdued` option was used and the chosen math version is *normal* or *bold*. (1.4)

The effect of this and the other commands of this item is local to the group or environment in which it has been issued.

The description above about the command not being executed if the letter shape is italic or slanted is not quite right, as it refers only to the act of inserting or not italic correction. The `\MTmathactiveletters` component is always executed, however it will be a no-op in `subdued` *normal* and *bold*.

`\MTnoicinmath`: this command deactivates the package added italic corrections. It can be used inside as well as outside of math mode (or in the preamble of the document). Note that it does not deactivate the mathematical activation of the ascii letters. Use `\MTmathstandardletters` for that.

`\MTICinmath`, `\MTnoICinmath`: only acts on the uppercase letters. But recall that `\MTicinmath` is done by default, thus using italic corrections only for uppercase needs to go via `\MTnoicinmath` then `\MTICinmath`.

`\MTicalsoinmathxx`: this command de-activates the de-activation of the italic corrections inside the arguments to the math alphabet commands. It can be issued inside as well as outside of math mode. Will be effective only if `\MTicinmath` or `\MTICinmath` is in force. To cancel its effect either enclose it in a group or environment or re-issue `\MTicinmath` after it.

- `\MTeasynonlettersobeymathxx`, `\MTeasynonlettersdonotobeymathxx`: the former is done by default, it makes characters `.`, `/`, `|`, `\`, `#`, `$`, `%`, and `&` (if not excluded by package options) obey math alphabet commands. See also subsubsection 1.8.4. This functionality does *not* make the characters “math active” (but it does modify `\mathcode`’s, naturally).

- `\MTnonlettersobeymathxx`, `\MTnonlettersdonotobeymathxx`: the former will make (except if excluded by relevant package options) `!`, `?`, `,`, `:`, `;`, `+`, `-`, `=`, `(`, `)`, `[`, `]`, `<`, and `>` obey the math alphabet commands (when not used as delimiters). These characters are made “math active”, and each one now expands to two tokens. This makes for example `$a^!$` illegal input and it will have to be coded `$a^{!}$`. Hence, by default, the package does `\MTnonlettersdonotobeymathxx`.

- `\MTexplicitbracesobeymathxx`: does `\MTnonlettersobeymathxx` and extends it to also apply to `\{` and `\}`. Note that then `\left\{`, `\right\}` must be coded `\left\lbrace`, `\right\rbrace` rather. There is also `\MTexplicitbracesdonotobeymathxx`.

  If under the `everymath` option, it is needed to issue explicitly `\MTnonlettersobeymathxx` in addition to `\MTexplicitbracesobeymathxx`.

  Whether or not with `everymath`, if the command is issued while in `subdued` normal or bold, it has no effect immediately, but if switching later to a non-`subdued` math version via `\MTversion`, a `\MTnonlettersobeymathxx` issued then will automatically apply also to `\{` and `\}`.

- `\MTeverymathdefault`: this hook is executed by `\MTversion{`⟨*version_name*⟩`}` (1.3j) (but if under option **subdued** and switching to the *normal* or *bold* math version its default effect is overruled by an `\MTeverymathoff` executed after it). Its default meaning is:

```
\MTactiveasterisk % this has no effect without option asterisk
\MTprimedoesskip  % this makes prime glyph obey extra space
\MTeasynonlettersobeymathxx
\MTicinmath       % this does \MTmathactiveletters, hence the extra
```

```
                        % skips from \MTsetmathskips are obeyed.
    \MTmathactivedigits % this is a no-op in absence of option activedigits
    \MTfixfonts         % only operant under LuaLaTeX.
```

Under `subdued` option, switching to the *normal* or *bold* version does `\MTeverymathoff` which includes `\MTnonlettersdonotobeymathxx`.

The default `\MTeverymathdefault` which is issued when going back to a non-*normal* or *bold* math version doesn't do `\MTnonlettersobeymathxx`: thus it is up to the user to correct this if desired.

Notice also that `\MTversion{`⟨*version_name*⟩`}`, except for *normal* or *bold* if `subdued` does `\MTforalldoesskip` and `\MTexistsdoesskip`, which are not included in `\MTeverymathdefault` actions as they are not related to `\everymath` and `\everydisplay`.

At `1.4`, the name of this command diverges somewhat from its action as mathematical activation of ascii letters (or, optionally, of digits) will happen on the spot and not during `\everymath` or `\everydisplay` execution.

- `\MTeverymathoff`: does `\MTnormalasterisk`, `\MTnormalprime`, `\MTnonlettersdonotobeymathxx`, `\MTeasynonlettersdonotobeymathxx`, `\MTmathstandardletters`, `\MTmathstandarddigits` and `\MTdonotfixfonts`. `(1.3j)` `(1.4)`

  Automatically done by `\MTversion{normal}` (or `{bold}`) under option `subdued` (and also `\MTnormalexists` and `\MTnormalforall` are executed then).

  The commands `\url`/`\nolinkurl` of package `hyperref` and `\url` from `url.sty` (which use math mode under the hood) are patched by **mathastext** to do `\MTeverymathoff` automatically: this is needed because **mathastext** modifies anew some mathcodes *each time math mode is entered,* hence may overwrite to some extent the specific preparations done by `{url,hyperref}.sty`. `(1.3i)`

  However, in some cases it may be interesting to be able to apply hacks as described in subsubsection 1.8.5. You can use now `\MTeverymathoff` as a hook inside `\url` and `\nolinkurl` commands. Perhaps redefine it (locally) to do all of the above except `\MTmathstandardletters` and/or `\MTmathstandarddigits`, and use the subsubsection 1.8.5 instructions to achieve special effects for some letters or digits in the URLs rendered via `\url` and `\nolinkurl`. `(1.4)`

2.2.3 Expert commands usable only outside of math mode

There is only one pair of commands here: `\MTfixfonts` and `\MTdonotfixfonts`. They are operant only under LuaLaTeX. As `\MTeverymathdefault` and `\MTeverymathoff` use them, they arguably could have been listed here, but only for engine LuaLaTeX.

- `\MTfixfonts`: this is operant only under LuaLaTeX. It has the effect that each time math mode is entered macro `\MTfixmathfonts` will be executed. The latter forces so-called `base` mode for the used text font in math mode, in an effort to (only partially, see code comments) fix the fact that OpenType features such as Lining Figures were in some cases not being applied in math mode when one uses text fonts there (text fonts are declared by LuaTeX+`luaotfload` to use `node` or `harf` mode, which are non-functional in math.) It is invoked automatically by the package (except for `normal` and `bold` math versions under `subdued` option), and in normal situations, there is no reason to use it directly. `(1.3o)`

  `changed:` The hack was updated at `1.4` in order to also handle fonts using `Renderer=HarfBuzz`. `(1.4)`

- `\MTdonotfixfonts`: cancels the job of `\MTfixfonts`. Done automatically in `subdued` mode when in the `normal` or `bold` math version; in normal contexts, there is no reason to use this command. Only operant under LuaLaTeX. (1.3o)

2.2.4 Expert commands usable only in math mode

- `\MTfixmathfonts`: this used to be an internal package macro but it is given a public name at 1.3p because I discovered that `$..\hbox{\mathversion{foo}$..$}..$` causes an issue and one needs to invoke again `\MTfixmathfonts` *after* the `\hbox`, for some reason. To be used *only* under LuaLaTeX and only for such rare cases where it may be needed. (1.3p)

3 Package options

3.1 Summary of main options

`italic`: tells **`mathastext`** to typeset the ascii letters in math using italic shape; indeed, its legacy historical default is to typeset them in roman (upright) shape.

`frenchmath`: lowercase ascii letters in italic shape, uppercase in upright shape. Also lets the Greek letters, if the latter are under **`mathastext`** influence, be upright, i.e. also the lowercase ones.

`subdued`: tells **`mathastext`** to not change the default fonts or the math alphabets for the normal and bold math versions. The **`mathastext`**-ification activates only after `\MTversion`{⟨*version_name*⟩} usage in the document body, where the ⟨*version_name*⟩ was declared as an **`mathastext`** enriched math version in the preamble via `\Mathastext`[⟨*version_name*⟩] or akin package commands.

`LGRgreek`, `eulergreek`, `symbolgreek`: the Greek letters will be taken, respectively from the text font itself (which must be available in `LGR` encoding), or respectively the Euler or Symbol font.

`symbolmax`: all characters other than letters and digits, are taken from the Symbol font. This option also makes a number of further glyphs available, such as some basic mathematical arrows, and the sum and product signs. For documents with very simple needs in mathematical symbols, **`mathastext`** with option `symbolmax` may give in the end a PDF file size quite smaller than the one one would get without the package.[67]

`defaultmathsizes`: prevents **`mathastext`** from setting up, as it does per default, larger subscripts and superscripts in math mode, and from copying code from the moresize package[68] in order to redefine `\Huge` and define a `\HUGE` command.

---

[67] It is even better if compiled via `latex+dvipdfmx`.

[68] Christian CORNELSSEN, *Allows font sizes up to 35.83pt*, `https://ctan.org/pkg/moresize`.

3.2 Complete list of options

Some items are described succinctly as more developed descriptions were given earlier. They may sometimes simplify by omission and not consider all possible configurations, particularly those resulting from usage of the package commands in the preamble to configure math versions.

Note that this list in not in alphabetical order, the items are grouped roughly by themes. So, objectively, the best for the diligent reader is to read thoroughly all descriptions.

- basic: only mathastextify (ascii) letters and digits. Can be combined with either `nodigits` or `noletters`.

- subdued: acts in a subdued way, which means that the LaTeX "normal" (default) and "bold" (triggered by `\boldmath` or `\mathversion{bold}`, undone by `\unboldmath` or on exit from a scope limiting context such as an environment) math versions are left (not quite: check subsubsection 1.3.4 for specifics) unchanged and the `mathastext` action is triggered only when switching via `\MTversion{`⟨*version_name*⟩`}` (or its starred variant) in the document body to a version previously defined in the preamble via `\Mathastext[`⟨*version_name*⟩`]` (or alternative declarative interface such as `\MTDeclareVersion`). [69]

- italic: let the Latin letters (both lowercase and uppercase) use the italic shape (`\itdefault`) in math mode. If the package handles Greek letters, also lowercase (but not uppercase) Greek letters will use this a priori italic shape except if some other option such as `upgreek` was used.[70]

- frenchmath: configures the lowercase Latin letters to use italic shape (`\itdefault`), and uppercase Latin letters to be in same shape as for digits and operator names (i.e. a priori `\shapedefault`).

  If the package handles Greek letters both lowercase (if under control of `mathastext`, i.e. not for `selfGreek`) and uppercase Greek letters will use the same shape as operator names, except if some other option such as `itgreek` was used.[71]

> This configuration (i.e. that uppercase Latin letters will be in the same shape as the one for digits and operator names) is **not** undone in the `subdued` "normal" and "bold" math versions. It holds throughout the document.
>
> As a bonus, note that doing
>
> `\usepackage[basic,subdued,frenchmath]{mathastext}`

---

[69] Under this option `\MTversion{normal}` and `\MTversion{bold}` execute automatically `\MTmathoperatorsdonotobeymathxx`, `\MTeasynonlettersdonotobeymathxx`, `\MTnonlettersdonotobeymathxx`, `\MTmathstandardletters`.

[70] Since `1.3y`, in presence of the `LGRgreek` option in addition to `italic`, the `\MTgreekitdefault` shape is then used for lowercase Greek letters and `\MTgreekupdefault` for uppercase.

[71] Under `LGRgreek` and since `1.3y`, the `\MTgreekupdefault` is used for Greek letters if no other option such as `itgreek` was employed.

provides a simple manner to obtain the expected shapes of Latin letters in French mathematical typography, in an arbitrary math font configuration from other packages, in case those packages do not provide an option to achieve this.

But, even if **mathastext** is used via `LGRgreek` to configure Greek letters, on the other hand the control sequences for Greek letters are all really restored to their defaults (or whatever was configured by other packages loaded prior to **mathastext**) in the subdued "normal" math version, which limitates the usefulness of the previous paragraph.

On the bright side, the `\alphaup`, `\alphait`, ..., control sequences will however be with their **mathastext** meaning, see `LGRgreek` for more information.

It is *not* possible (except of course if one is ready to do some low-level TEX coding to re-execute where needed in the document body a few lines of the package internals with appropriate modifications; I said TEX, not LATEX, as the latter is very much decided to make impossible any kind of math configuration change at this level if not in the preamble) to achieve a "French math" style only in some math versions and not in others. The reason why is that to achieve distinct shapes for uppercase versus lowercase Latin letters, the uppercase letters are assigned internally to the font (which can change from math version to math version) used for operator names. One can still make them slanted using `\MTshape`, but this will also slant the digits, as they are picked from the same font. On the other hand if we do not use the `frenchmath` option, both uppercase and lowercase Latin letters are always assigned to the same font, so no math version can give them separate distinct shapes. For a small demo though, one can naturally painstakingly use either the `\mathrm` or `\mathnormal` alphabet commands to obtain, say under the `italic` option and no additional configuration, respectively the up shape and the italic shape.

None of the `frenchmath`, `frenchmath*`, and `frenchmath+` options bear any direct connection with the frenchmath package by Antoine MISSIER (this is in contrast with the fact that the `decimalcomma` option is directly related with the decimalcomma package by the same author as it tells **mathastext** to require it). But see subsubsection 1.4.13 for important information about the utility of `frenchmath*` if the two packages are to be used concurrently.

new behavior
- defaultalphabets: **mathastext** always defines `\Mathnormal`, `\Mathrm`, `\Mathbf` etc... to refer to the **mathastext**-ified text fonts, and redefines the math alphabets `\mathrm`, `\mathit`, `\mathtt` etc... (but not `\mathcal` of course) to use them. To avoid the remapping and keep the `\mathrm` et al. to refer to the non **mathastext**-ified fonts, use this option. The `\Mathnormal` et al. commands with an initial uppercase will always be available whether or not this option is made use of. (1.3za)

  Prior to 1.3za (and since 1.15f), this option also prevented the package to declare the `\Mathnormal` et al. and `\mathnormalbold` commands. In this context, recall that the dreaded "too many math alphabets" error can only occur on *use* in the document of too many of such commands, and not at the time of their declarations. The author's notes from time of 1.15f release (2012/10/25) only say that it may not be "useful" to package user to have both (for example) `\mathrm` and `\Mathrm`, which sounds weird if they are to acquire distinct meanings. So since 1.3za both will exist. In the default package configuration `\mathrm` is configured to expand to `\Mathrm` (with some extra behavior under `LGRgreek+`), and with this option or the `defaultrm` option `\mathrm` is kept with its original meaning (and the `LGRgreek+` extras do not work).

changed:
- defaultnormal, defaultrm, defaultbf, defaultit, defaultsf, defaulttt: tell **mathastext** to not set up, respectively, the `\mathnormal`, `\mathrm`, `\mathbf`, `\mathit`, (1.3za)

`\mathsf`, and `\mathtt` commands to use the mathastext-ified font which are accessible always via `\Mathnormal`, `\Mathrm`, `\Mathbf`, `\Mathit`, etc...

Prior to `1.3za` these options also prevented the creation of the corresponding **`mathastext`** command with an uppercased initial.

- ncccomma: it triggers the loading of the ncccomma package[72] and configures **`mathastext`** for compatibility (this is canceled if `nopunctuation` option is used, or `basic` as it implies it). *Note that* **`mathastext`** *has* NO *auto-detection mechanism of* ncccomma*, the correct way is to use the eponymous option.* (`1.3y`)

  The effect of the ncccomma package will apply to the entire document body, even to portions using the *normal* or *bold* math versions with **`mathastext`** having been loaded with the `subdued` option. Also, in case of usage of package `babel` with `french` option, the effect of ncccomma will also apply to those parts of the document using another language than French.[73]

- decimalcomma: it triggers the loading of the decimalcomma package[74]. The same remarks apply as for the `ncccomma` option. In particular *note that* **`mathastext`** *has* NO *auto-detection mechanism of* `decimalcomma`*, the correct way is to use the eponymous option.* (`1.3zb`)

- binarysemicolon: sets (except if `nopunctuation` is used) the semi-colon to let TEX use spacing of binary type, not punctuation type, around the semi-colon (it is often used in French mathematical typesetting as separator in interval denotations, when the extremities are decimal numbers, as the comma is used as decimal separator). (`1.3y`)

  The effect applies to all math versions, even the *normal* and *bold* math versions with **`mathastext`** having been loaded with the `subdued` option.

- CHANGED! frenchmath*: does all three of `frenchmath`, `decimalcomma` and `binarysemicolon`. (`1.3zb`)

  Prior to `1.3zb`, this option did what is now available via `frenchmath+`. The `1.3zb` change was made as a follow-up consecutive to the `2.7` release frenchmath. Indeed this option as explained in subsubsection 1.4.13 is provided as a compatiblity layer with frenchmath, and it was mandatory to modify its meaning to refer to package decimalcomma, not ncccomma, consecutive to the internal change of frenchmath at its `2.7` release to use decimalcomma.

- frenchmath+: does all three of `frenchmath`, `ncccomma` and `binarysemicolon`. This is what used to be called `frenchmath*` prior to `1.3zb`. (`1.3zb`)

---

[72] Alexander I. ROZHENKO, *Use comma as decimal separator in mathematics*, `https://ctan.org/pkg/ncccomma`.

[73] There is a ‘feature’ of babel-french that the effect of package ncccomma is canceled if one switches from French to English; and switching back to French does not reenact it. For background on this issue see `https://github.com/latex3/babel/issues/190`.

This does not apply to decimalcomma `1.3` or later.

[74] Antoine MISSIER, *Comma for decimal numbers*, `https://ctan.org/pkg/decimalcomma`.

- endash, emdash: use the text font en-dash (–) or even the em-dash (—, but this seems crazy) for the minus sign rather than -. `endash` option is default for the package.

- unicodeminus: use the `MINUS SIGN U+2212` (requires fontspec.) Or, in the form `unicodeminus=HHHH` with four *uppercased* hexadecimal digits: use the `U+HHHH` code point. As `noendash` really means "use the hyphen from the text font", `unicodeminus` remains without effect under it, or, naturally, under `nominus`. Without this option, **`mathastext`** uses the `EN DASH U+2013` by default for OpenType fonts. (1.3q)

- asterisk: this tells **`mathastext`** to replace the binary math operator $*$ and the control sequence `\ast` with versions which uses the text asterisk * suitably lowered, and with the correct spaces around it as binary operator. The amount of lowering[75] is decided by the mandatory argument to the command `\MTlowerast{`⟨*dimen*⟩`}`. The package does `\MTlowerast{.3\height}`. Using the `ex` unit as in `\MTlowerast{.5ex}` is not a good idea as it does not scale properly in the script and scriptscript styles.

  Attention that if using this option, inputs such as `$R^*$` or `$R^\ast$` raise errors and *must* be replaced by `$R^{*}$`, respectively `$R^{\ast}$`. The * is now 'mathematically active'[76] and `*` and `\ast` will obey the math alphabet commands (see subsubsection 1.8.4).

new: - activedigits: it makes all digits mathematically active! This is reserved to expert LaTeX users. See subsubsection 1.8.5 for an example. Do not use this option if you don't intend to make use of such techniques to achieve special effects at some location of your document at least. (I know this goes without saying, but passing this option and not using it is only adding overhead to all your equations with digits). (1.4)

- nodigits: no **`mathastext`**-ification of digits.

new: - noletters: This is quite antithetical to the package core aim, but here you go. (1.4b)

  This option implies `nohbar` and `defaultimath`.

  Note though that **`mathastext`** will still declare to LaTeX two extra symbol fonts and define alphabet commands such as `\Mathrm`, `\Mathnormal` and `\mathnormalbold` which map to them. But ascii letters will if not in arguments of these commands not use the **`mathastext`**-font and all business of active letters (see subsubsection 1.8.5) is now irrevocably inaccessible.

  If you simply want to avoid the overhead of math active letters, use rather `\MTmathstandardletters` in the preamble after having loaded the package (although this does not undo the fact that **`mathastext`** will have redefined what a letter does if it is catcode active but that remark should have been made in an even smaller typeface).

---

[75] With the option `symbolmisc`, the asterisk is picked from the Symbol font, and the amount of lowering is non-customizable; however if a math alphabet command is used, the asterisk is then again from a text font and the lowering will be as specified by `\MTlowerast`.

[76] In a hopefully safe way, for example `$\label{eq*1}$` is ok.

- nohbar: prevents **mathastext** from defining its own `\hbar`.

- noendash: the minus sign will be the - from the text font, not the en-dash –.

- nolessnomore: besides ! ? , . : ; + − = ( ) [ ] / # $ % & **mathastext** treats also < > | { } and \. Use this option to let it not do it. This is the default in case of `OT1`-encoding.

- further excluding options: noexclam ! ? nopunctuation , . : ; noplus, nominus, noplusnominus + − noequal = noparenthesis ( ) [ ] / nospecials # $ % & (and `nodigits` and `noletters` mentioned earlier).

- alldelims: true by default, means that the characters excluded by `nolessnomore` are treated. Use this option in case of a mono-width `OT1`-encoded font.

- nosmalldelims: this prevents **mathastext** from trying to pick up in the text font the 'small variants' of some math delimiters; it only affects what happens when a character such as a left parenthesis (or [is used as a delimiter, and in the event that TEX has chosen the smallest sized variant. This has no impact on what happens when they are not used as delimiters: then, and if not disabled by the corresponding options, these characters are always picked up from the text font.[77]

- symbolgreek, symboldigits: to let Greek letters (digits) use the Symbol font.

- symbolre: replaces `\Re` and `\Im` by the Symbol glyphs $\Re, \Im$ and defines a `\DotTriangle` command ($\therefore$).

- symbolmisc: takes quite a few glyphs, including logical arrows, product and sum signs from Symbol. They are listed *supra*. Doing `\renewcommand{\int}{\smallint}` will maximize even more the use of the Symbol font.

- symboldelimiters: the characters apart from letters and digits will be taken from the Symbol font.

- symbol: combines `symbolgreek`, `symbolre`, and `symbolmisc`.

- symbolmax: combines `symbol` and `symboldelimiters`.

- eulergreek, eulerdigits: to let Greek letters (digits) use the Euler font.

- LGRgreek: this configures the Greek letters in math mode to use the text font (i.e. a priori the font which was default at time of loading the package) in LGR-encoding. The command `\MTgreekfont` can be used to set a specific (LGR-encoded) font family. Each use of `\MTgreekfont` must be followed at some point

---

[77] in this very special situation of option nosmalldelims, the braces are an exception to this rule and they require both of `\MTnonlettersobeymathxx` and `\MTexplicitbracesobeymathxx` for being picked up from the text font when not used as delimiters.

by a `\Mathastext` or `\Mathastext`[⟨*version_name*⟩] to be effective. Any subsequent math version declaration will be influenced by it until `\MTgreekfont` is used again to configure another font for Greek letters.[78]

If `\MTgreekfont` is never used the font family for Greek under option `LGRgreek` will be, in all math versions except under `subdued` for the "normal" and "bold", the family which was the default at time of loading the package. You must use `\MTgreekfont` to change it.

See further on this topic the `upgreek`, `itgreek`, `upGreek` and `itGreek` options as well as the `\MTupgreek`, `\MTitgreek`, `\MTupGreek` and `\MTitGreek` commands.

It is up to the user to ascertain that the font family is indeed available in the LGR encoding; if it is not, only at time of the first math mode typesetting will LaTeX issue warnings such as this one:

```
Font shape `LGR/ptm/m/n' undefined
using `LGR/cmr/m/n' instead on input line 28
```

The `LGRgreek` option also triggers pre-definition of Greek character tokens such as `\alphaup` or `\betait`, see subsubsection 1.7.3 for the explanations.

Although under `subdued` option `mathastext` restores Latin (but see `frenchmath`) and Greek letters in the "normal" and "bold" math versions it still under `LGRgreek` option keeps in these "subdued" math versions the package declared `\alphaup`, `\alphait`, ..., and the associated `\mathgreekup` and `\mathgreekit` commands to access the underlying fonts, and also since `1.3za` `\mathgreekupbold` and `\mathgreekitbold`.

The font used by these math alphabet commands in the subdued "normal" and "bold" is either the one in `LGR` encoding which was the family default at time of loading the package or the one configured last by `\MTgreekfont` when the command `\Mathastext` (without optional argument) was used in the preamble.

`1.3za` fixes here a bug which froze the target font to be the one at time of loading the package: this bug applied (only) to the subdued "normal" and "bold" math versions and was not readily visible as there is a priori no reason to use in these subdued math versions these `mathastext`-provided Greek font alphabets.

- LGRgreeks: each declared math version will be supposed to be with a font which is also available in LGR-encoding. This is a shortcut to using `\MTgreekfont` systematically to keep in sync in all declared math versions the font for Greek with the font for Latin letters. Please note that macro `\MTgreekfont` becomes then inoperant, and if you need one math version without this Latin-Greek synching, you will have to use rather `LGRgreek` and then `\MTgreekfont` manually appropriately.
- LGRgreek+ and LGRgreeks+: they extend respectively `LGRgreek` or `LGRgreeks` `(1.3za)` to let Greek letters control sequences when in the scope of `\mathrm`, `\mathit`, and `\mathbf` behave as would be expected by LaTeX users who have not read `fntguide.pdf` or any other LaTeX documentation but have used unicode-math. See subsubsection 1.7.5 for details.

---

[78] You can check the documentation of the `https://ctan.org/pkg/lgrmath` package for how to find out systematically which fonts are available on your system in `LGR` encoding.

- selfGreek: this is for a font which is also available in `OT1`-encoding and contains the glyphs for the default eleven capital Greek letters.

  This option should have been named `OT1Greek` as it bears about the same relation with `OT1` encoding (for eleven capital Greek letters) as `LGRgreek` does with the `LGR` encoding (for the complete no-diacritics Greek alphabet).

- selfGreeks: each declared math version will be supposed to be with a font with the eleven capital Greek letters in its `OT1`-encoded version.

- upgreek, itgreek: options to tell **`mathastext`** to use `\MTgreekupdefault` or `\MTgreekitdefault` for the lowercase and uppercase Greek letters shape. These two commands can be defined prior to loading the package. This option is operant only under the `LGRgreek(s)` or `selfGreek(s)` options.

- upGreek, itGreek: influence only uppercase Greek.

- mathaccents: use the text font also for the math accents. As in vanilla LaTeX, they are taken from the font for the digits and `\log`-like names. Obey the alphabet changing commands.

- unimathaccents: extends `mathaccents` to OpenType fonts. Gave bad results in my brief testing. (1.3u)

- defaultimath: do not overwrite `\imath` and `\jmath` to use `\inodot` and `\jnodot`.

- defaultmathsizes: do not change the LaTeX defaults for the sizes of exponents and subscripts.

- fouriervec: provides a `\fouriervec` command. The user can then add in the preamble `\let\vec=\fouriervec`. There is also always available a “poor man” vec accent `\pmvec` for upright letters.

new: - everymath: this option tells **`mathastext`** to employ as in the `1.3` releases the `\everymath`/`\everydisplay` registers to store certain actions to be executed at each entrance into math mode. The main change with `1.4` is that mathcode changes done by user (or possibly via language changes in a multilingual babel context) in the document body, and which apply to those characters which **`mathastext`** used to handle as part of `\everymath`/`\everydisplay`, will now have an effect, whereas with earlier releases or this one with the `everymath` option, they may get overruled by **`mathastext`**. Use `everymath` for backward compatibility if needed (and *only* if needed). Notice though that regarding ascii letters, there is no return to the pre-`1.4`: they will not acquire their mathcode active status only at each entrance into math mode, even with this option, but once and for all at package loading time, if not `subdued`, or when using `\MTversion` to enter a non-`subdued` math version. (1.4)

  Please report breakages due to the `1.4` release, which made it necessary for you to use this option `everymath` as workaround.

The option is destined to be removed at next major release and it will warn the user about this.

Thanks to Kevin KLEMENT, Tariq PERWEZ and Ricard TORRES for sending bug reports and feature requests when the first version of the package was issued.

Numerous examples will be found there:
`http://jf.burnol.free.fr/mathastext.html`
`http://jf.burnol.free.fr/showcase.html`

4 Change log

1.4e [2024/10/26]

∗ Fix a bug dating back to `1.3y` which caused under **`LGRgreek`** or **`LGRgreeks`** options an `\MTversion{normal}` to raise an `Extra \else` error (not if **`subdued`**). The unit test which could have shown this had been left aside at `1.4d` because it was so old that some matters unrelated to the package prevented its immediate use... alas...

∗ Fix one more `1.4` regression: the (not really recommended) option **`unimathaccents`** was broken. Again a test file existed but it had been left apart from those automatically built, for some unknown reason...

1.4d [2024/10/26]

∗ Fix one more `1.4` regression (the worst one by far): `$f''$` caused a crash. Fortunately, the **`everymath`** option which re-enacts legacy code could be used as a *temporary emergency workaround.* Thanks to Enrico Gregorio for report.

∗ The `1.4b` option **`noletters`** did not prevent the package from defining math mode symbols `\inodot` and `\jnodot`. It now does.

∗ Fix a longstanding legacy bug which would have caused a crash if **`\MTnonlettersobeymathxx`** had been used with XƎLaTeX in a document with the character `-` being a Babel shorthand.

∗ Usage of **`\Mathastext`** or **`\MTDeclareVersion`** to declare a math version associated with an exotic non-text font encoding such as `OML` is theoretically possible (although of dubious practical use except for stress tests) but was broken since `1.3u` due to internals relying on LaTeX commands associated with text font encodings (for example to define `\inodot` using `\i`, or the `\hbar` with the help of the accent command `\=`) and which are not available in such context. This release handles such situation gracefully via warnings, rather than causing low level errors.

∗ Improve looks in the PDF of this Change log.

1.4c [2024/10/21]

∗ Fix a regression at `1.4`: under XƎLaTeX, loading **`mathastext`** caused an error if **`unicode-math`** and its `\setmathfont` had been used prior ("*Extended mathchar used as mathchar*" in relation to the minus sign character). Thanks to Michael Roland for report.

Keep in mind though that the documentation (see 1.4.15 Unicode engines) has always said that the package "*is expected to be most definitely incompatible with unicode-math*".

∗ Fix another regression at `1.4`, related to the same code and also avoided with **`everymath`** option, but showing with all engines. It caused **`\MTnonlettersdonotobeymathxx`**, if used, to reset the minus sign to its status as prior to the loading of **`mathastext`**, whereas it should have used the (by default) mathversion dependent one (which defaults to the en-dash in the font encoding as stored by **`mathastext`** in the extended math version).

∗ **`\Relbar`** and **`\relbar`** redeclarations (needed as they are used by LaTeX for arrows, and the equal and minus signs if picked from the text font may not work well in-there) are now done via **`\DeclareRobustCommand`** also with package **`amsmath`** to match the current behavior of the latter. Note though that they will use the equal and minus signs as in place at package loading time, the documentation does say that **`mathastext`** should be loaded last.

1.4b [2024/07/27]

∗ Fix a regression at `1.4` regarding option **`nodigits`**.

∗ New option: **`noletters`**. *I am in old age now so I can provide an option completely antithetical to my life work.*

∗ Option **`everymath`** which is destined to be removed at next major release warns user about it. As major releases tend to happen once per decade I think the constant nagging will become insufferable and you will adapt to the `1.4` changes rather than persisting into using this option.

1.4a [2024/07/20]

∗ There was a documentation glitch in 1.4 and also a problem with the **`\MTprimeskip`** feature being lost under the emergency fall-back **`everymath`** option.

1.4 [2024/07/20]

∗ Since 1.2 of 2012/12/20, mathastext has used *mathematically active* characters to propose certain advanced functionalities. For reasons half lost in the mists of time but whose main one was surely to keep the meaning of the active shape of characters unchanged outside of math mode, this mathematical activation, and (in most cases) the definitions of what active characters do, were done again at *each* entrance into math mode. At this 1.4 release, mathastext does *not inject any code whatsoever* into the `\everymath` and `\everydisplay` toks registers anymore (*except* for one font-related hack needed under LuaLaTeX, see below). Your documents will compile a tiny bit faster.

∗ In (*unsual*) documents where users play with catcodes and mathcodes it is *impossible* to keep exact backward compatibility, because documented user commands which acted formerly as toggles with delayed action now will enact changes immediately if in the document body. In practice consequences are expected to be few, because catcode active characters are (as was already the case with earlier releases) hacked only when they are Babel shorthands and they are then modified in a way altering only their action in math mode. The precise description of what mathastext does when mathematically activating (or not) a character, depending on circumstances, is to be found among small-print comments in the section "Extended scope of the math alphabets commands". See also the documentation of the `\MTmathactiveletters` command for some specifics regarding ascii letters.

∗ New option: `everymath`. It instructs mathastext to revert (partially) to its legacy code which uses `\everymath/\everydisplay`. This reversal is partial, the handling of ascii letters not being included into it. The `everymath` option is there *only to try as a quick fix* in case transition to this release causes a major problem in a user document and time is lacking to investigate. *Please report to the author such issues.* Option `everymath` is destined to be removed at next major release.

∗ New option: `activedigits`. Enjoy.

∗ It is now easier to hook into the mathastext architecture for mathematically activated ascii letters. See the new section "Hacking letters (and even digits) for special tasks".

∗ Bugfix: do not override special behavior of the math mode dot in babel-spanish.

∗ Bugfix: A *desperate* hack related to LuaLaTeX font matters and dating back to 1.3o 2016/05/03 had been for some years in dire need of an update regarding fonts using `Renderer=HarfBuzz`. This is done now. Thanks to tex.sx `user691586` for report. This is currently the sole remaining usage of `\everymath/\everydisplay`.

∗ Bugfix: `\MTexplicitbracesobeymathxx` (which is related to `\{` and `\}`) was without effect since an upstream LaTeX change at its 2020-02-02 release.

∗ With option symbolmisc, those math symbol macros formerly defined via `\DeclareRobustCommand` are now declared via `\protected\def`.

∗ Removal of legacy branches previously kept to support LaTeX earlier than 2020-02-02.

∗ Removal of support for EU1 and EU2 font encodings.

∗ Option `noasterisk` deprecated at 1.2d 2013/01/02 has (finally) been removed.

∗ Four test files previously included and auto-extracted from the distributed dtx have been dropped. One of them is still available on the package homepage.

∗ Some parts of the documentation have been massively re-ordered and even to some extent improved. But there may be some occasions where obsolete statements will be found having the legacy `\everymath/\everydisplay` situation as background.

1.3zb [2023/12/29]

∗ Update to the `frenchmath*` option to maintain compatibility with the [frenchmath](https://ctan.org/pkg/frenchmath) package whose release 2.7 (2023/12/23) has replaced the ncccomma package by the decimalcomma package.

∗ The `frenchmath+` option holds the former meaning of `frenchmath*`.

∗ Option `decimalcomma` to load the eponymous package by Antoine Missier. This is tacitly done by `frenchmath*`.

∗ No more messages sent to the console output during loading, only info messages going into the log, and using (more or less) the official LaTeX interface: after close to 13 years of development of this package it was perhaps finally the time to do it.

* Documentation improvements. Close to 13 years after the birth of the package, and as it nowadays rarely wakes up from dormancy, this was almost last chance to try to improve a few things.

1.3za [2023/12/20]

* Under `LGRgreek` and `LGRgreeks` options, new math alphabets `\mathgreekupbold` and `\mathgreekitbold`.

* New options `LGRgreek+` and `LGRgreeks+`. Thanks to Holger Gerhardt for feature request and code ideas. Please find and read the relevant documentation in the PDF.

* The meaning of `defaultalphabets` and related individual options such as `defaultbf` has been modified (reverted to pre `1.15f` release): even under these options, the package always creates `\mathnormalbold`, `\Mathnormal`, `\Mathrm`, `\Mathbf` etc..., commands. This may break documents which used these options in order to reserve these command names. This was done with some hesitancy, but for the sake of internal logical coherence.

* Fix an obscure bug with no real consequences regarding interaction of `subdued` with `LGRgreek` and `\MTgreekfont`. See the `LGRgreek` documentation in the complete list of options for details.

* Fix long-standing hyperlink problems in the documentation: blue color words should now all be functioning hyperlinks.

1.3z [2023/09/01]

Fix 1.3y regression which broke `selfGreek` option due to internal renamings. Thanks to Stephan Korell for report.

1.3y [2022/11/04]

(the 1.3x had an annoying documentation bug, and had already been pushed to CTAN, hence the version increase to 1.3y)

* mathastext now requires the `\expanded` primitive (which is available with all major engines since TeXLive 2019).

* Revisit parts of the documentation (mainly the Examples, and the section on Greek letters) and shuffle the other parts to surely improve things. Mention the [mathfont](https://ctan.org/pkg/mathfont) and [frenchmath](https://ctan.org/pkg/frenchmath) packages.

* Add the `ncccomma` option which loads the [ncccomma](https://ctan.org/pkg/ncccomma) package to allow the comma as decimal separator.

* Add the `binarysemicolon` option to let the semi-colon be of type `\mathbin`, not `\mathpunct`.

* Add the `frenchmath*` option which does all three of `frenchmath`, `ncccomma` and `binarysemicolon`.

* Under the `LGRgreek` and `LGRgreeks` options only:
- make available upright and italic Greek letters in math mode via `\alphaup`, `\alphait`, ... control sequences, in addition to those not using such postfixed-names.
- add `\mathgreekup` and `\mathgreekit` math alphabets.
- add `\MTgreekupdefault` and `\MTgreekitdefault`. The former replaces `\updefault` which was used in some places and since LaTeX `2020-02-02` caused systematic Font Warnings about the substitution of `up` by `n`.

These new features required an extensive internal refactoring which is expected to not induce changes to most existing documents. But it may induce changes to those using some unusual configuration in the preamble, as made possible via the package macros; this can apply only to documents authored by those few people who actually read the documentation. For full details make sure to read the PDF documentation about this change.

* Fix "`\Digamma` under `LGRgreek` option uses the shape for lowercase not uppercase Greek".

* Fix some incongruities in log messages related to Greek letters and emitted during math version creation in the preamble.

1.3w [2019/11/16]

* LaTeX `2019-10-01` release (up to patch level 3 inclusive) together with `amsmath` conspired `:-)` to break `mathastext`, in connexion with math accents. This has been fixed upstream, but I am re-

leasing nevertheless a hot fix to this https://github.com/latex3/latex2e/issues/216 issue (this is compatible with future LaTeX releases).

∗ Fix: the `\hbar` is originally a robust command but becomes a `\mathchardef` token if (e.g.) `amsfonts` is loaded and then with recent LaTeX `\hbar<space>` is made undefined and `mathastext` definition of it remained without effect. The `\mathastext` own `\hbar` is now defined `\protected`.

∗ Fix: option `noendash` (or `symboldelimiters` which implies it) caused (since `1.3u`) a bug under Unicode engines when setting up the minus sign.

∗ Version names declared via the optional argument of `\Mathastext` or as first argument of `\MTDeclareVersion` must not be `normal` or `bold`. Enforce that! (this was marked as a bug to fix since `2012/10/24`...)

1.3v [2019/09/19]

∗ LaTeX 2019-10-01 release has made more math macros robust. This applies in particular to the math accents and to the `\hbar`. This required for mathastext to adapt. Also `\leftarrowfill` and `\rightarrowfill` are now defined robust by the kernel, hence mathastext does the same. These changes are dropped if mathastext detects an older LaTeX format.

∗ These LaTeX kernel changes motivated an examination of some redefinitions done (optionally) by mathastext:

- The user math alphabet macros got redefined as expanding to some other (robust) math alphabet macros, but were not robust in the strict sense. This does cause some issues for moving arguments in the context of multiple math versions, hence it was a bug. The special behavior of the math alphabet commands (they redefine themselves and other macros on first use) makes is somewhat problematic for mathastext to keep them updated across math versions and at the same time strictly LaTeX 2ε robust. Thus mathastext now requires the e-TeX primitive `\protected` and uses it for the definitions of the user level math alphabet macros.

- There are a number of `\mathchardef` tokens which (under certain options and/or configuration via the package user interface), mathastext redefines as macros. These macros cause no issue in moving arguments (they are not "fragile"), still it is probably better if they expand only at the time of typesetting. To this effect they are now also `\protected`: `\exists`, `\forall`, `\colon`, `\setminus`, `\mid`, `\prod`, `\sum`, `\imath`, `\jmath`.

- The macro `\vert` (which expands to a `\delimiter`) is now defined robust by LaTeX. Its mathastext redefinition is a `\protected` one rather.

- The `\{` and `\}` (which get redefined only under `\MTexplicitbracesobeymathxx` regime) are now strictly robust in the LaTeX 2ε sense (formerly they were `\let` to some robust macros, and this did not make them strictly LaTeX 2ε-robust entities).

∗ The various changes in mathastext described in the previous item apply independently of the LaTeX release version. The LaTeX format itself requires the e-TeX extensions since 2015.

1.3u [2019/08/20]

∗ new feature: the initial release dealt with only one font, and although shortly thereafter the 1.11 version added support for extended math versions, it was documented that some font-dependent set-up (minus as endash, dotless i and j, hbar, math accents) was done only once. This release makes the relevant characters font encoding savvy in each mathastext-extended math version. Thus, they should render correctly even with multiple math versions using fonts with varying encodings.

This reinforces importance of using `\MTversion` and not the LaTeX `\mathversion` when switching to a new math version (which got declared via the package interface). The implementation is compatible with Unicode engines and mixed usage of `TU` encoding (OpenType fonts) with traditional 8bits TeX font encodings. For all engines, all used (8bits) encodings must have been passed as options to the `fontenc` package.

Thanks to Falk Hanisch for feature request and code suggestions.

∗ new option `unimathaccents`: this adds to option `mathaccents` the demand to use the text font accents for OpenType fonts in math mode via the `\Umathaccent` primitive. Indeed, as my testing showed that this gave non-satisfactory results both with XeTeX and LuaTeX regarding the horizontal placement of the accents, the main option `mathaccents` acts only on 8bits encoded fonts.

∗ bugfix: the `\Mathastext` without optional argument forgot to repeat some font-encoding dependent initialization set-up done originally during package loading.

∗ bugfix: under the `subdued` option macros `\MTmathactiveletters` or `\MTnonlettersobeymathxx` now act like no-ops if issued explicitly while in the `normal` or `bold` math version. Formerly, this was not the case and could cause bugs such as a disappearing minus sign in math mode.

∗ bugfix: the letter `h` used in the `\hbar` obeyed the extra skips as set-up by `\MTsetmathskips`, badly interfering with the horizontal positioning of the bar accent. They are now ignored (as well as the added italic correction).

1.3t [2018/08/22]

∗ bugfix: the 1.3s bugfix about `subdued` compatibility with `fontspec` was deficient.

∗ bugfix: very old (v1.2, 2012/12/20) bug causing low-level TeX error during package loading (with pdflatex) when setting up the math minus sign to be the text font endash character, in cases with `\encodingdefault` other than OT1, T1 or LY1, e.g. something like T2A.

∗ `\imath` and `\jmath` obey the `subdued` regime. And the minus sign is now handled especially to ensure perfect compatibility with the `subdued` option.

∗ breaking change: `mathastext` does not redefine anymore `\i` and `\j` to let them be usable both in text and math mode.

1.3s [2018/08/21]

∗ fix to an issue with `subdued` option in a `fontspec` context.

1.3r [2016/11/06]

∗ documentation tweaks.

1.3q [2016/10/31]

∗ new option `unicodeminus`.

∗ the `Recent Changes` section of the documentation has been removed as it was a duplicate of information available in the `Change Log`.

∗ some other changes in the documentation, in particular the use of straight quotes in verbatim.

1.3p [2016/05/13]

∗ bugfix: release `1.3n` had forgotten to activate by default its new customization of the amsmath macro `\newmcodes@` (it was done from using `\MTversion` in the document body but not by default at start of body.)

∗ public name `\MTfixmathfonts` for a `1.3o` macro.

1.3o [2016/05/03]

∗ `mathastext` fixes an issue related to a feature of LuaLaTeX and `luaotfload` that OpenType fonts are declared in one of two modes: `node` and `base`, and only the latter is functional in math mode. But by default text fonts are declared in mode `node`. Thus `mathastext` now intervenes to make it so that the font it declares in math mode will use mode `base`. This fixes issues with for example old style figures being used while the text font used lining figures (or vice versa, depending on the font). But see the code comments for more.

1.3n [2016/04/22]

∗ at long last, `mathastext` takes care properly of annoying and perplexing amsmath's `\newmcodes@`. The very recent change in `amsopn.sty` finally made it compatible with Unicode engines, but anyhow, `mathastext` must do its own patch to use the correct font. All of this taking into account the various options passed to the package. Lots of trouble for a tiny thing.

1.3m [2016/04/02]

∗ minor code maintenance before annual TL freeze.

1.3l [2016/01/29]

∗ compatibility with fontspec's upcoming switch from `EU1/EU2` to `TU` common to both Unicode engines.

1.3k [2016/01/24]

∗ typos fixed in the documentation. In particular, the README link to the package homepage had remained broken from day one of the package releases: `mathastext.html` therein was misspelled as `mathsastext.html`! (but the pdf documentation had the correct link; as well as the CTAN catalogue).

1.3j [2016/01/15]

∗ renamed and modified recent `1.3i`'s

`\MTactivemathoff` into `\MTeverymathoff`. Added `\MTeverymathdefault`.

* `subdued` mode is a bit stronger: also the asterisk reverts to the default (if it was modified due to option `asterisk`), the added extra `\mskip`'s (useful with upright fonts) for `'`, `\exists`, and `\forall` are suppressed rather than re-configured to use `0mu`. Related new commands `\MTexistsdoesskip`, `\MTforalldoesskip`, `\MTprimedoesskip`, `\MTnormalexists`, `\MTnormalforall`, `\MTnormalprime`.

* the toggle for using mathematically active letters is only emitted once during package loading; the `\Mathastext` command does not do it anymore; the use in the preamble of `\MTmathstandardletters`, or `\MTnoicinmath` and related commands is not overruled by later use of `\Mathastext`.

* quite a few documentation improvements and rewrites, particularly in the description of commands which are related to the modifications of mathcodes (mainly for math activation of characters or letters) as done by mathastext at `\everymath` or `\everydisplay`.

1.3i [2016/01/06]

* `\url` from `url.sty` as well as `\url` and `\nolinkurl` from `hyperref.sty` use math mode and (by default) the monospace text font. To avoid mathastext overwriting the special preparation done by `{url,hyperref}.sty` the commands `\url/\nolinkurl` are patched to do automatically `\MTactivemathoff` (now `\MTeverymathoff`) before entering math mode.

* the extra skips specified by `\MTsetmathskips` are not inserted around letters if inside the arguments of math alphabet commands, or within operator names.

* the added explicit italic corrections (for non-oblique fonts) were disabled within math alphabet scopes, except `mathnormal`; they are now disabled within all math alphabets, inclusive of `mathnormal`.

1.3h [2015/10/31]

* bugfixes: since `1.3d 2014/05/23` the option `symbolgreek` caused `\ell` to become undefined, and, similarly but far worse, options `selfGreek`, `selfGreeks` caused all lowercase Greek letters `\alpha`, `\beta`, etc.. to become undefined.

1.3g [2015/10/15]

* following `2015/10/01` LaTeX release, removal of the `"luatex"` prefix from the names of the LuaLaTeX math primitives. Compatibility maintained with older LaTeX formats.

1.3f [2015/09/12]

* the replacement of amsmath's `\resetMathstrut@`, when it is done, emits an Info rather than a Warning as this could be potentially stressful to some users.

* the `README` self-extracts from the `dtx` source, as a text file `README.md` with Markdown syntax.

1.3e [2015/09/10]

* bugfix: under option nosmalldelims, `\lbrace` and `\rbrace` were redefined as math symbols and could not be used as delimiters.

1.3d [2014/05/23]

* A 2015/02/26 edit to the documentation mentions the improved compatibility of mathastext with the latest (3.34) beamer release: no more need for `\usefonttheme{professionalfonts}`.

* new commands `\MTstandardgreek` and `\MTcustomgreek`.

* The Greek letters, in case of use of one of the package related options, are left to their defaults in the normal and bold math versions if the subdued option was also used (this was so far the case only with options LGRgreek/LGRgreeks).

* `\newmcodes@` of amsmath is left untouched if package lualatex-math is detected.

1.3c [2013/12/14]

* added a starred variant to `\MTversion` which tells mathastext to only do the math set-up and not modify the text fonts.

* added second optional version name argument to `\Mathastext` and to `\MTDeclareVersion`, to transfer settings for things not otherwise changed by mathastext from a math version to the one declared. This is mainly for symbols and large symbols to be the bold ones when the user sets up the series of a mathastextified font to be bold in a mathastext-declared version.

* renamed `\defaultprod` to `\MToriginalprod`, `\defaultsum` to `\MToriginalsum`, (this is in case of option symbolmisc).

* changes to the dtx organization; options for generating the documentation can be customized in generated mathastext.tex file.

* 1.2d code for `\#`, `\$`, `\%`, and `\&` modified erroneously the earlier correct 1.2c code and created a bug showing up with more than 16 math families (a possibility only with lualatex or xelatex).

1.3a [2013/09/04]

* the somewhat silly `\string`'s are removed from the `\MTsetmathskips` command of release 1.3, thus allowing its first argument to be a macro, or any expandable code, giving a letter.

* the amsmath `\resetMathstrut@`, which is incompatible with a mathematically active parenthesis (is now modified only if necessary (i.e.\@ only when `\MTnonlettersobeymathxx` is issued) and is restored to its original value if not needed anymore (i.e. after `\MTnonlettersdonotobeymathxx`, as for example when switching to the normal version under option subdued).

* improved documentation.

1.3 [2013/09/02]

* commands `\MTsetmathskips` and `\MTunsetmathskips` added.

* commands `\MTmathactiveletters` and `\MTmathstandardletters` to govern the math activation of letters independently of its use for insertion of the italic corrections (`\MTicinmath` and `\MTnoicinmath` correspondingly modified).

* the new `\luatexUmathcodenum` as available since TL2013 allows identical treatment by mathastext of = and - under both LuaTeX and XeTeX.

* `\newmcodes@` of amsmath is left untouched in case of option basic.

* a sentence containing | which was written to the log during the loading caused a problem if | was active (typically if `\MakeShortVerb`{\|} was added to the preamble prior to the loading of mathastext).

* some preemptive measures taken regarding things such as `\mid`, `\lbrace`, and `\rbrace`, as some packages define these things in manners which made the re-definitions done by mathastext issue errors.

1.2f [2013/01/21]

* minor code improvements. Change log added to the user manual.

1.2e [2013/01/10]

This version should be the last one in the 1.2 series as it seems to correct most of the main problems which were introduced with the massive use of mathematically active characters in versions 1.2 and 1.2b.

* It is indeed a thorny point when one wants to modify an active character in math mode only (without breaking usage in label's and ref's for example). The package now does that _only_ if the activation originated in the Babel system as it is then possible to modify appropriately the Babel macros `\user@active<char>` and `\normal@char<char>`, at the time of entering math mode (mathastext does all its activation job at `\everymath` and `\everydisplay`).

The relevant issues are discussed in section 2.10 of the user manual, in the test file mathastexttestalphabets.tex, and in the source code comments for macro `\mst@mathactivate`. The inherent incompatibility of Babel with packages having made mathematically active the characters itself makes document active is circumvented by this interference of mathastext. A generally applicable Babel patch could be derived from the method used by mathastext.

For the non catcode active characters, mathematical activation is used. This is done at the entrance in math mode.

* Sadly, the feature of added italic corrections introduced in version 1.2b did not behave as described in the user manual, due to forgotten group braces. Fixed.

* The command `\MTlowerast` from the user manual of v1.2d was not the one implemented in the source code. Fixed.

* The test files automatically extracted from a latex run on the dtx file have been revised and extended.

* The code is better documented.

1.2d [2013/01/02]

* an incompatibility with amsmath (its macro `\resetMathstrut@`), exists since version 1.2 of the package. This is fixed here.

∗ various improvements in dealing with the asterisk and in the mechanism of letting non-letter symbols obey the math alphabet commands.

∗ the `noasterisk` option is deprecated and made a no-op.

∗ documentation extended and improved.

1.2c [2012/12/31]

∗ mathastext now inserts automatically after all (latin) letters in math mode their italic corrections, if the font used is upright (sic). This improves the spacings for the positioning of subscripts. The feature is de-activated inside the math alphabets commands (apart from `\mathnormal`), so as to not prohibit the formation of ligatures.

∗ the documentation has been extended to explain in detail the issues which are relevant to the new feature of added italic corrections.

∗ version 1.2 had some bad bugs when confronted to active characters. This is corrected and additionally `\MTnonlettersdonotobeymathxx` is made the default, as the user input is too much constrained in its absence.

∗ a less fatal, but still annoying, typo had made the dot in 1.2 of type `\mathpunct` rather than `\mathord`.

∗ the inner namespace has been rationalized a bit.

1.2 [2012/12/20]

∗ a new command sets up the amount of space to be automatically inserted before the derivative glyph (useful when using an upright font).

∗ the scope of the math alphabets has been extended to apply to the non-alphabetical characters, and also to operator names.

∗ the format of the dtx file has changed. The package file is self-extracting from the dtx, and four additional test files are also produced during `latex mathastext.dtx`.

1.15f and 1.15g [2012/10/25]

∗ `\$`, `\#`, `\&`, and `\%` had been re-defined by mathastext since its inception in a rather strange (but working) way, which could cause surprises to other packages. Fixed.

∗ the subdued mechanism for the math alphabets is implemented in a simpler and more efficient manner than in 1.15e.

∗ the `defaultxx` options act a bit differently, and are more useful in case of a `too many math alphabets` situation.

∗ various improvements in the documentation.

∗ general clean up and better commenting of the source code.

1.15e [2012/10/22]

∗ new user commands to specify skip or glue to be inserted after the math symbols `\exists` and `\forall`

∗ complete (user transparent) rewrite of the code implementing the subdued option; and its action has been extended to apply also to the `\mathbf`, `\mathit`, `\mathsf`, `\mathtt` alphabets and not only to `\mathrm` and `\mathnormal` as in the previous versions.

∗ improvements in the documentation.

1.15d [2012/10/13]

∗ the Unicode situation is now correctly treated, throughout the code (this had been left in a half-done way from version 1.14 of April 2011).

∗ this includes an issue related to amsmath and its DeclareMathOperator macro which has been fixed,

∗ and the code related to `\relbar` and `\Relbar` (and `\models`) has been revised.

1.15c [2012/10/05]

∗ it is now possible to use distinct fonts in LGR encoding for the Greek letters according to the current math version.

∗ improvements to the documentation.

1.15b

∗ corrected a 'feature' of 1.15 which was backward-incompatible

∗ improvements to the pdf documentation

1.15 [2012/09/26]

∗ the subdued option allows the mathastextification to act only locally.

∗ some measures taken to deal with amsmath related issues when using xetex or luatex.

1.14c

* a bug is fixed: the `\Mathastext` macro reinitializes the fonts in the normal and bold math versions, but it also erroneously redeclared the math alphabet changing commands which could have been set up in previously defined math versions (via earlier calls to `\Mathastext`\[version_name\]).

1.14b [2011/04/03]

* there was a bug with `\$`, `\#`, `\&`, `\%` in math mode which showed up when ten or more math families had been declared. This bug affected also the minus sign under the same circumstances, when Unicode engines were used. Fixed.

* the options LGRgreek and selfGreek act now a bit differently, and new options LGRgreeks and selfGreeks have been defined.

* I also cleaned up a bit the code, for a more structured namespace.

1.14

* mathastext now modifies also the math alphabets `\mathit`, `\mathsf` and `\mathtt`, thus making it a quite generic complete manner to adapt the math configuration to fonts provided with no math support.

1.13d

* new macros `\MTstandardgreek` and `\MTcustomgreek`

1.13b

* when the Symbol font is used for `\prod` and `\sum` this will be only for inline math; display math will use the default glyphs

1.13 [2011/03/11]

* the LGRgreek option is added.

* internal changes for better readability of the code.

1.12

* various bugs have been corrected.

* the endash and alldelims options are active by default.

* the package is more Unicode aware.

* the `\Mathastext` command has been improved to facilitate the mechanism of math versions also when using XeTeX or LuaTeX (with package fontspec.)

* the en-dash and dotless i and j now work with all encodings, Unicode inclusive.

1.11 [2011/02/06]

* optional argument to `\Mathastext` macro.

1.1 [2011/02/01]

* options italic and frenchmath.

1.0 [2011/01/25]

* Initial version.

5 Implementation

The comments are a kind of palimpsest. Indeed they are not destined to the reader but to the author: when coming back to the source code perhaps years after having last looked at it, all bits of past information even if obsolete are useful. Not only comments but perhaps also some ancient parts of the code itself are a bit strange (the author hardly new any LaTeX at that time).

For about the same reason, there may be some long macro names which do not fit in the margins and that one sees only partially. The author sometimes has used a workaround to hyphenate, but not systematically. Life is time-limited. At `1.4` large chunks of code have been re-ordered but the global architecture is still somewhat of a mess and doing some such large diffs makes it sometimes difficult to follow small bits of code across commit history, so this is done reluctantly.

The usual catcode regime for letters and digits is assumed and some characters such as `*`, `` ` ``, `"`, `=` are supposed to be of catcode other at the time of loading of **`mathastext`**. The source of **`mathastext`** takes precautions for some other characters such as the right quote `'`, which may thus be active with no harm at the time of loading (note added `2024/07/16`: very hard to understand why I was so paranoid, but I have kept this annoying constraint).

By the way, I think LaTeX2e should have provided to authors a standard macro to be used at the beginning of a style file to make sure the catcodes are standard. Shorthands created by Babel should be mostly no problem as Babel does the activation only at the `\begin{document}`.

`1.4` removes a few code branches still there for support to old LaTeX and requires LaTeX `2020-02-02` (which made `\{` and `\}` `\protected`; earlier release `2019-10-01` had math macros such as the math accents and `\hbar` robust and so far we kept supporting both new and old contexts).

```
\NeedsTeXFormat{LaTeX2e}[2020/02/02]
\ProvidesPackage {mathastext}
  [2024/10/26 v1.4e Use the text font in math mode (JFB)]
```

`1.3zb` avoids writing **`mathastext`** info messages also to console output, only log file. Make prefix occupy 20 not 25 characters for alignment with `LaTeX Font Info`, as the latter often issues info messages. For similar reason the usages of `\PackageInfo` will be done with empty lines above and below for better visual separation from the voluminous output of the LaTeX font system.

```
\def\mst@infoline#1{\immediate\write\m@ne
                    {(\space\space\space mathastext:\space\space\space) #1}}
\immediate\write\m@ne{}
\PackageInfo{mathastext}{Starting the math mode configuration\@gobble}
```

Testing for XeTeX and LuaLaTeX.

`1.3g` 2015/10/15: update for the naming of primitives, the situation has evolved both on XeTeX side and on the LuaLaTeX side (LaTeX base `2015/10/01`): I was told "U" named math primitives were always available for LuaLaTeX. For XeTeX, the XeTeX prefix got replaced by U prefix with 0.99.. a certain number of 9. I opted for rather simple approach of just trying the "modern" names and if they don't exist fall back on earlier (and in danger of being deprecated) names.

```
  \let\mst@Umathcharnumdef\Umathcharnumdef
  \let\mst@Umathcodenum   \Umathcodenum
  \let\mst@Umathcode      \Umathcode
  \let\mst@Umathchardef   \Umathchardef
  \let\mst@Umathaccent    \Umathaccent
\newif\ifmst@XeTeX
```

```
\ifx\XeTeXinterchartoks\@undefined
  \mst@XeTeXfalse
\else
  \mst@XeTeXtrue
  \ifx\mst@Umathcharnumdef\@undefined
    \let\mst@Umathcharnumdef\XeTeXmathcharnumdef
    \let\mst@Umathcodenum   \XeTeXmathcodenum
    \let\mst@Umathcode      \XeTeXmathcode
    \let\mst@Umathchardef   \XeTeXmathchardef
    \let\mst@Umathaccent    \XeTeXmathaccent
  \fi
\fi
\newif\ifmst@LuaTeX
\ifx\directlua\@undefined
  \mst@LuaTeXfalse
\else
  \mst@LuaTeXtrue
  \ifx\mst@Umathcharnumdef\@undefined
    \let\mst@Umathcharnumdef\luatexUmathcharnumdef
    \let\mst@Umathcodenum   \luatexUmathcodenum
    \let\mst@Umathcode      \luatexUmathcode
    \let\mst@Umathchardef   \luatexUmathchardef
    \let\mst@Umathaccent    \luatexUmathaccent
  \fi
\fi
\newif\ifmst@XeOrLua
\ifmst@LuaTeX\mst@XeOrLuatrue\fi
\ifmst@XeTeX \mst@XeOrLuatrue\fi
```

`1.4`. I only checked this is correct with TL2024. The macro will be used with `#1` a catcode 11 or 12 token or a one character control sequence such as `\#`.

`1.4c` adds some definitions to avoid separate branches for non-Unicode vs Unicode.

```
\ifmst@XeTeX
  \def\mst@activemathcodenum@space{"1FFFFF }%
  \let\mst@mathcodenum\mst@Umathcodenum
\else
\ifmst@LuaTeX
  \def\mst@activemathcodenum@space{"1000000 }%
  \let\mst@mathcodenum\mst@Umathcodenum
\else
  \def\mst@activemathcodenum@space{"8000 }%
  \let\mst@mathcodenum\mathcode
\fi\fi
\def\mst@OnlyIfNotMathActive#1{%
    \ifnum\mst@mathcodenum`#1=\mst@activemathcodenum@space
         \expandafter\@gobble
    \else\expandafter\@firstofone
    \fi
}
```

1.2: all inner macros of **mathastext** now starts with `\mst@` for a cleaner name-space.

1.3l 2016/01/29: hmmm... at this late stage where nobody would expect me to still look at the code, I have found at least two macros which still didn't: `\do@the@endashstuff` and `\do@the@emdashstuff`.

Ok, doing something more serious: compatibility with upcoming TL2016 fontspec and its switch to `` `TU' `` NFSS font encoding in replacement of `` `EU1/EU2' `` Anyhow, the code in **mathastext** has been common to the two Unicode engines for a while, hence it is not hard to adapt to the replacement of `EU1/EU2` by `TU`, maintaining compatibility with legacy installations.

At 1.4 the support for obsolete `EU1` and `EU2` font encodings has been removed.

`\mst@OneifUniEnc`

The `\mst@OneifUniEnc` is expandable but must be used after having set `\mst@tmp@enc`...

```
\def\mst@oti{OT1}
\def\mst@tu{TU}
\def\mst@OneifUniEnc{\ifx\mst@tmp@enc\mst@tu1\else0\fi}
\newif\ifmst@goahead
\newif\ifmst@abort
```

`\mst@enc`
`\mst@fam`
`\mst@ser`
`\mst@opsh`
`\mst@bold`
`\mst@ltsh`

Macros to store the font settings, each math version will store its own records.

```
  \def\mst@enc{\encodingdefault}
  \def\mst@fam{\familydefault}
  \def\mst@ser{\seriesdefault}
```

`\mst@opsh` will hold default shape for operator names.
`\mst@ltsh` will hold default shape for letters.

```
  \def\mst@opsh{\shapedefault}
  \def\mst@bold{\bfdefault}
  \def\mst@ltsh{\shapedefault}
```

`\mst@greekfont`

1.15c: for use by the LGRgreek and selfGreek options. Defined as an `\edef` in order to be able to set-up once and for all the Greek at the time of `\usepackage`. Modifiable in the preamble via `\MTgreekfont{`⟨*font_name*⟩`}\Mathastext`.

```
  \edef\mst@greekfont{\familydefault}
```

`Package options`

2011/03/09: 1.13 introduces the option LGRgreek and systematic use of `\if...` conditionals, for better readability (by myself) of the code.

1.3y of 2022/11/03 adds `ncccomma`, `binarysemicolon` and `frenchmath*` options.

1.3za adds `LGRgreek+` and `LGRgreeks+`.

1.3zb adds `decimalcomma` and modifies `frenchmath*` to use it. And provides `frenchmath+` as an alias to former `frenchmath*`. Consecutive to the change at `v2.7` of frenchmath which replaced ncccomma by decimalcomma and broke the compatibility recipe explained in subsubsection 1.4.13.

1.4 adds options `everymath` and `activedigits`.

1.4b adds `noletters`.

```
\newif\ifmst@italic
\newif\ifmst@frenchmath
\newif\ifmst@ncccomma
\newif\ifmst@decimalcomma
\newif\ifmst@binarysemicolon
    \DeclareOption{italic}{\mst@italictrue
```

```
        \def\mst@ltsh{\itdefault}}
    \DeclareOption{frenchmath}{\mst@frenchmathtrue\mst@italictrue
      \def\mst@ltsh{\itdefault}}
    \DeclareOption{ncccomma}{\mst@ncccommatrue}
    \DeclareOption{decimalcomma}{\mst@decimalcommatrue}
    \DeclareOption{binarysemicolon}{\mst@binarysemicolontrue}
    \DeclareOption{frenchmath*}{\mst@frenchmathtrue\mst@italictrue
      \def\mst@ltsh{\itdefault}\mst@decimalcommatrue\mst@binarysemicolontrue}
    \DeclareOption{frenchmath+}{\mst@frenchmathtrue\mst@italictrue
      \def\mst@ltsh{\itdefault}\mst@ncccommatrue\mst@binarysemicolontrue}
\newif\ifmst@endash\mst@endashtrue
    \DeclareOption{endash}{\mst@endashtrue}
    \DeclareOption{noendash}{\mst@endashfalse}
\newif\ifmst@emdash
    \DeclareOption{emdash}{\mst@emdashtrue\mst@endashfalse}
\newif\ifmst@alldelims
\edef\mst@tmp{\encodingdefault}\ifx\mst@oti\mst@tmp\else\mst@alldelimstrue\fi
    \DeclareOption{alldelims}{\mst@alldelimstrue}
    \DeclareOption{nolessnomore}{\mst@alldelimsfalse}
\newif\ifmst@nosmalldelims
    \DeclareOption{nosmalldelims}{\mst@nosmalldelimstrue}
\newif\ifmst@noplus
    \DeclareOption{noplus}{\mst@noplustrue}
\newif\ifmst@nominus
    \DeclareOption{nominus}{\mst@nominustrue}
\DeclareOption{noplusnominus}{\ExecuteOptions{noplus,nominus}}
\newif\ifmst@noparen
    \DeclareOption{noparenthesis}{\mst@noparentrue}
\newif\ifmst@nopunct
    \DeclareOption{nopunctuation}{\mst@nopuncttrue}
\newif\ifmst@noequal
    \DeclareOption{noequal}{\mst@noequaltrue}
\newif\ifmst@noexclam
    \DeclareOption{noexclam}{\mst@noexclamtrue}
\newif\ifmst@asterisk
    \DeclareOption{asterisk}{\mst@asterisktrue}
\newif\ifmst@nospecials
    \DeclareOption{nospecials}{\mst@nospecialstrue}
\newif\ifmst@basic
    \DeclareOption{basic}{\mst@basictrue
    \ExecuteOptions{noparenthesis,nopunctuation,%
                    noplusnominus,noequal,noexclam,nospecials,nolessnomore}}
\newif\ifmst@nohbar
    \DeclareOption{nohbar}{\mst@nohbartrue}
```

1.4 adds option `activedigits`.

```
\newif\ifmst@activedigits
    \DeclareOption{activedigits}{\mst@activedigitstrue}
\newif\ifmst@nodigits
    \DeclareOption{nodigits}{\mst@nodigitstrue}
```

```
\newif\ifmst@defaultimath
    \DeclareOption{defaultimath}{\mst@defaultimathtrue}
```

1.4b adds option `noletters`.

```
\newif\ifmst@noletters
    \DeclareOption{noletters}{\mst@nolettersture\ExecuteOptions{nohbar,defaultimath}}
\newif\ifmst@mathaccents
    \DeclareOption{mathaccents}{\mst@mathaccentstrue}
\newif\ifmst@unimathaccents
    \DeclareOption{unimathaccents}{\mst@mathaccentstrue\mst@unimathaccentstrue}
\newif\ifmst@needsymbol
\newif\ifmst@symboldelimiters
    \DeclareOption{symboldelimiters}{\mst@needsymboltrue\mst@symboldelimiterstrue}
\newif\ifmst@symboldigits
    \DeclareOption{symboldigits}{\mst@needsymboltrue\mst@symboldigitstrue}
\newif\ifmst@symbolgreek
\newif\ifmst@customgreek
    \DeclareOption{symbolgreek}{\mst@needsymboltrue\mst@symbolgreektrue
                                \mst@customgreektrue }
\newif\ifmst@symbolre
    \DeclareOption{symbolre}{\mst@needsymboltrue\mst@symbolretrue}
\newif\ifmst@symbolmisc
    \DeclareOption{symbolmisc}{\mst@needsymboltrue\mst@symbolmisctrue}
    \DeclareOption{symbol}{\ExecuteOptions{symbolgreek,symbolmisc,symbolre}}
    \DeclareOption{symbolmax}{\ExecuteOptions{symbol,symboldelimiters}}
\newif\ifmst@needeuler
\newif\ifmst@eulerdigits
    \DeclareOption{eulerdigits}{\mst@needeulertrue\mst@eulerdigitstrue}
\newif\ifmst@eulergreek
    \DeclareOption{eulergreek}{\mst@needeulertrue\mst@eulergreektrue
                               \mst@customgreektrue }
\newif\ifmst@selfGreek
    \DeclareOption{selfGreek}{\mst@selfGreektrue\mst@customgreektrue}
\newif\ifmst@selfGreeks
    \DeclareOption{selfGreeks}{\mst@selfGreekstrue\mst@selfGreektrue
                               \mst@customgreektrue }
\newif\ifmst@LGRgreek
    \DeclareOption{LGRgreek}{\mst@LGRgreektrue\mst@customgreektrue}
\newif\ifmst@LGRgreeks
    \DeclareOption{LGRgreeks}{\mst@LGRgreekstrue\mst@LGRgreektrue
                              \mst@customgreektrue}
\newif\ifmst@greekplus
    \DeclareOption{LGRgreek+}{\ExecuteOptions{LGRgreek}\mst@greekplustrue}
    \DeclareOption{LGRgreeks+}{\ExecuteOptions{LGRgreeks}\mst@greekplustrue}
\def\mst@greek@select{0}
\newif\ifmst@itgreek
\newif\ifmst@upgreek
    \DeclareOption{itgreek}{\mst@itgreektrue}
    \DeclareOption{upgreek}{\mst@upgreektrue}
    \DeclareOption{itGreek}{\def\mst@greek@select{1}}
    \DeclareOption{upGreek}{\def\mst@greek@select{2}}
```

Starting with `1.15f` the meaning of the ‘defaultxx’ options has changed. They now prevent `mathastext` from defining additional alphabets rather than prevent it from identifying the ‘mathxx’ with the new ‘Mathxx’. The ‘Mathnormal’ and ‘Mathrm’ alphabet commands are always created as they are SymbolFontAlphabets.

This was again changed at `1.3za`. The additional alphabets are always declared, the options only prevent mapping the existing ‘mathxx’ to the new ‘Mathxx’. This may be breaking change if people used these options because they had a need for the `\Mathbf` etc... names.

```
\newif\ifmst@defaultnormal
    \DeclareOption{defaultnormal}{\mst@defaultnormaltrue}
\newif\ifmst@defaultrm
    \DeclareOption{defaultrm}{\mst@defaultrmtrue}
\newif\ifmst@defaultbf
    \DeclareOption{defaultbf}{\mst@defaultbftrue}
\newif\ifmst@defaultit
    \DeclareOption{defaultit}{\mst@defaultittrue}
\newif\ifmst@defaultsf
    \DeclareOption{defaultsf}{\mst@defaultsftrue}
\newif\ifmst@defaulttt
    \DeclareOption{defaulttt}{\mst@defaultttttrue}
```

Here and elsewhere `1.3za` has removed an `\ifmst@nonormalbold` conditional.

```
\DeclareOption{defaultalphabets}{\ExecuteOptions{defaultnormal,defaultrm,%
defaultbf,defaultit,defaultsf,defaulttt}}
```

`mathastext` considers the default script and especially scriptscript sizes to be far too small, and it will modify them. An option maintains the default.

```
\newif\ifmst@defaultsizes
    \DeclareOption{defaultmathsizes}{\mst@defaultsizestrue}
\newif\ifmst@twelve
    \DeclareOption{12pt}{\mst@twelvetrue}
\newif\ifmst@fouriervec
    \DeclareOption{fouriervec}{\mst@fouriervectrue}
```

`1.15`: the `subdued` option.

```
\newif\ifmst@subdued
    \DeclareOption{subdued}{\mst@subduedtrue}
```

`1.4`: the `everymath` option.

```
\newif\ifmst@everymath
    \DeclareOption{everymath}{\mst@everymathtrue
\PackageWarningNoLine{mathastext}{%
        `everymath\string' will be removed at next major release.\MessageBreak
        Please report to the author the circumstances\MessageBreak
        which mandated its use in the present document}%
}
```

`1.3q`: the `unicode` option. Thanks to Tobias Brink for suggesting its incorporation. The parsing of `\CurrentOption` does not seek any robustness, it just does its job if the option is used correctly.

```
\def\mst@unicodeminus {2013}
```

```
\def\mst@checkoption #1unicodeminus#2\mst@#3\mst@@
   {\ifx\\#3\\\PackageWarningNoLine{mathastext}
               {Unknown option `\CurrentOption\string'}\else
    \ifx\\#2\\\def\mst@unicodeminus {2212}\else
    \expandafter\def\expandafter\mst@unicodeminus\expandafter{\@secondoftwo#2}%
    \fi\fi}
\DeclareOption*%
  {\expandafter\mst@checkoption\CurrentOption\mst@ unicodeminus\mst@\mst@@}
```

```
\ProcessOptions\relax
```

`mst@DeclareMathAccent`

I somehow missed realizing LaTeX 2019-10-01 if used together with `amsmath` made repeated usage of `\DeclareMathAccent` trigger an error: `https://github.com/latex3/latex2e/issues/216`.

This broke usage of `\Mathastext` macro in preamble.

`1.3w` works around this via `\mst@DeclareMathAccent`. And other changes were made in `mathastext` code to cope with these complications around robustness.

```
\def\mst@DeclareMathAccent#1{\let#1\mst@undefined
  \expandafter
  \let\csname\expandafter\@gobble\string#1\space\endcsname\mst@undefined
  \DeclareMathAccent{#1}}
```

`mst@normalversionname`
`\mst@boldversionname`

Helper macros to test math version names. User is not allowed to redefine via `\Mathastext` with optional argument or via `\MTDeclareVersion` the `normal` and `bold` math versions. Added at `1.3w`, about 7 years late.

```
\def\mst@normalversionname{normal}%
\def\mst@boldversionname{bold}%
```

`\mst@OnlyIfNotSubdued`

`1.3u` adds this check that we are not in a subdued normal or bold math version. We do not push the `#1` out of the TeX conditionals, and anyhow there was no real need for expandable coding.

```
\def\mst@OnlyIfNotSubdued#1{%
    \ifmst@subdued
      \ifx\math@version\mst@normalversionname
      \else
        \ifx\math@version\mst@boldversionname
        \else
          #1%
        \fi
      \fi
    \else
      #1%
    \fi
}%
```

`\exists`
`\mst@exists@skip`
`\forall`
`\mst@forall@skip`
`\MTnormalexists`
`\MTexistsdoesskip`
`\MTnormalforall`
`\MTforalldoesskip`

`1.15e` 2012/10/21: math skip/glue *after* `\exists` and `\forall`, this is useful with upright letters in math mode. Each math version has its own user defined values for the skips, stored as macros. The redefinitions of $\exists$ and $\forall$ are done only at the end of the package as the `symbol` option will also want to redefine these math symbols.

The subdued option (later and only for the normal and bold math version) and the italic option (here) set to zero the package default skips. With `1.2` the skips can be modified on the fly in the document, they are not necessarily set in the preamble once and for all for each math version.

1.3j adds `\MTnormalexists`, `\MTexistsdoesskip`, `\MTnormalforall`, `\MTforalldoesskip`.

Earlier to 1.3j, `\let\mst@exists@original\exists` was done at End of Package, now it is done at Begin Document, and same for `\forall`. We pay attention that use of `\MTnormalexists` etc... inside the preamble does not create self-let's.

Also `subdued` mode will do `\MTnormalexists`, `\MTnormalforall` (earlier than 1.3j, it only set the muskips to 0mu.) Same when using `\MTversion{normal}`, if `subdued`.

For some (random, legacy) reason, the handling of ∃ and ∀ is part of the things not included inside `\everymath`/`\everydisplay`.

1.3v The **mathastext**-defined `\exists` and `\forall` are created `\protected`. We feel this matches better with their default definition as `\mathchardef` tokens than dealing with LaTeX2e robust macros. Besides, the coding is simpler.

```
\newmuskip\mst@exists@muskip
\newmuskip\mst@forall@muskip
\def\mst@exists@skip{1mu}
\def\mst@forall@skip{.6667mu}
\ifmst@italic\ifmst@frenchmath\else
    \def\mst@exists@skip{0mu}
    \def\mst@forall@skip{0mu}
    \def\mst@prime@skip {0mu}
\fi\fi
\protected\def\mst@exists{\mst@exists@original\mskip\mst@exists@muskip}
\protected\def\mst@forall{\mst@forall@original\mskip\mst@forall@muskip}
\AtBeginDocument{%
   \let\mst@exists@original\exists
   \let\mst@forall@original\forall
   \def\MTnormalexists   {\let\exists\mst@exists@original }%
   \def\MTexistsdoesskip {\let\exists\mst@exists }%
   \def\MTnormalforall   {\let\forall\mst@forall@original }%
   \def\MTforalldoesskip {\let\forall\mst@forall }%
```

The document body starts in the normal math version, whether or not `\Mathastext` command as been used in the preamble (which either re-defines the normal/bold math version or defines another one in case of optional argument), and in case of `subdued` option should use the standard ∀ and ∃.

```
   \ifmst@subdued
   \else
     \MTexistsdoesskip
     \MTforalldoesskip
   \fi
}%
\newcommand*\MTnormalexists   {\AtBeginDocument {\MTnormalexists   }}
\newcommand*\MTexistsdoesskip {\AtBeginDocument {\MTexistsdoesskip }}
\newcommand*\MTnormalforall   {\AtBeginDocument {\MTnormalforall   }}
\newcommand*\MTforalldoesskip {\AtBeginDocument {\MTforalldoesskip }}
```

`\prime`
`\mst@prime@skip`
`\active@math@prime`
`\MTnormalprime`
`\MTprimedoesskip`

1.2 2012/12/17: math skip/glue *before* the `\prime` glyph. This is useful with the default CM glyph and upright letters (in contrast the prime from `txfonts` works fine with upright letters).

For this we replace the LaTeX kernel `\active@math@prime` with our own skip-enhanced version `\mst@active@math@prime`.

1.2b 2012/12/31: doing

```
{\catcode`\'=\active \global\let'\mst@active@math@prime}
```

is awfully wrong when the right quote is made active at begin document by some other package (as happens with `babel` for some languages). So **`mathastext`** treats now the right quote with the same method as applied to the other characters it makes mathematically active. This uses the macro `\mst@mathactivate` which is defined later in the package.

Babel does `\let\prim@s\bbl@prim@s` when `'` is made active via its services (the czech and slovak languages also store the initial version of `\prim@s`, else the quote would not work correctly when being again of `catcode 12`), and it doesn't matter if **`mathastext`** is loaded before or after this happens, as the `\mst@mathactivate` does its job only as part of the `\everymath` and `\everydisplay` token lists.

1.2e being paranoid, we take precautions against a possibly catcode active right quote at the time of loading mathastext.

1.3i adds `\MTactiveprime`.

1.3j renames it to `\MTprimedoesskip`. Besides, it makes use in the preamble of `\MTnormalprime` or `\MTprimedoesskip`.

1.4 adds the support for the new “non-everymath” implementation, which has to satisfy to very different constraints.

The `\MTprimedoesskip` is not a `no-op` in `subdued` math version. This is legacy situation, not changed at 1.4.

1.4d fixes a (very) bad bug introduced at 1.4 which caused `$f''$` to error. The fix is to be found in the definition of `\mst@@mathactivate` (done later in the package) used for the non-`everymath` branch.

```
\newmuskip\mst@prime@muskip
\def\mst@prime@skip{.5mu}
\ifmst@italic\ifmst@frenchmath\else\def\mst@prime@skip{0mu}\fi\fi
```

Shouldn't I have rather hacked `\prim@s`? (answer: perhaps related to Babel see a comment above).

TODO: clarify why I used `\sp` not ^ in `\mst@active@math@prime` in 2012/2013.

TODO: clarify why I am paranoid regarding the `'` catcode here.

The `everymath` branch was initially superficially refactored at 1.4, but unfortunately as `\mst@mathactivate` had been modified to not handle mathematically active characters, this meant that the **`mathastext`** feature of extra math skip was lost under the `everymath` option, due to an oversight by the author. For this among other reasons having mainly to do with code comments we did 1.4a.

At 1.4 we can not go via `\mst@mathactivate` (defined further down in the code) which filters out the already mathematically active characters.

Why am I paranoid here about `'` catcode at package loading time? (this looks ridiculous...).

```
\def\mst@tmp#1{%
\def\mst@mathactivateprime{%
  \ifnum\catcode`#1=\active
    \@ifundefined{active@char#1}
                 {}
                 {\mst@do@activecase #1{}{\mst@active@math@prime}}%
  \else
    \mst@@mathactivate #1{}{\mst@active@math@prime}%
  \fi
}%
```

The “undo” is needed at 1.4 (but not if `everymath`). The 1.4c (with Unicode engines) uses `\Umathcodenum` in `\mst@mathdeactivate` and not `\mathcode` hence no `"8000` here.

```
\def\mst@undo@mathactivateprime{\mst@mathdeactivate#1{\mst@activemathcodenum@space}}%
}\expandafter\mst@tmp\string'
\def\mst@active@math@prime{\sp\bgroup\mskip\mst@prime@muskip\prim@s}
\ifmst@everymath
    \newcommand*\MTnormalprime  {\let\mst@modifyprime\@empty}
    \newcommand*\MTprimedoesskip{\let\mst@modifyprime\mst@mathactivateprime}%
    \AtBeginDocument{%
      \everymath\expandafter
               {\the\everymath    \mst@modifyprime \MTnormalprime}%
      \everydisplay\expandafter
               {\the\everydisplay \mst@modifyprime \MTnormalprime}%
```

MEMO: `subdued` case will do its own `\MTnormalprime` at `\begin{document}` later as part of `\MTeverymathoff`.

```
      \MTprimedoesskip
    }
\else
```

1.4 must do things a bit differently.

```
    \newcommand*\MTnormalprime  {\mst@undo@mathactivateprime}
    \newcommand*\MTprimedoesskip{\mst@mathactivateprime}
```

MEMO: `subdued` case will do its own `\MTnormalprime` at `\begin{document}` later as part of `\MTeverymathoff`.

```
    \AtBeginDocument{\MTprimedoesskip}
\fi
```

`\MTexistsskip`
`\MTforallskip`
`\MTprimeskip`

1.15e: These user macros set up the amount of muglue after `\exists` or `\forall`. The normal and bold math versions inherit the same skips; these skips are set to zero in case of the subdued, or the italic option. Each command `\Mathastext[`⟨*version_name*⟩`]` stores the current values in the definition of the math version.

1.2: `\MTprimeskip` added, the silly `\@onlypreamble` are removed and the macros are modified to have immediate effect in the document, independently of their possible use in the preamble for the math versions to store values.

Note (september 2013): the names were badly chosen; `\MTsetprimeskipto` for example would have been a better choice.

```
\newcommand*\MTexistsskip[1]{\edef\mst@exists@skip{#1}%
    \mst@exists@muskip\mst@exists@skip\relax}
\newcommand*\MTforallskip[1]{\edef\mst@forall@skip{#1}%
    \mst@forall@muskip\mst@forall@skip\relax}
\newcommand*\MTprimeskip[1]{\edef\mst@prime@skip{#1}%
    \mst@prime@muskip\mst@prime@skip\relax}
\let\Mathastextexistsskip\MTexistsskip
\let\Mathastextforallskip\MTforallskip
\let\Mathastextprimeskip\MTprimeskip
\let\mathastextexistsskip\MTexistsskip
```

```
\let\mathastextforallskip\MTforallskip
\let\mathastextprimeskip\MTprimeskip
```

`\resetMathstrut@`

2012/12/31: The `amsmath` macro `\resetMathstrut@` is not compatible with a mathematically active opening parenthesis: it does

```
\mathchardef\@tempa\mathcode`\(\relax
```

and is made a part of the hook `\every@math@size` inside `\glb@settings`. This is called from `\check@mathfonts` which is done in particular in `\frozen@everymath`, hence *before* (but wait) what **mathastext** puts in `\everymath`. Also, `\glb@settings` is triggered by `\mathversion` which must be done outside of math mode.

Alas, with things such as `$...\hbox{...$..$..}...$` **mathastext** will have already made the parenthesis (mathematically) active. And `\boldsymbol` from `amsbsy` disables the `\@nomath` switch and executes `\mathversion{bold}` directly in math mode. So we have a problem with `\resetMathstrut@`.

`lualatex-math` replaces `\resetMathstrut@` with its own version (which also looks at `)`) and no error is signaled when **mathastext** has done ``\mathcode`(="8000``, but the `\Mathstrutbox@` created by **mathastext** is then wrong.

The replacement macro avoids a potentially math active `(`. It assumes that there is still some appropriate glyph in slot 40 of `operators` and it sets the height and depth of `\Mathstrutbox@` to be large enough to accomodate both this glyph and the one from the mathastext font (both in the current math version). If option `noparenthesis` was used, we leave everything untouched.

In `1.3a`, 2013/09/04, the modification is done only at the time of `\MTnonlettersobeymathxx`. It is canceled by `\MTnonlettersdonotobeymathxx`. So the code has been moved to these macros and here we just store at the begin document the then meaning of `\resetMathstrut@`, and check also if `\MTnonlettersobeymathxx` has been invoked in the preamble.

`1.3f` 2015/09/12 issues only an Info message not a Warning, as I am becoming aware from another context (etoc) that Warnings are stressful to users, in some integrated environments for editing and compiling LaTeX source files.

`1.4` adds here an `\AtEndOfPackage`. As this must be executed (if not in the legacy `everymath` context) after the redefinition of `\MTnonlettersobeymathxx` which happens at begin document.

```
\ifmst@noparen\else
\AtEndOfPackage{%
  \AtBeginDocument{%
    \@ifundefined{resetMathstrut@}
      {}%
      {%
       \let\mst@savedresetMathstrut@\resetMathstrut@
```

The `\ifx\mst@the\the` is true iff `\MTnonlettersobeymathxx` was used in preamble (and after possibly some `\MTnonlettersdonotobeymathxx`). Setting `\mst@the` to `\@gobble` here wiil cause `\MTnonlettersobeymathxx` to indeed modify `\resetMathstrut@` (and it resets `\mst@the` to `\the` for its behavior in `\everymath`).

```
       \ifx\mst@the\the
         \let\mst@the\@gobble
         \MTnonlettersobeymathxx
       \fi
      }%
  }%
}%
\fi
```

1.2 2012/12/20 does some rather daring *math* activation of `; , : ! ? + - = < > ( ) [ ]` in math mode to achieve something I wanted to do since a long time: overcome the mutually excluding relation between the variable-family concept and the automatic spacing concept. After loading **`mathastext`**, these characters now obey the math alphabets commands but still have the automatic spacing. The use as delimiters for those concerned is also ok.

The activation is done via setting the `\mathcode` to `"8000` through the macro `\mst@mathactivate` which in turn is put into the `\everymath` and `\everydisplay` token lists. No character is made active in the sense of the `\catcode` (the issues with catcode active characters at the entrance of the math mode are discussed later),

but the concerned characters will now expand in math mode to *two* tokens.

1.2c 2012/12/31: hence, this current implementation puts constraints on the input: `$x^?$` or `$x\mathrel?y$` now create errors. They must be input `$x^{?}$`, respectively `$x\mathrel{?}y$`.

The disactivating macro `\MTnonlettersdonotobeymathxx` is made the default.

The mechanism is (even more) off by default for `\{` and `\}` as this is not compatible with their use as delimiters (`\lbrace` and `\rbrace` should be used instead) but it can be activated for them too.

1.2b 2012/12/30: there were bad oversights in the 1.2 code for `\mst@mathactivate` related to the possibility for some characters to have been made active (in the sense of the catcode) elsewhere (something which often is done by language definition files of the `babel` system). The code from `v1.2b` tried to provide correct behavior using a prefix called `\mst@fork` (its definition and its use has since been modified) which let the active character expand to the **`mathastext`** re-definition *only* in math mode and *only* if `\protect` was `\@typeset@protect`. This indeed took care of situations such as `$\hbox{?}$` with an active `?` or `$\label{eq:1}$` with an active `:` (assuming for the latter that things would have worked ok before the twiddling by **`mathastext`**).

1.2e 2013/01/09: alas `$\ref{eq:1}$` still was a problem. Indeed in that case the **`mathastext`** prefix had no means to know it was inside a `\ref` so it made the character expand to its **`mathastext`** redefinition, which is not acceptable inside a `\csname...\endcsname`. What happens with Babel is that it patches things such as `\ref`, `\newlabel`,... we can test the `\if@safe@actives` flag to detect it in that case, but this is Babel specific. After having thought hard about this I see no general solution except patching all macros such as `\ref`...(in an imitation of what Babel does). So the final decision is to not do anything when the character is catcode active *except* it it seems that Babel is behind the scenes.

Incidently, Babel and TikZ are buggy with characters which are mathcode actives. For example the combination of `[french]{babel}` and `mathtools` with its `centercolon` turns `$:$` into an *infinite loop* !!

In the case of Babel the reason is that, generally (but not always, the right quote `'` is an exception), the `\normal@char`⟨*char*⟩ fall-back is `\string`⟨*char*⟩. But this is wrong if the mathcode is 32768! The fall-back becomes the default if the user switches to a language where ⟨*char*⟩ is ‘normal’ and then an infinite loop arises.

As a further example (I am not familiar with other languages from the Babel system) with `frenchb` the active `!?;:` expand in math mode to `\string! or ? or ; or :`. This creates an infinite loop if the mathcode is 32768.

For the special case of the right quote `'` when it is made active by Babel, its fall-back does not invoke `\string'` so being still of mathcode 32768 is not a problem.

I have posted online how Babel should possibly modify its definitions and I use this here. I simplify a bit my proposed replacement of `\normal@char`⟨*char*⟩ as the check for `\protect` is superfluous, I think, having been done already at the level of the Babel prefix.

Replacing `\user@active`⟨*char*⟩ is indeed not enough, and `\normal@char`⟨*char*⟩ also must be changed, because when the user switches back to a language where the character is 'normal' it remains catcode active. The crucial thing is the test of `\if@safe@actives` in the replacement of the `\normal@char`⟨*char*⟩, besides of course the test for math mode in both replacements.

When the character is not catcode active, then **mathastext** uses the math activation method. As the mathcode is not looked at in `\edef`, `\write` or inside `\csname...\endcsname` nothing special needs to be done, I think, in terms of protection against premature expansion. (I did not know that initially).

So, to recapitulate, **mathastext** will use the mechanism of the active `mathcode` if the character is not `catcode` active, and in the opposite case will do something only in the context of Babel, modifying directly its `\user@active`⟨*char*⟩ and its `\normal@char`⟨*char*⟩ macros and it does NOT then set the mathcode to 32768!!, rather it makes *sure* the character is not mathematically active.

As `1.2e` is a bit paranoid it takes precautions against the possibility of characters it treats being active at the time of its loading. Excepted from the scope of the paranoia are the latin letters (that would be crazy!) and also `*`, `"` and the left quote `` ` ``.

`1.2f` `2013/01/21` with earlier versions (*) it was important not to do twice the business of `\mst@mathactivate` (think `$\hbox{$?$}$`), so I used (this was a bit wasteful) some sort of boolean macro for each character. But now that there are the `\mst@the..` prefixes, let's just use them! (don't know why I did not think of that earlier; perhaps I had in mind some more general character per character customization initially, which I just dropped.)

(*) it is still important to not do twice the thing when the character is active, in which case the `babel` macros are patched.

As an aside, ``$\hbox{\catcode`?=\active $?$}$`` for an `?` which was unactive at the first `$` will just make **mathastext** overwrite the definition (assumed here to have been done earlier) of an active `?`, but the result is that the inner `?` can not be used in `\label` or `\ref`. So testing for active characters should be done always... many things should be done always... I leave as is.

`1.3i` `2016/01/06` removes a spurious end of line space in `\mst@mathactivate` (did not show as anyhow done in math mode).

`1.4` has significantly refactored the coding, there were simplication from a changed way to use `\mst@mathactivate` for letters and added branches to accomodate the renouncement to `\everymath`.

Some auxiliaries. The last two (which use the first two) are injected inside the redefinitions of active Babel shorthands done by `\mst@do@activecase`.

```
\def\mst@magic@v #1#2#3#4#5{#1#3#4}
\def\mst@magic@vi #1#2#3#4#5#6{#1#2#4#5}
\def\mst@fork{\ifmmode\mst@magic@v\fi\@thirdofthree}
\def\mst@safefork{\ifmmode\if@safe@actives\else\mst@magic@vi\fi\fi\@thirdofthree}
```

`\mst@do@activecase`

Called by `\mst@mathactivate` if `#1` is catcode active. At `1.4` it is also required that the token activation originated frombabel. Else this `\mst@do@activecase` is not executed. If executed, it hooks into babel associated macros so that in math mode the catcode active token does what **mathastext** expects it to do. Its `\mathcode` is set to the one of the associated `\mathalpha` symbol as declared by **mathastext**, except for asterisk and right tick which simply use their ascii number as mathcode.

In the case of babel-spanish which has a catcode active right tick, resetting its mathcode has the advantage to reveal in output the case of a faulty input using the curly right tick U+8217 which one can get easily from copy paste, for example from the `babel-spanish.pdf` file.

And in general it is dangerous to have a character both catcode active and mathcode active. Some refactoring (and streamlining) at `1.4` here. There was at some point in the case of a Babel-active token a special hanlding for `#1` to use `\MTmathcharletter#1` (and also for digits!). But this is almost impossible to arise in practice:

- Babel commands to set up a shorthand are preamble only (they use `\AtBeginDocument`),
- A shorthand is made active at begin document, and we don't want that for a letter!
- Activation is recorded as an instruction in the `.aux` file which breaks it almost surely (except perhaps for letter `Z` or such not likely to appear in macro names in the `.aux` file).
- Even if we somehow hack against that (which I did to test) a more subtle problem arises that Babel updates `\nfss@catcodes`, so that the latter will want to make (e.g.) `a` of catcode 12! This causes breakage very easily at the time of `\process@table` almost as easily as with the `a` set to be an active character...

`#1` here is a character token of catcode 11 or 12. We do something only if the catcode activation appears to have been triggered by babel. The test has been moved earlier at `1.4`.

```
\def\mst@do@activecase#1#2#3{%
```

As letters and digits are impossible here, `#2` is either empty (which happens only now for asterisk and right tick) or a single token (`\mathclose`, `\mathopen`,...). When `#2` is not empty, `#3` is always (not quite see below) a mathchar token. For safety we make sure then the mathcode is not active.

The rationale is to avoid context where people have both catcode and math active and use `\string`. In the case of babel-spanish with `activeacute` option it allows to reveal more easily input errors using curly right tick U+8217.

Arguably I probably mainly wanted to do this if the character was found catcode active but not a Babel shorthand. But it ended into the Babel branch and the non-Babel branch for active catcode is to do nothing. As it is now to time-intensive to re-check the whole logic I am keeping this.

`#2` is empty exactly for the case of the right tick and the asterisk. In all other cases, then `#3` is a mathchar token, except in the exceptional case of `#1` being `-` (necessarily of catcode 12 here), `#3` with Unicode engines is `\mst@varfam@minus` which is a macro expanding to a math-version dependent `\Umathchardef` token. Then the `\mathcode` assignment breaks under XeLaTeX. It appears unlikely for `-` to be a Babel shorthand. But to be extra safe, and at same time avoid undue complications of needed separate branches for non-Unicode and Unicode engines, we simply drop the assignment in that case (done at `1.4c`).

The matter with the `\string-` is inherited stupid paranoia from early times regarding catcodes at package loading, as I do it elsewhere in the package code, also here.

```
    \expandafter\if\string-#1\else
      \ifx\relax #2\relax\mathcode`#1=`#1\relax\else\mathcode`#1=#3\relax\fi
    \fi
```

Less `\expandafter`'s at `1.4`. Also, it is now required to make sure:

- not do it twice in succession (else infinite loop),
- and undo it when deactivating.

This was easily done under legacy code (now only used under `everymath` option, and only for non-letters) as it activated only when entering into math mode.

This is more annoying when activation happens in an uncontrolled scope, and we handle this via a `\mst@hackedshorthand@<char>` flag.

MEMO: the undoing is done in `\mst@mathdeactivate` which is invoked only for those non-letters submitted to `\mst@mathactivate`.

```
    \ifmst@everymath\else\@ifundefined{mst@hackedshorthand@#1}{\fi
    \expandafter\let\csname mst@orig@user@active#1\expandafter\endcsname
                    \csname user@active#1\endcsname
    \expandafter\let\csname mst@orig@normal@char#1\expandafter\endcsname
```

```
                    \csname normal@char#1\endcsname
    \ifmst@everymath\else}{}\fi
```

No more `\edef`'s at 1.4. At this release, in letter case `#3` is empty and `#2` is a single token which needs no extra brace pair as this brace pair will come from its expansion. We can use `{#2}{#3}` in all cases rather than either `{{#2#3}}{}` versus `{#2}#3` depending on whether `#1` is letter or not as was done in the past. Attention that `#2` and `#3` may each be empty.

```
    \def\mst@tmp##1##2{\def##1{\mst@fork{#2}{#3}##2}}%
    \expandafter\mst@tmp\csname user@active#1\expandafter\endcsname
                        \csname mst@orig@user@active#1\endcsname
    \def\mst@tmp##1##2{\def##1{\mst@safefork{#2}{#3}##2}}%
    \expandafter\mst@tmp\csname normal@char#1\expandafter\endcsname
                        \csname mst@orig@normal@char#1\endcsname
    \ifmst@everymath
    \else
      \expandafter\let\csname mst@hackedshorthand@#1\endcsname\@empty
    \fi
}
```

`\mst@@mathactivate`

```
\begingroup
  \catcode`\~=\active
  \def\x{%
\endgroup
```

`##1` is always a token of catcode 11 or 12.

```
\ifmst@everymath
   \def\mst@@mathactivate##1##2##3{%
      \begingroup
      \lccode`~=`##1
      \lccode`##1=`##1
```

Careful here as `##2` is empty in the asterisk and prime case. And `##3` also is at 1.4 empty for letters.

```
      \lowercase{\endgroup
```

Refactoring at 1.4 merged the catcode 11 and catcode 12 branches here and in particular avoided in former case an `\edef`. This helped into providing the `\MTcommandletter`⟨*letter*⟩ as a customizable macro. There was no equivalent to this macro, which here is `##2`, prior to 1.4.

```
        \mathcode`##1="8000
        \def~{##2##3}%
      }%
    }%
\else
   \def\mst@@mathactivate##1##2##3{%
      \begingroup
      \lccode`~=`##1
      \lccode`##1=`##1
      \lowercase{\endgroup
        \mathcode`##1="8000
```

Also, 1.4 possibly execute `\mst@mathactivate` everywhere in document body (at version changes for example) or at begin document. Hence this modifies globally the active meaning. So we add some safeguard here using an `\ifmmode`. And we store the original meaning of the active variant of the token to reset it when “undoing”.

Careful not to do it twice in a row... (but this means `\mst@mathdeactivate` might not revert to a user custom redefinition done in-between but to a prior one; although if the character is made active the second `\mst@mathactivate` would not have been done).

```
        \@ifundefined{mst@prioractivemeaning@##1}
                     {\expandafter\let
                      \csname mst@prioractivemeaning@##1\endcsname ~}
                     {}%
```

The 1.4 added this `\ifmmode` test but overlooked the case of the right tick `'` which requires to be moved to after the conditional. Let’s do this in a safer way.

```
        \def~{\ifmmode\expandafter\@firstoftwo\else\expandafter\@secondoftwo\fi
              {##2##3}{##1}}%
      }%
    }%
\fi
}\x
```

At 1.4 the macro was split into two, to test first whether the character which is encountered is currently with active catcode.

At 1.4 the non-catcode active branch is entered only if the character isn’t mathematically active at time of use.

Careful that `#2` and `#3` may each be empty. The test for being a Babel shorthand is done here and not as prior to **mathastext** @release1.4 inside `\mst@do@activecase`.

```
\def\mst@mathactivate#1#2#3{%
    \ifnum\catcode`#1=\active
      \@ifundefined{active@char#1}{}{\mst@do@activecase #1{#2}{#3}}%
    \else
      \mst@OnlyIfNotMathActive{#1}{\mst@@mathactivate #1{#2}{#3}}%
    \fi
}
```

`\mst@mathdeactivate`

This is needed at 1.4 for non-letters which are mathematically activated. See comments above in `\mst@do@activecase`. Works in sync with `\mst@addtodo@nonletters`.

The `\mst@mathdeactivate` is active (sic) only in the non `everymath` situation. It will restore a previously existing active meaning if it has been changed.

The 1.4c uses `\Umathcodenum` here with Unicode engines (the actual conditions why this became necessary were related to the minus character which however does not use `\mst@mathdeactivate` anyway at time of finishing the 1.4c updates). The `\string` thing with `-` is due to some old paranoia for which I see no underlying reason (why on earth would `-` be catcode active here?), but maintaining it. For `#1=-` the `#2` is empty and unused as the mathcode reassignment is then skipped (because `\mst@subduedminus` and `\mst@nonsubduedminus` take care of such things in an engine dependent manner; see them).

```
\ifmst@everymath
\else
\def\mst@mathdeactivate#1#2{%
  \if#1\string-\else\mst@mathcodenum`#1=#2\relax\fi
  \@ifundefined{active@char#1}
      {\@ifundefined{mst@prioractivemeaning@#1}
             {}%
             {\mst@restoreactivemeaning#1%
              \expandafter\let\csname mst@prioractivemeaning@#1\endcsname\relax}%
      }
      {\@ifundefined{mst@hackedshorthand@#1}
       {}
       {\expandafter\let\csname user@active#1\expandafter\endcsname
                        \csname mst@orig@user@active#1\endcsname
        \expandafter\let\csname normal@char#1\expandafter\endcsname
                       \csname mst@orig@normal@char#1\endcsname
        \expandafter\let\csname mst@hackedshorthand@#1\endcsname\relax
       }%
      }%
}%
```

`@restoreactivemeaning`

At 1.4 when we undo the mathematical activation we now also restore the prior existing active meaning, if any. Only in the “no `everymath`” branch (because in the `everymath` TeX itself takes care of that on exiting the scope of the math mode).

```
\def\mst@restoreactivemeaning#1{%
    \begingroup
    \lccode`~=`#1
    \lccode`#1=`#1
    \lowercase{\endgroup
      \expandafter\let\expandafter~\csname mst@prioractivemeaning@#1\endcsname
    }%
}
\fi
```

`\mst@do@nonletters`
`st@addtodo@nonletters`
`\mst@the`
`Tnonlettersobeymathxx`
`ettersdonotobeymathxx`

These macros are modified in version 1.3a `2013/09/04` in order to cleverly adjust, or not, the `amsmath` `\resetMathstrut@`. When used in the preamble, they just modify `\mst@the`. And there is code at begin document to check the status there of `\mst@the` and if its meaning is `\the`, then `\MTnonlettersobeymathxx` is activated again to do the patch. When used in the body they adjust `\resetMathstrut@`.

Notice that the saved meaning is the one at begin document (thus, possibly patched by `lualatex-math` — not anymore since 1.5 of March 2016, as `amsmath.sty` now maintained by LaTeX team has modified `\resetMathStrut@` to make it compatible to Unicode engines) but modifications done after that would not be seen in `\mst@savedresetMathstrut@`.

The new version of `\resetMathStrut@` from LaTeX team release `2016/03/03 v2.15a` of `amsmath.sty` is still not compatible with a math active opening parenthesis. Hence my patch here is still needed.

At 1.3u `\MTnonlettersobeymathxx` and `\MTeasynonlettersobeymathxx` are made no-ops under subdued mode. This fixes some bug if for example the former was used in preamble or immediately after `\begin{document}` making the minus sign math active although the **mathastext** action was supposedly subdued. Similarly `\MTmathactiveletters` is now a no-op if issued under subdued mode in the *normal* or *bold* math versions.

```
\newtoks\mst@do@nonletters
\ifmst@everymath
```

```
\else
    \newtoks\mst@undo@nonletters
\fi
```

`#1` is a category 12 character, `#2` is a `\mathopen`, or `\mathclose`, etc..., `#3` is a `\mathchar` (of variable family type).

At `1.4`, `\mst@mathactivate` will not do anything if `#1` is mathcode active (but not catcode active) at time of use.

```
\ifmst@everymath
    \def\mst@addtodo@nonletters#1#2#3{\mst@do@nonletters\expandafter
                   {\the\mst@do@nonletters\mst@mathactivate#1#2#3}%
    }%
\else
```

We need to automatize some safeguards related to `\mst@do@activecase` when deactivating.

`1.4c` fixes two problems of `1.4` code. First a bug due to XeLaTeX diverging implementation of `\mathcode` compared to LuaLaTeX(or rather vice versa), second the fact that the mathcode here is the one at package loading time, which for most non-letter characters was fine because it had been set by **mathastext**, but for the minus character it was too early.

The generic `\mst@addtodo@nonletters` can not be used with `-`, some specific code is used, see `\mst@nonsubduedminus`.

```
    \def\mst@addtodo@nonletters#1#2#3{%
        \mst@do@nonletters\expandafter
                   {\the\mst@do@nonletters\mst@mathactivate#1#2#3}%
        \edef\mst@tmp{\noexpand\mst@mathdeactivate#1{\the\mst@mathcodenum`#1}}%
        \mst@undo@nonletters\expandafter\expandafter\expandafter
                  {\expandafter\mst@tmp\the\mst@undo@nonletters}%
    }%
\fi
\let\mst@the\@gobble
```

As `\mst@savedresetMathstrut@` will only be defined at begin document, the next two macros are no-op in the preamble.

```
\def\mst@redefine@resetMathstrut@{%
      \@ifundefined{mst@savedresetMathstrut@}
      {}
      {%
        \ifmst@symboldelimiters
          \def\resetMathstrut@{%
           \setbox\z@\hbox{\the\textfont\symmtpsymbol\char40
                        \the\textfont\symmtoperatorfont\char40
                        \the\textfont\symoperators\char40}%
           \ht\Mathstrutbox@\ht\z@ \dp\Mathstrutbox@\dp\z@}%
        \else
          \def\resetMathstrut@{%
           \setbox\z@\hbox{\the\textfont\symmtoperatorfont\char40
                        \the\textfont\symoperators\char40}%
           \ht\Mathstrutbox@\ht\z@ \dp\Mathstrutbox@\dp\z@}%
        \fi
        \PackageInfo{mathastext}{\string\resetMathstrut@\space
        from amsmath replaced for this\MessageBreak group or environment}%
      }%
}%
\def\mst@restore@resetMathstrut@{%
      \@ifundefined{mst@savedresetMathstrut@}{}{%
        \PackageInfo{mathastext}{restoring for this group or environment
                                 the original\MessageBreak
                                 amsmath \protect\resetMathstrut@}%
      \let\resetMathstrut@\mst@savedresetMathstrut@}%
}%
\ifmst@everymath
  \newcommand*\MTnonlettersobeymathxx{%
    \mst@OnlyIfNotSubdued{%
        \ifx\mst@the\the
        \else
           \mst@redefine@resetMathstrut@
        \fi
        \let\mst@the\the
    }%
  }%
  \newcommand*\MTnonlettersdonotobeymathxx{%
    \ifx\mst@the\@gobble
    \else
      \mst@restore@resetMathstrut@
    \fi
    \let\mst@the\@gobble
  }%
\else
  \newcommand*\MTnonlettersobeymathxx{%
     \mst@OnlyIfNotSubdued{%
        \AtBeginDocument{\MTnonlettersobeymathxx}%
     }%
  }%
  \newcommand*\MTnonlettersdonotobeymathxx{%
     \AtBeginDocument{\MTnonlettersdonotobeymathxx}%
  }%
  \AtBeginDocument{%
    \renewcommand*\MTnonlettersobeymathxx{%
        \mst@OnlyIfNotSubdued{%
             \the\mst@do@nonletters
             \ifx\mst@the\the
             \else
                \mst@redefine@resetMathstrut@
             \fi
             \let\mst@the\the
         }%
    }%
```

The next one gets executed (even if not used by user) via `\MTeverymathoff`, which itself is done automatically at begin document if in `subdued` mode. The mechanism of `subdued` mode is to reassign the `mtoperatorfont` and `mtletterfont` to the fonts used by default in the normal math version. The reassignment of mathcodes is needed in case a priori `\MTnonlettersobeymathxx` made some characters math active. Normally the slots as defined by **`mathastext`** match default slots. The case of minus character is special as it by default with legacy TEX fonts and encodings will try to be the EN DASH, whose slot depends on the encoding. The used mathcode's for the minus sign are thus stored by **`mathastext`**, but under the shape of `\(U)mathchar` control sequences.

```
    \renewcommand*\MTnonlettersdonotobeymathxx{%
        \the\mst@undo@nonletters
        \ifx\mst@the\@gobble
        \else
           \mst@restore@resetMathstrut@
        \fi
        \let\mst@the\@gobble
    }%
  }%
\fi
```

`ynonlettersobeymathxx`
`ettersdonotobeymathxx`

```
\newtoks\mst@do@easynonletters
\ifmst@everymath
    \newcommand*\MTeasynonlettersdonotobeymathxx{\let\mst@theeasy\@gobble}%
    \newcommand*\MTeasynonlettersobeymathxx{%
                \mst@OnlyIfNotSubdued{\let\mst@theeasy\the}%
    }%
    \MTeasynonlettersobeymathxx
\else
    \newtoks\mst@undo@easynonletters
    \newcommand*\MTeasynonlettersdonotobeymathxx{\the\mst@undo@easynonletters}
    \newcommand*\MTeasynonlettersobeymathxx{%
                \mst@OnlyIfNotSubdued{\the\mst@do@easynonletters}%
    }%
    \AtEndOfPackage{\MTeasynonlettersobeymathxx}%
\fi
```

`ddtodo@easynonletters` `#1` is a one character control sequence (`\.`, `\/`, `\#`, `\%` or `\&`) and `#2` is a mathchar.

Perhaps I should use the Unicode engine `\Umathcode` et alia. I do this at other places. However I realized in 2013 and it is still true in 2024 that LATEX interface `\DeclareSymbolFont` does not allow to declare more than 16 font families even with LuaLATEX despite the latter allowing 256 such.

So why bother?

`1.4` adds to this legacy branch a test to not override a mathematically active "easy" non-letter. Main case is the dot with Babel Spanish.

```
\ifmst@everymath
    \def\mst@addtodo@easynonletters#1#2{%
      \mst@do@easynonletters\expandafter{%
          \the\mst@do@easynonletters
          \mst@OnlyIfNotMathActive{#1}{\mathcode`#1=#2}%
      }%
    }%
    \def\mst@addtodo@easynonletters@U#1#2{%
      \mst@do@easynonletters\expandafter{%
          \the\mst@do@easynonletters
          \mst@OnlyIfNotMathActive{#1}{\mst@Umathcodenum`#1=#2}%
      }%
    }%
\else
```

The character may have been made mathcode active exterior to **mathastext**. We can not test this for sure at begin document as it may happen later. If such an "easy" character is mathcode active, this can not originate in **mathastext**. So we should not overwrite when we issue `\MTeasynonlettersdonotobeymathxx`. Example I know is with babel Spanish which makes the dot math active. With PDFLaTeX it is also catcode active

This macro may be used with Unicoe engines, and LuaTeX and XeTeX differ regarding math active characters. I have only tested this (anew) for `1.4` for which this branch is needed, so in 2024.

```
    \def\mst@addtodo@easynonletters#1#2{%
      \mst@do@easynonletters\expandafter
                            {\the\mst@do@easynonletters
                             \mst@OnlyIfNotMathActive{#1}{\mathcode`#1=#2}}%
      \def\mst@tmp##1\relax{%
          \def\mst@tmp{\mst@OnlyIfNotMathActive{#1}{\mathcode`#1=##1\relax}}%
      }%
      \expandafter\mst@tmp\the\mathcode`#1\relax
      \mst@undo@easynonletters\expandafter\expandafter\expandafter
               {\expandafter\mst@tmp\the\mst@undo@easynonletters}%
    }%
    \def\mst@addtodo@easynonletters@U#1#2{%
      \mst@do@easynonletters\expandafter{%
         \the\mst@do@easynonletters
         \mst@OnlyIfNotMathActive{#1}{\mst@Umathcodenum`#1=#2}%
      }%
      \def\mst@tmp##1\relax{%
          \def\mst@tmp{\mst@OnlyIfNotMathActive{#1}{\mst@Umathcodenum`#1=##1\relax}}%
      }%
      \expandafter\mst@tmp\the\mst@Umathcodenum`#1\relax
      \mst@undo@easynonletters\expandafter\expandafter\expandafter
               {\expandafter\mst@tmp\the\mst@undo@easynonletters}%
    }%
\fi
```

`\newmcodes@`
`\mst@newmcodes@`
`\MTresetnewmcodes`
`\MTcustomizenewmcodes`

`1.15d`: the `\newmcodes@` amsmath macro causes an error in Unicode engines as soon someone assigns a Unicode mathcode to the minus sign, and then makes a `\DeclareMathOperator` declaration. Furthermore it hard-codes the font family 0 as being the one to be used. Moreover just putting the concerned signs `-`, `:`, `.`, `\`, `'`, `*` inside braces emulates enough the behavior (although the tick will give a prime).

`1.3`: now tests if 'basic' option was used.

`1.3d`: I should re-examine the situation with `\newmcodes@`. In the meantime its relaxification will not be done if lualatex-math is loaded. And the whole thing is put at begin document.

1.3m: `lualatex-math 1.5` n'a pas modifié son traitement de `\newmcodes@` mais par contre a supprimé le patch de `\resetMathstrut@`. Mais la date de release est restée à 2015/09/22 (date de 1.4a) au lieu de quelque chose comme 2016/03/13 (date pour l'annonce sur CTAN). Il faudra suivre l'évolution future de `amsmath.sty` maintenant assurée par D.C.

1.3n 2016/04/22: there is no more a patch of `\newmcodes@` by `lualatex-math 1.6 (2016/04/16)`, as `amsmath 2016/03/10 v2.15b` has now a version compatible with LuaLaTeX.

My very radical `\let\newmcodes@\relax` was only a temporary measure I adopted for lack of time on October 13, 2012, and apart from avoiding to do that in case `lualatex-math` was detected, I never came back... finally I handle it myself for 1.3n. The remaining problem of this macro (now that it does not anymore crash `lualatex` or vice versa) is that (also with `amsmath` version `2016/03/10 v2.15b`) it hardcodes the font used. The aim of the macro is to modify the type of spacing affected to symbols `'`, `*`, `.`, `-`, `/`, `:`, in case they are used in operator names.

- As I don't want to monopolize a count register only for computations, let's just be mean if ε-TeX not there.
- **mathastext** makes (or not, depending on commands issued by the user) these characters math active (the right tick already is), which complicates recovery of former mathcode. We have mathchar type *macros*, but then the complication is in diverging behaviors of the engines: `\numexpr\mst@varfam@minus\relax` works with LuaTeX, not with XeTeX.
- the `*` must presumably really be the non-lowered text glyph.
- for the `-` I hesitated but do use the hyphen in the end.
- seems I simply don't understand what the amsmath code does with `\std@minus`. It is used in `\relbar` and it escapes me why `\newmcodes@` would ever want to redefine it, and more importantly why on earth it tests the mathcode of `-` for that? yes, `\std@minus` is defined (at begin document) using the mathcode of `-`, but what's the connexion to `\newmcodes@` ?? Any way **mathastext** defines `\relbar` with `\mst@minus@sign`. Thus I just drop this conditional.
- things are complicated by the options such as `nominus`, `noparenthesis`.
- the `\newmcodes@` macro is anyhow assuming that if a new math font is used it occupies math groups `0` and `1` !! very bad; fixing it in passing if the character has not been handled by **mathastext** could be envisioned, but that's not **mathastext**'s job.
- years go by, and I remain as baffled as ever about the story of "more than 16 math families". I will not test again, but I am pretty sure that `\DeclareMathSymbol` does not work with more than 16 families, thus when I try to be a good boy and use `\Umathcode` syntax with `symmtoperatorfont` I am perhaps doing unnecessary efforts.
- I noticed that LuaLaTeX does not apply the "TeX Ligature" (bad name) regarding the right tick APOSTROPHE being transformed into RIGHT SINGLE QUOTATION MARK in math mode, but XeLaTeX does. From the point of view of **mathastext**, the behavior of XeLaTeX is the coherent one. It appears that LuaLaTeX use in math mode of a text font does not obey the set features.I opened a ticket at https://github.com/wspr/fontspec/issues/238, but as usual it is hard to figure out the best place where to report font matters. *This item might be obsolete – not checked (1.3q).*
- Some hesitation about what to do under option `symboldelimiters`. I temporarily used `\symmtpsymbol`, except for the right quote and for the hyphen, but finally I drop that and use `\symmtoperatorfont` always. (after testing how it looked like).

All in all this is a great deal of trouble and I understand I postponed back in 2012! I spent some hours on this small thing, with consequent testing and for example this TeX Ligature issue with Unicode engines.

Since `1.3v` we require e-TEX extensions, so a test for `\numexpr` has been dropped here.

```
\ifmst@basic
\else
 \ifmst@XeOrLua
  \AtBeginDocument {%
  \ifx\newmcodes@\@undefined\else
  \edef\mst@newmcodes@{%
   \mst@Umathcode `\noexpand\' 0 \symmtoperatorfont 39\relax
   \ifmst@asterisk
    \mst@Umathcode `\noexpand\* 0 \symmtoperatorfont 42\relax
   \else\mathcode`\noexpand\* 42
   \fi
   \ifmst@nopunct\mathcode `\noexpand\."613A \mathcode `\noexpand\: "603A
   \else
    \mst@Umathcode `\noexpand\. 6 \symmtoperatorfont 46\relax
    \mst@Umathcode `\noexpand\: 6 \symmtoperatorfont 58\relax
   \fi
   \ifmst@nominus\mathcode`\noexpand\- 45
   \else
      \mst@Umathcode `\noexpand\- 0 \symmtoperatorfont 45\relax
   \fi
   \ifmst@noparen\mathcode `\noexpand\/  47
   \else
      \mst@Umathcode `\noexpand\/ 0 \symmtoperatorfont 47\relax
   \fi
  }%
  \let\mst@originalnewmcodes@\newmcodes@
  \fi
  }%
 \else
  \AtBeginDocument {%
  \ifx\newmcodes@\@undefined\else
  \edef\mst@newmcodes@{%
    \mathcode`\noexpand\' \the\numexpr\symmtoperatorfont*\@cclvi+39\relax
    \mathcode`\noexpand\*
      \the\numexpr\ifmst@asterisk\symmtoperatorfont*\@cclvi\fi+42\relax
    \ifmst@nopunct\mathcode `\noexpand\."613A \mathcode `\noexpand\: "603A
    \else
      \mathcode`\noexpand\. \the\numexpr\mst@varfam@dot-"1000\relax
      \mathcode`\noexpand\: \the\numexpr\mst@varfam@colon-"1000\relax
    \fi
    \mathcode`\noexpand\-
     \the\numexpr\unless\ifmst@nominus\symmtoperatorfont*\@cclvi\fi+45\relax
    \mathcode`\noexpand\/
     \the\numexpr\unless\ifmst@noparen\symmtoperatorfont*\@cclvi\fi+47\relax\relax
    }%
  \let\mst@originalnewmcodes@\newmcodes@
  \fi
  }%
 \fi
\fi
\newcommand*\MTresetnewmcodes{\ifx\mst@originalnewmcodes@\undefined\else
                                \let\newmcodes@\mst@originalnewmcodes@\fi}
\newcommand*\MTcustomizenewmcodes{\ifx\mst@originalnewmcodes@\undefined\else
                                \let\newmcodes@\mst@newmcodes@\fi}
```

`mtoperatorfont`

Declaration of the current default font as our math font. The characteristics of the used font can be changed by a user call to the macros `\Mathastext` or `\Mathastextwilluse`, which will be defined next. We will also make one internal call to `\Mathastext` to set up the normal and bold math versions, so we will also employ `\SetSymbolFont` later.

```
\DeclareSymbolFont{mtoperatorfont}{\mst@enc}{\mst@fam}{\mst@ser}{\mst@opsh}
```

`\operator@font`

We modify this LaTeX internal variable in order for the predefined `\cos`, `\sin`, etc. . . to be typeset with the `mathastext` font. This will also work for things declared through the amsmath package command `\DeclareMathOperator`. The alternative would have been to redefine the 'operators' Math Symbol Font. Obviously people who expect that `\operator@font` will always refer to the 'operators' math font might be in for a surprise. . . well, we'll see.

`\MTmathoperatorsobeymathxx`
`\MTmathoperatorsdonotobeymathxx`

1.2: rather than just replacing `\symoperators` by `\symmtoperatorfont` I add a modification which makes the declared operator names sensitive to the math alphabets... ouh le vilain!

```
\newcommand*{\MTmathoperatorsobeymathxx}
{\def\operator@font{\mathgroup\ifnum\fam=\m@ne\symmtoperatorfont\else\fam\fi}}
\newcommand*{\MTmathoperatorsdonotobeymathxx}
{\def\operator@font{\mathgroup\symmtoperatorfont}}
\MTmathoperatorsobeymathxx
```

`mtletterfont`

At version 1.1, we add the possibility to mimick the standard behavior, that is to have italic letters and upright digits. Thanks to Tariq Perwez and Kevin Klement who asked for such a feature.

```
\DeclareSymbolFont{mtletterfont}{\mst@enc}{\mst@fam}{\mst@ser}{\mst@ltsh}
```

`\MTfixmathfonts`

There is a long-standing issue `https://github.com/lualatex/luaotfload/issues/204` on LuaLaTeX not applying OpenType features in math mode (this impacts `\url` macro too, as it uses math mode.) LuaTeX has two modes for handling of OpenType fonts, the default in text being to use the `node` mode, and this mode is non-working in math, thus **`mathastext`** needs to force use of `base` mode. Else one sees old style figures where one does not expect them, or the opposite, depending on the default font feature.

Once we know the cause, the fix is relatively easy. I will go for the `\everymath` way, because I don't want to dwelve at all with the details of LaTeX's handling of math fonts, of size changes, of math versions etc... perhaps in the future LaTeX will fix the issue upstream by modifying `\DeclareSymbolFont` under LuaLaTeX + `luaotfload` regime, then the present patch by **`mathastext`** will be unneeded. Naturally, here we care only about the two math fonts used by **`mathastext`**: `mtoperatorfont` and `mtletterfont`.

For the `\url` situation, I have posted online a patch.

Not all is resolved, as I comment online at `https://github.com/lualatex/luaotfload/issues/204#issuecomment-216465680` that with `TeX Gyre Termes` for example I can not get simultaneously Old Style and Tabular Figures to work in math mode, although the font name as constructed by my patch (which is like the code below, only simpler as we only have to consider `\textfont0`) is the correct one. Similarly with `Vollkorn`: I can then not get the two features `lnum` and `tnum` to work simultaneously when specified with `mode=base`. It does work with `mode=node` but this mode "does not work in math mode."

Done for `1.3o` of 2016/05/03.

`1.3p` renames the macro to `\MTfixmathfonts` for public access.

`1.4` intercepts also `mode=harf`. Cf `https://tex.stackexchange.com/questions/722084/change-number-style-with-mathastext` (thanks to `user691586` for bug report). The new code unconditionally replaces `mode=foo` by `mode=base`. The complications due to the output of `\fontname` using only category 12 characters are handled in a different way than the 2016 code.

The `https://github.com/latex3/fontspec/issues/525` problem causes additional complications.

Perhaps I should simply zap spaces always rather than check for quotes? Anyway this appears to work.

```
\def\mst@fixmathfonts@#1.#2.#3.{%
\def\mst@fixmathfonts@##1##2#1=##3;##4##5\relax##6\@empty##7{%
  \ifx##3\empty\else
   \if"##1%
    \font\mst@mathfont=##1##2#1=base;##4##5\relax
   \else
    \mst@arrrrgh@fixmathfonts##1##2#1=base;##4##5#2\empty#3\relax
   \fi
   ##7=\mst@mathfont
  \fi
}%
\def\mst@arrrrgh@fixmathfonts##1#2##2##3#3##4\relax{%
  \ifx##2\empty
    \font\mst@mathfont="##1"\relax
  \else
    \font\mst@mathfont="##1"#2##2##3#3\relax
  \fi
}%
}%
\expandafter\mst@fixmathfonts@\detokenize{mode. at.pt.}%
\def\MTfixmathfonts#1{%
\def\MTfixmathfonts{%
  \expandafter\mst@fixmathfonts@\fontname\textfont\symmtoperatorfont
  \relax\relax #1=;\empty\relax\@empty{\textfont\symmtoperatorfont}%
  \expandafter\mst@fixmathfonts@\fontname\scriptfont\symmtoperatorfont
  \relax\relax #1=;\empty\relax\@empty{\scriptfont\symmtoperatorfont}%
  \expandafter\mst@fixmathfonts@\fontname\scriptscriptfont\symmtoperatorfont
  \relax\relax #1=;\empty\relax\@empty{\scriptscriptfont\symmtoperatorfont}%
  \expandafter\mst@fixmathfonts@\fontname\textfont\symmtletterfont
  \relax\relax #1=;\empty\relax\@empty{\textfont\symmtletterfont}%
  \expandafter\mst@fixmathfonts@\fontname\scriptfont\symmtletterfont
  \relax\relax #1=;\empty\relax\@empty{\scriptfont\symmtletterfont}%
  \expandafter\mst@fixmathfonts@\fontname\scriptscriptfont\symmtletterfont
  \relax\relax #1=;\empty\relax\@empty{\scriptscriptfont\symmtletterfont}%
}}%
\expandafter\MTfixmathfonts\expandafter{\detokenize{mode}}%
\ifmst@LuaTeX
  \everymath\expandafter{\the\everymath\mst@fixmathfonts}%
  \everydisplay\expandafter{\the\everydisplay\mst@fixmathfonts}%
\fi
\newcommand*\MTfixfonts{\let\mst@fixmathfonts\MTfixmathfonts}%
\newcommand*\MTdonotfixfonts{\let\mst@fixmathfonts\empty}%
\MTfixfonts
```

`\Mathnormal` `\Mathrm` `\Mathbf` `\Mathit` `\Mathsf` `\Mathtt` `\Mathnormalbold`

We redefine the default normal, rm, bf, it, sf, and tt alphabets, but this will be done via `\renewcommand*{\mathrm}{\Mathrm}` etc… (not anymore, see comment below).

We follow the standard LaTeX behavior for `\mathbf`, which is to pick up the bold series of the roman font (digits and operator names).

We will access (if no option is passed for Greek) the `\omicron` via `\mathnormal`. But unfortunately the fourier package with the upright option does not have an upright omicron obtainable by simply typing `\mathnormal{o}`. So if `fourier` is loaded we use `\mathrm` and not `\mathnormal`.

Actually math alphabet macros are created robust since LaTeX from 2005, so at `1.3v` 2019/09/19 I decided to modify the old **`mathastext`** approach a bit. Indeed with the old approach a `\mathtt` in a moving argument translates ultimately into `\Mathtt` but if for example the new context where it gets expanded is a subdued normal math version, this does not give the same as `\mathtt` would have given there. This was a bug: imagine `\section{$\mathtt{X}$}` issued in a math version, but the TOC is done in subdued normal version; the output in TOC will often differ (fontsize being put aside) both from out it looked at the section title and from what direct usage of `\mathtt` in the TOC would have given. I have no strong preference between the two possibilities (to be as in section title, or to be as if `\mathtt` gets executed in TOC and obeys its local regime), but it is a bug if the result is still a third one. Thus I decided to follow LaTeX2e and that `\mathtt` had to remain `\mathtt` when moving.

But a math alphabet command such as `\Mathtt` redefines its unprotected meaning on first use as well as the one of the math version macro, hence a `\letrobustmacro\mathtt\Mathtt` of sorts is no good at all. I thus opted to not hack into the math LaTeX font support across math versions and to simply use `\protected\def` in place of obeying strictly LaTeX2e robustness (except of course in the subdued math versions as there the math alphabets acquire back their original robust meanings.)

Potential breaking change at `1.3za`, the `defaultbf` etc... options do not prevent the package declaring `\Mathbf` etc... commands.

`1.3za` defines a `\Mathnormalbold` and then defines `\mathnormalbold` in terms of it in place of defining directly `\mathnormalbold` as a math alphabet. This is in relation to implementation of the `LGRgreek+` option. There was some hesitation though to restrict this change to that option only or not.

```
\let\mst@alph@omicron\mathnormal
\@ifpackageloaded{fourier}{\ifsloped\else\let\mst@alph@omicron\mathrm\fi}{}
\DeclareSymbolFontAlphabet{\Mathnormal}{mtletterfont}
\DeclareSymbolFontAlphabet{\Mathrm}{mtoperatorfont}
    \DeclareMathAlphabet{\Mathnormalbold}{\mst@enc}{\mst@fam}{\mst@bold}{\mst@ltsh}
    \protected\def\mathnormalbold{\Mathnormalbold}
    \DeclareMathAlphabet{\Mathbf}{\mst@enc}{\mst@fam}{\mst@bold}{\mst@opsh}
    \DeclareMathAlphabet{\Mathit}{\mst@enc}{\mst@fam}{\mst@ser}{\itdefault}
```

```
  \DeclareMathAlphabet{\Mathsf}{\mst@enc}{\sfdefault}{\mst@ser}{\mst@opsh}
  \DeclareMathAlphabet{\Mathtt}{\mst@enc}{\ttdefault}{\mst@ser}{\mst@opsh}
```

The `\mathxx` macros being LaTeX2e robust, or course the meanings here are known, and «original» macros are sort of superfluous but well it works.

```
\let\mst@original@normal\mathnormal
\let\mst@original@rm\mathrm
\let\mst@original@bf\mathbf
\let\mst@original@it\mathit
\let\mst@original@sf\mathsf
\let\mst@original@tt\mathtt
\def\mst@restorealphabets{%
    \let\mathnormal\mst@original@normal
    \let\mathrm\mst@original@rm
    \let\mathbf\mst@original@bf
    \let\mathit\mst@original@it
    \let\mathsf\mst@original@sf
    \let\mathtt\mst@original@tt
  }
\ifmst@greekplus
```

`\mst@mathalph` The LaTeX kernel code regarding math fonts is too complex and rigid for there to be a robust and easy way to know when one is in the argument of `\mathrm` or `\mathbf`, and the code is spread on various chapters of `source2e.pdf`, and the comments are often not up-to-date. So I did not try a too complex hack and decided for `1.3za` to add a numeric indicator to let Greek letters react to it. It incorporates a space to be self-delimiting in an `\ifcase` to maintain expandability in numeric context of the to-be-defined Greek control sequences.

```
  \def\mst@mathalph{-1}
  \def\mst@setalphabets{%
    \protected\def
    \mathnormalbold##1{\def\mst@mathalph{4 }\Mathnormalbold{##1}\def\mst@mathalph{-1 }}%
    \ifmst@defaultnormal\else
        \protected\def
        \mathnormal##1{\def\mst@mathalph{0 }\Mathnormal{##1}\def\mst@mathalph{-1 }}%
    \fi
    \ifmst@defaultrm\else
        \protected\def
        \mathrm##1{\def\mst@mathalph{1 }\Mathrm{##1}\def\mst@mathalph{-1 }}%
    \fi
    \ifmst@defaultbf\else
        \protected\def
        \mathbf##1{\def\mst@mathalph{2 }\Mathbf{##1}\def\mst@mathalph{-1 }}%
    \fi
    \ifmst@defaultit\else
        \protected\def
        \mathit##1{\def\mst@mathalph{3 }\Mathit{##1}\def\mst@mathalph{-1 }}%
    \fi
    \ifmst@defaultsf\else\protected\def\mathsf{\Mathsf}\fi
    \ifmst@defaulttt\else\protected\def\mathtt{\Mathtt}\fi
  }
```

```
\else
  \def\mst@setalphabets{%
    \ifmst@defaultnormal\else\protected\def\mathnormal{\Mathnormal}\fi
    \ifmst@defaultrm\else\protected\def\mathrm{\Mathrm}\fi
    \ifmst@defaultbf\else\protected\def\mathbf{\Mathbf}\fi
    \ifmst@defaultit\else\protected\def\mathit{\Mathit}\fi
    \ifmst@defaultsf\else\protected\def\mathsf{\Mathsf}\fi
    \ifmst@defaulttt\else\protected\def\mathtt{\Mathtt}\fi
  }
\fi
\ifmst@subdued\else\mst@setalphabets\fi
```

LGRgreek
\MTgreekupdefault
\MTgreekitdefault
selfGreek

`1.14b`: We can not move the `\DeclareSymbolFont` to the `\Mathastext` macro because it resets the font family in *all* math versions, and some could have been defined by the user with previous calls to `\Mathastext`. So we have to have them here. The problem is that at this stage it is impossible to know if we really need (in the case of LGRgreek) two separate shapes for upper and lowercase, and (in the case of selfGreek) a shape distinct from the one used in `mtoperatorfont`. So I opted in the end for declaring possibly one too many font. To achieve more economy the only way would be to keep cumulative track of all previously declared math versions and to redeclare appropriately the LGR or self greek fonts at each call to `\Mathastext` (with no optional argument): a bit painful, and as I am possibly the sole user in the world of this possibility of multiple math versions with this package. Also the advantage to systematically allocate a font for the selfGreek option is that we can force the use of the OT1 encoding.

First we establish the cumulative effect of the greek related options.

`1.15c` introduces some possibilities to change the shapes of Greek letters in each math versions, and even the Greek font (in LGR encoding). The commands `\MTitgreek` etc... will be used in-between calls to `\Mathastext` and re-adjust the shapes. And the command `\MTgreekfont` changes the Greek font family.

Note that `\mst@ltsh` expands to `\shapedefault` or `\itdefault` at this location.

Note added 2022/11/02: using `\MTitgreek` etc... once implies that from then on, for subsequent **mathastext**-math versions, the shape of Greek letters will not be kept in sync with the shape and lettershape version parameters, but only react to the configuration decided by these commands (and `italic/frenchmath` options).

Note 2022/10/29: for some time `\updefault` was made into `up` by LaTeX (since 2020-02-02 now that I check this out). As a result this triggered Font Warnings in the log about the replacement of `up` by `n`.

`1.3y` refactors completely the handling of Greek letter shapes under the `LGRgreek(s)` options (and only under them). Under these options we don't use one font for lowercase Greek and anotherone for uppercase Greek (some above code comments have not been updated) but one math font `mtgreekit` for italic Greek and one math font `mtgreekup` for upright Greek. What 'italic' and 'upright' mean is decided by the expansion of `\MTgreekitdefault` and `\MTgreekupdefault`, which give respectively `it` and `n` per default.

If no `itgreek` et al. options or `\MTitgreek` et al. commands have been used, we need to map `\mst@ltsh` (which was used for lowercase Greek, except under `frenchmath` option) and `\mst@opsh` to either 'italic' or 'upright'. This is done by testing if they hold 'it' or 'sl'. If yes we map to 'italic' by setting to false an 'up' Boolean, if not we leave the 'up' Boolean to true.

In order to maintain perfect identical code for non-`LGRgreek`, the `LGRgreek` related code is simply added to previously shared constructions. The `LGRgreek` behavior will remain identical in most documents, but for example those who used some adventurous '`sc`' for the main shape (the one used per default for operator names) need to adjust `\MTgreekupdefault` to be ‘`sc`’, for the math version being defined, or the default one if this is followed by usage of `\Mathastext`.

The new `LGRgreek`-specific commands `\MTgreekupdefault` and `\MTgreekitdefault` are the only ones in the package which can possibly be defined previously to loading it. (Perhaps some other macros could be also converted to being modifiable prior to loading `mathastext`, thus avoiding potential need to use `\Mathastext` at least once after loading the package; to be examined next time — which may be a long time in future!).

Unfortunately the `1.3y` did some internal renamings here (using `@lgr@` in macro names in place of `@greek@`) which were not everywhere followed up, and this broke the `selfGreek` option. Fixed at `1.3z`.

Prior to `1.4e`, the code of `\mst@update@greeksh`, which starts with a definition `\mst@greek@lsh` also updated it, depending on branches. But, contrarily to `\mst@greek@ush` it is used nowhere else. Also the code now has two parameters `#1#2` for it to be sharable with `\MTDeclareVersion@@` needs.

```
\providecommand*\MTgreekupdefault{n}
\providecommand*\MTgreekitdefault{it}
\newif\ifmst@greek@lower@up
\newif\ifmst@greek@upper@up
\def\mst@update@greeksh#1#2{
    \def\mst@greek@lsh{#1}
    \def\mst@greek@ush{#2}
    \mst@greek@lower@uptrue
      \expandafter\in@\expanded{{\mst@greek@lsh.}}{it.,sl.}%
      \ifin@\mst@greek@lower@upfalse\fi
    \mst@greek@upper@uptrue
      \expandafter\in@\expanded{{\mst@greek@ush.}}{it.,sl.}%
      \ifin@\mst@greek@upper@upfalse\fi
    \ifmst@itgreek
      \def\mst@greek@ush{\MTgreekitdefault}
        \mst@greek@lower@upfalse
        \mst@greek@upper@upfalse
    \fi
    \ifmst@upgreek
      \def\mst@greek@ush{\MTgreekupdefault}
        \mst@greek@lower@uptrue
        \mst@greek@upper@uptrue
    \fi
    \ifmst@frenchmath
      \ifmst@itgreek\else
      \ifmst@upgreek\else
        \def\mst@greek@ush{#2}
          \mst@greek@lower@uptrue
          \mst@greek@upper@uptrue
      \fi\fi
    \fi
    \ifcase\mst@greek@select
        \or
          \def\mst@greek@ush{\MTgreekitdefault}
            \mst@greek@upper@upfalse
```

```
       \or
          \def\mst@greek@ush{\MTgreekupdefault}
            \mst@greek@upper@uptrue
    \fi
}
\mst@update@greeksh{\mst@ltsh}{\mst@opsh}
```

`mtgreekup`
`mtgreekit`
`\mathgreekup`
`\mathgreekit`
`\mathgreekupbold`
`\mathgreekitbold`

The 1.3y refactoring was done in order to be able to define `\alphaup`, etc … control sequences (`\mathchar`'s), as well as the italic ones. Formerly two math fonts were created but to be used respectively with lowercase or uppercase Greek. Now we have two fonts indexed by their shape, and we take advantage to create two math alphabets mapping to the two defined symbol fonts `mtgreekup` and `mtgreekit`.

1.3za adds `\mathgreekupbold` and `\mathgreekitbold`.

```
\ifmst@LGRgreek
    \DeclareFontEncoding{LGR}{}{}
    \DeclareSymbolFont{mtgreekup}{LGR}{\mst@fam}{\mst@ser}{\MTgreekupdefault}
    \DeclareSymbolFont{mtgreekit}{LGR}{\mst@fam}{\mst@ser}{\MTgreekitdefault}
    \DeclareSymbolFontAlphabet{\mathgreekup}{mtgreekup}
    \DeclareSymbolFontAlphabet{\mathgreekit}{mtgreekit}
    \DeclareMathAlphabet{\mathgreekupbold}{LGR}{\mst@fam}{\mst@bold}{\MTgreekupdefault}
    \DeclareMathAlphabet{\mathgreekitbold}{LGR}{\mst@fam}{\mst@bold}{\MTgreekitdefault}
\else
```

`mtselfGreekfont`

```
\ifmst@selfGreek
    \DeclareSymbolFont{mtselfGreekfont}{OT1}{\mst@fam}{\mst@ser}{\mst@greek@ush}
\fi\fi
```

`mteulervm`
`\MathEuler`
`\MathEulerBold`

In case we need the Euler font, we declare it here. It will use `uzeur.fd` from the eulervm package of Walter Schmidt

```
\ifmst@needeuler
\mst@infoline{will use Euler font; command \string\MTEulerScale}
  \DeclareSymbolFont{mteulervm}{U}{zeur}{m}{n}
  \DeclareSymbolFontAlphabet{\MathEuler}{mteulervm}
  \DeclareMathAlphabet{\MathEulerBold}{U}{zeur}{\mst@bold}{n}
\fi
\newcommand*\MTEulerScale[1]{\edef\zeu@Scale{#1}}
\let\MathastextEulerScale\MTEulerScale
```

LaTeX 2$_\varepsilon$ has a strange initial configuration where the capital Greek letters are of type `mathalpha`, but the lower Greek letters of type `mathord`, so that `\mathbf` does not act on them, although lowercase Greek letters and Latin letters are from the same font. This is because `\mathbf` is set up to be like a bold version of `\mathrm`, and `\mathrm` uses the 'operators' font, by default `cmr`, where there are NO lowercase greek letters. This set-up is ok for the Capital Greek letters which are together with the Latin letters in both `cmmi` and `cmr`.

The package eulervm sets the lowercase Greek letters to be of type `mathalpha`, the default `\mathbf` and `\mathrm` will act wierdly on them, but a `\mathbold` is defined which will use the bold series of the Euler roman font, it gives something coherent for Latin and Greek *lowercase* letters, and this is possible because the same font contains upright forms for them all.

Here in `mathastext`, Latin letters and Greek letters (lower and upper case) must be (generally) assumed to come from two different fonts, as a result the standard `\mathbf` (and `\mathrm`) will give weird results when used for Greek letters. We could coerce `\mathbf` to do something reasonable (cf `http://tug.org/pipermail/texhax/2011-January/016605.html`) but at this time `30-01-2011 09:42:27 CET` I decided I would not try to implement it here. I prefer to respect the default things.

I followed the simpler idea of the eulervm package and defineed `\MathEuler` and `\MathEulerBold` alphabet commands (the eulervm package does this only for the bold font).

`mtpsymbol`
`\MathPSymbol`

In case we need the Symbol font, we declare it here. The macro `\psy@scale` will be used to scale the font (see at the very end of this file).

```
\ifmst@needsymbol
  \mst@infoline{will use Symbol font; command \string\MTSymbolScale}
  \def\psy@scale{1}
  \DeclareSymbolFont{mtpsymbol}{U}{psy}{m}{n}
  \DeclareSymbolFontAlphabet{\MathPSymbol}{mtpsymbol}
  \AtBeginDocument{%
    \DeclareFontFamily{U}{psy}{}%
    \DeclareFontShape{U}{psy}{m}{n}{<->s*[\psy@scale] psyr}{}%
  }
\fi
\newcommand*\MTSymbolScale[1]{\edef\psy@scale{#1}}
\let\MathastextSymbolScale\MTSymbolScale
```

I did not choose for name `\MathSymbol` as this may be defined somewhere for another thing. There is no bold for the postscript Symbol font distributed with the $\LaTeX\,2_\varepsilon$ `psnfss` core package.

`\pmvec`

Definition of a poor man version of the `\vec` accent. Done using `\protected\def` at 1.4.

```
\protected\def\pmvec#1{%
    \mathord{\stackrel{\raisebox{-.5ex}{\tiny\boldmath$\mathord{\rightarrow}$}}%
                      {{}#1}%
            }%
}
```

`\fouriervec`

The glyph is taken from the Fourier font of Michel Bovani. Note: (oct 2012) I should not allocate an entire symbol font just for one glyph! But I have not given any serious thought to what one can do to simulate a math accent without doing such a wasteful thing.

```
\ifmst@fouriervec
  \DeclareFontEncoding{FML}{}{}
  \DeclareFontSubstitution{FML}{futm}{m}{it}
  \DeclareSymbolFont{mathastextfourier}{FML}{futm}{m}{it}
  \SetSymbolFont{mathastextfourier}{bold}{FML}{futm}{b}{it}
  \mst@DeclareMathAccent{\fouriervec}{\mathord}{mathastextfourier}{"7E}
\fi
```

`\MTencoding`
`\MTfamily`
`\MTseries`
`\MTshape`
`\MTboldvariant`
`\MTlettershape`

Some public macros to modify our private internals, and we will use them also ourself.

In version 1.1 we add the possibility to have two distinct font shapes for letters and digits. So in fact we could as well have two really unrelated fonts but this is really not the spirit of the package.

Note that using these macros in the preamble allows `\Mathastext` to set up math versions with a given font for math mode, and at the same time not modifying the `\familydefault` or `\romandefault` etc. . .

At time of `1.3za` I considered letting `LGRgreeks` and `selfGreeks` support `\MTgreekfont` and this needed a chaneg to `\MTfamily` here but I dropped the idea. Too wary of documentation changes.

```
\newcommand*\MTencoding[1]{\def\mst@enc{#1}}
\newcommand*\MTfamily[1]{\def\mst@fam{#1}}
\newcommand*\MTseries[1]{\def\mst@ser{#1}}
\newcommand*\MTshape[1]{\def\mst@opsh{#1}\ifmst@italic\else\def\mst@ltsh{#1}\fi}
\newcommand*\MTboldvariant[1]{\def\mst@bold{#1}}
\newcommand*\MTlettershape[1]{\def\mst@ltsh{#1}}
\let\Mathastextencoding\MTencoding
\let\Mathastextfamily\MTfamily
\let\Mathastextseries\MTseries
\let\Mathastextshape\MTshape
\let\Mathastextboldvariant\MTboldvariant
\let\Mathastextlettershape\MTlettershape
```

`\MTitgreek`
`\MTupgreek`
`\MTitGreek`
`\MTupGreek`
`\MTgreekfont`

`1.15c`: These new macros can be used in-between calls to `\Mathastext`. They reset the shapes for Greek letters (applies to LGRgreek(s) and selfGreek(s) options). The `\MTgreekfont` presupposes either LGRgreek or selfGreek (it is inoperant under `LGRgreeks` or `selfGreeks`). `\MTgreekfont{\familydefault}` is somewhat like using LGRgreeks or selfGreeks.

At time of `1.3za` I let `\MTgreekfont` also have an effect under option `LGRgreeks` or `selfGreeks`, via a refactoring which also modified `\MTfamily`.

```
\newcommand*\MTitgreek{\mst@itgreektrue\mst@upgreekfalse\def\mst@greek@select{0}}
\newcommand*\MTupgreek{\mst@upgreektrue\mst@itgreekfalse\def\mst@greek@select{0}}
\newcommand*\MTitGreek{\def\mst@greek@select{1}}
\newcommand*\MTupGreek{\def\mst@greek@select{2}}
\let\Mathastextitgreek\MTitgreek
\let\Mathastextupgreek\MTupgreek
\let\MathastextitGreek\MTitGreek
\let\MathastextupGreek\MTupGreek
\newcommand*\MTgreekfont[1]{\def\mst@greekfont{#1}}
\let\Mathastextgreekfont\MTgreekfont
```

At (long...) last we now change the font for the letters of the latin alphabet. In version `1.1`, Latin letters have their own font (shape).

`1.2b` 2012/12/28 now that we understand the great advantages of "8000 we do it also for all letters a-z and A-Z to insert automatically the italic corrections. See the discussion in the user manual. Ironically I wrote the code initially for the `italic` option only to realize later it was more suitable to using an *upright* text font in math mode! So this mathematical activation of the letters is not done if the font shape is detected to be `it` or `sl`; to bypass this the command `\MTicinmath` is provided.

`1.2e` 2013/01/10 corrects a bad oversight of `1.2b` in `\mst@mathactivate` which made the reproduction of the user manual illustrations with `$f_i^i$` impossible. As `\mst@mathactivate` was originally used also to get the non-letters obey math alphabet while maintaining the TEX spacings, it added no extra braces. The braces should however be added for expansion of math active letters, in order of things like `x^y` to work as expected. (the group braces do not prevent ligatures when the letters are arguments to the math alphabet commands, the added macros `\mst@itcorr` and `\mst@before@<letter>` expanding to nothing).

Added note 2016/01/06: it should be explicitly said that the extra `{..}` in `\mst@mathactivate` for letters end up creating `\hbox`'es around each letter with its extra skips and explicit italic correction, when present. These skips are thus set at natural width and do not add any break point.

Added note at 1.4: the extra pair of braces is inserted here at `\mst@DeclareMathLetter`, not at `\mst@mathactivate`.

1.3 2013/09/02 extends the use of mathematically active letters to allow the user to specify muglue before and after the letter itself (see `\MTsetmathskips`, below). Mathematically active letters were previously used only to add the italic correction; the math activation has now been separated and put in `\MTmathactiveletters`. There is also `\MTmathactiveLetters` to allow math activation only for the uppercase letters. To cancel the (now default, even with option `italic`) math activation of letters, there is `\MTmathstandardletters`. Version 1.3a removes some silly `\string`'s from the code, which prevented to pass macros as first argument to the command.

Added note 2016/01/06: Notice that the initially `\relax` tokens `\mst@[before|after]@<letter>` formed with `\csname...\endcsname` do not modify TeX's math layout: `{\relax f\relax}` is like `f` (also for ligatures inside `\mathrm` for example).

The code here was refactored at 1.4 and this simplified `\mst@mathactivate` definition.

The `\mst@before`⟨*letter*⟩ were renamed at 1.4 into `\mst@before@`⟨*letter*⟩ and are incorporated at the `\mst@DeclareMathLetter` location. Formerly `\mst@`⟨*letter*⟩ was only the math symbol, now `\MTmathcharletter`⟨*letter*⟩ is used for that and `\MTcommandletter`⟨*letter*⟩ is the whole thing to which the active letter expands (the definition of the active letter done by `\mst@mathactivate` is done with a `\def`, not a `\let`).

`mst@DeclareMathLetter` 1.4b adds `noletters` option.

```
\ifmst@noletters
  \let\MTmathactiveletters\@empty
  \let\MTmathactiveLetters\@empty
  \let\MTmathstandardletters\@empty
\else
\def\mst@DeclareMathLetter#1#2#3#4#5{%
    \DeclareMathSymbol {#1}{\mathalpha}{mtletterfont}{`#1}%
    \DeclareMathSymbol {#4}{\mathalpha}{mtletterfont}{`#1}%
    \def#2{{#3#4#5\mst@itcorr}}%
}%
\@tfor\mst@tmp:=abcdefghijklmnopqrstuvwxyz\do{%
   \expandafter\expandafter\expandafter\mst@DeclareMathLetter
   \expandafter\mst@tmp
               \csname MTcommandletter\mst@tmp\expandafter\endcsname
               \csname mst@before@\mst@tmp\expandafter\endcsname
               \csname MTmathcharletter\mst@tmp\expandafter\endcsname
               \csname mst@after@\mst@tmp\endcsname
}%
\ifmst@frenchmath \def\mst@font@tbu{mtoperatorfont}%
   \else          \def\mst@font@tbu{mtletterfont}%
\fi
\def\mst@DeclareMathLetter #1#2#3#4#5{%
```

```
    \DeclareMathSymbol {#1}{\mathalpha}{\mst@font@tbu}{`#1}%
    \DeclareMathSymbol {#4}{\mathalpha}{\mst@font@tbu}{`#1}%
    \def#2{{#3#4#5\mst@ITcorr}}%
}%
\@tfor\mst@tmp:=ABCDEFGHIJKLMNOPQRSTUVWXYZ\do{%
   \expandafter\expandafter\expandafter\mst@DeclareMathLetter
   \expandafter\mst@tmp
               \csname MTcommandletter\mst@tmp\expandafter\endcsname
               \csname mst@before@\mst@tmp\expandafter\endcsname
               \csname MTmathcharletter\mst@tmp\expandafter\endcsname
               \csname mst@after@\mst@tmp\endcsname
}%
\def\mst@mathactivate@lowercase{%
    \@tfor\mst@tmp:=abcdefghijklmnopqrstuvwxyz\do{%
      \expandafter\expandafter\expandafter
      \mst@mathactivate\expandafter\mst@tmp\csname MTcommandletter\mst@tmp\endcsname{}%
     }%
}%
\def\mst@mathactivate@uppercase{%
    \@tfor\mst@tmp:=ABCDEFGHIJKLMNOPQRSTUVWXYZ\do{%
      \expandafter\expandafter\expandafter
      \mst@mathactivate\expandafter\mst@tmp\csname MTcommandletter\mst@tmp\endcsname{}%
     }%
}%
\def\mst@mathdeactivate@lowercase{%
    \@tfor\mst@tmp:=abcdefghijklmnopqrstuvwxyz\do{%
      \expandafter\mathcode\expandafter`\mst@tmp=\csname MTmathcharletter\mst@tmp\endcsna
     }%
}%
\def\mst@mathdeactivate@uppercase{%
    \@tfor\mst@tmp:=ABCDEFGHIJKLMNOPQRSTUVWXYZ\do{%
      \expandafter\mathcode\expandafter`\mst@tmp=\csname MTmathcharletter\mst@tmp\endcsna
     }%
}%
```

We redo the definitions with some added layer (silly because will never happen in pratice that a letter is Babel-active) related to `\mst@do@activecase` in babel context.

```
\ifmst@everymath
\else
  \def\mst@mathdeactivate@lowercase{%
    \@tfor\mst@tmp:=abcdefghijklmnopqrstuvwxyz\do{%
      \expandafter\mathcode\expandafter`\mst@tmp=
             \csname MTmathcharletter\mst@tmp\endcsname
     }%
  }%
  \def\mst@mathdeactivate@uppercase{%
    \@tfor\mst@tmp:=ABCDEFGHIJKLMNOPQRSTUVWXYZ\do{%
      \expandafter\mathcode\expandafter`\mst@tmp=
             \csname MTmathcharletter\mst@tmp\endcsname
     }%
  }%
\fi
```

\MTmathactiveletters
MTmathstandardletters

Important changes at 1.4.

Ascii letters are math-activated at package loading time, rather than again and again each time math mode is entered. But not under `subdued` mode, then the activation is done as part of switching to some math version.

The ordering in this code source is a bit of a mess, and it may be too early here to execute `\MTmathactiveletters` (I think it is fine now after some major moving around of code chunks but will not waste time checking it) so this will be postponed to end of package loading. Prior to 1.4 `\MTmathactiveletters` could be executed here as it only set some toggle to be obeyed at every math mode entrance.

```
\def\mst@mathactivateletters{%
    \mst@mathactivate@lowercase
    \mst@mathactivate@uppercase
}%
\newcommand*\MTmathactiveletters{%
    \mst@OnlyIfNotSubdued
    \mst@mathactivateletters
}%
\AtEndOfPackage{\MTmathactiveletters}%
```

Duplication due to some legacy reasons, do not rely on these internal macro names, beware they change at future release.

```
\def\mst@mathactivateLetters{\mst@activate@uppercase}%
\newcommand*\MTmathactiveLetters{%
    \mst@OnlyIfNotSubdued
    \mst@mathactivateLetters
}%
\newcommand*\MTmathstandardletters{%
    \mst@mathdeactivate@lowercase
    \mst@mathdeactivate@uppercase
}%
\fi
```

\MTicinmath
\MTICinmath
\MTnoicinmath
\MTicalsoinmathxx

`\MTnoicinmath` can also be used from inside math mode.

`\MTicalsoinmathxx` is destined to be used inside `\mathnormalbold` as I didn't want to add the complication of extracting the family number used inside `\mathnormalbold` (will perhaps come back if I have time to spend on `source2e`). Added note `2016/01/06`: this number is a priori simply `symmtletterfont+1`.

`\MTicinmath` can also be used inside math mode, to revert an earlier `\MTnoicinmath` from inside the same math group.

1.3i `2016/01/06`: For some reason which I have now forgotten I did until then:

```
% \def\mst@itcorr{\ifnum\fam=\m@ne\/\else\ifnum\fam=\symmtletterfont\/\fi\fi}%
%
```

hence italic corrections were also applied inside `\mathnormal` (for upright fonts; `\mathnormalbold` math alphabet was not treated like `\mathnormal`). I now drop this to be more in sync with the handling of the extra skips around letters. Everything gets suppressed inside all math alphabets, allowing ligatures, even for `\mathnormal`.

```
\newcommand*\MTicinmath{%
    \MTmathactiveletters
    \def\mst@itcorr{\ifnum\fam=\m@ne\/\fi}%
    \let\mst@ITcorr\mst@itcorr
}
\newcommand*\MTICinmath{%
    \MTmathactiveLetters
    \def\mst@ITcorr{\ifnum\fam=\m@ne\/\fi}%
}
\newcommand*\MTnoicinmath{\let\mst@itcorr\@empty\let\mst@ITcorr\@empty}
\newcommand*\MTnoICinmath{\let\mst@ITcorr\@empty}
\newcommand*\MTicalsoinmathxx{%
    \ifx\mst@itcorr\@empty\else\def\mst@itcorr{\/}\fi
    \ifx\mst@ITcorr\@empty\else\def\mst@ITcorr{\/}\fi
}
```

`\MTsetmathskips`
`\MTunsetmathskips`

1.3 2013/09/02: user level command to specify extra spaces in math mode around the letters (only the 7bit a,b,..,z and A,B,..,Z). First parameter is the letter, second is the math skip to be inserted before, and third the skip to be inserted after; for example `\thickmuskip` or explicitly `0.1mu`.

For this, letters are made mathematically active. This is now the package default (version 1.2 did this only in the absence of option `italic`, or more precisely when the font used was not of shape `it` or `sl`). But if `\MTsetmathskips` has not been used for that letter, the only effect of the math activation is, as in 1.2, to add the italic correction automatically, except when the font shape is detected to be `it` or `sl`; in these latter cases, although mathematically active, the letter acts in the standard way.

The command `\MTmathstandardletters` turns off math activation and its effects for all letters.

Ligatures within the argument of a math alphabet command are impeached by skips; so `\MTunsetmathskips` is provided to cancel the skips for one specific letter (`f` for example).

1.3a 2013/09/04: I strangely had `\string#1` inside `\MTsetmathskips`. Phobic of catcode active letters... but with `\string` one needs some `\expandafter` to use `\MTsetmathskips` in an `\@for` loop for example. It is better to allow the first argument to be a macro or anything expanding to a letter, and to not be paranoid about improbable catcode active letters (the user just has to tame them at the time of the `\MTsetmathskip`) so I take out these `\string`'s.

1.3i 2016/01/06: the extra skips are suppressed for the arguments of math alphabet commands. This applies in particular for amsmath's `\DeclareMathOperator`.

```
\newcommand*\MTsetmathskips[3]{%
    \@namedef{mst@before@#1}{\ifnum\fam=\m@ne\mskip#2\relax\fi}%
    \@namedef{mst@after@#1}{\ifnum\fam=\m@ne\mskip#3\relax\fi}%
}
\newcommand*\MTunsetmathskips[1]{%
    \@namedef{mst@before@#1}{}%
    \@namedef{mst@after@#1}{}%
}
```

`\mst@DeclareMathDigit`
`\MTmathactivedigits`
`\MTmathstandarddigits`

In version 1.1, we have now separated digits from letters, so paradoxically it is less problematic to give them the `mathalpha` type.

1.4 had defined some no-op `\MTmathstandarddigits` and `\MTmathactivedigits` but if not under option `nodigits`. As they are used in `\MTeverymathoff` and `\MTeverymathdefault` though this broke the option `nodigits` (either with `subdued` or when using `\MTversion`).

Defining them as `\empty` so that LaTeX defensive mechanism with `\newcommand` works but this is sooooooo hypothetical context!

```
\let\MTmathactivedigits\empty
\let\MTmathstandarddigits\empty
\ifmst@nodigits\else
  \def\mst@font@tbu{mtoperatorfont}%
  \ifmst@symboldigits \def\mst@font@tbu{mtpsymbol}\fi
  \ifmst@eulerdigits  \def\mst@font@tbu{mteulervm}\fi
  \DeclareMathSymbol{0}{\mathalpha}{\mst@font@tbu}{`0}%
  \DeclareMathSymbol{1}{\mathalpha}{\mst@font@tbu}{`1}%
  \DeclareMathSymbol{2}{\mathalpha}{\mst@font@tbu}{`2}%
  \DeclareMathSymbol{3}{\mathalpha}{\mst@font@tbu}{`3}%
  \DeclareMathSymbol{4}{\mathalpha}{\mst@font@tbu}{`4}%
  \DeclareMathSymbol{5}{\mathalpha}{\mst@font@tbu}{`5}%
  \DeclareMathSymbol{6}{\mathalpha}{\mst@font@tbu}{`6}%
  \DeclareMathSymbol{7}{\mathalpha}{\mst@font@tbu}{`7}%
  \DeclareMathSymbol{8}{\mathalpha}{\mst@font@tbu}{`8}%
  \DeclareMathSymbol{9}{\mathalpha}{\mst@font@tbu}{`9}%
```

1.4 adds possibility of mathematically active digits.

```
  \ifmst@activedigits
    \def\mst@DeclareMathDigit #1#2#3{%
      \DeclareMathSymbol{#3}{\mathalpha}{\mst@font@tbu}{`#1}%
      \def#2{#3}%
    }%
    \@tfor\mst@tmp:=0123456789\do{%
      \expandafter\expandafter\expandafter\mst@DeclareMathDigit
      \expandafter\mst@tmp
               \csname MTcommanddigit\romannumeral\mst@tmp\expandafter\endcsname
               \csname MTmathchardigit\romannumeral\mst@tmp\endcsname
    }%
    \def\mst@mathactivatedigits{%
      \@tfor\mst@tmp:=0123456789\do{%
      \expandafter\expandafter\expandafter\mst@mathactivate
      \expandafter\mst@tmp\csname MTcommanddigit\romannumeral\mst@tmp\endcsname{}%
     }%
    }%
    \def\MTmathactivedigits{\mst@OnlyIfNotSubdued\mst@mathactivatedigits}%
    \MTmathactivedigits
    \ifmst@everymath
      \def\MTmathstandarddigits{%
        \@tfor\mst@tmp:=0123456789\do{%
        \expandafter\mathcode\expandafter`\mst@tmp
            =\csname MTmathchardigit\romannumeral\mst@tmp\endcsname
         }%
      }%
    \else
```

We inject some extra layer (silly because will never happen in pratice that a digit token is

Babel-active) related to `\mst@do@activecase` in babel context.

```
      \def\MTmathstandarddigits{%
        \@tfor\mst@tmp:=0123456789\do{%
        \expandafter\mathcode\expandafter`\mst@tmp
            =\csname MTmathchardigit\romannumeral\mst@tmp\endcsname
         }%
      }%
    \fi
  \fi
\fi
```

When `symboldelimiters` is passed as an option, we use the Symbol font for the printable characters other than letters and digits.

```
\ifmst@symboldelimiters
\def\mst@font@tbu{mtpsymbol}%
\mst@endashfalse
\mst@emdashfalse
\else
\def\mst@font@tbu{mtoperatorfont}%
\fi
```

1.2 adds the tricks to let non letters/digits obey math alphabets. We have to double the definitions for easy switch on-off of the mechanism, via a token list which is put into `\everymath` and `\everydisplay`.

```
\ifmst@noexclam\else\mst@infoline{\string! and \string?}%
\DeclareMathSymbol{!}{\mathclose}{\mst@font@tbu}{"21}%
\DeclareMathSymbol{\mst@varfam@exclam}{\mathalpha}{\mst@font@tbu}{"21}%
\expandafter\mst@addtodo@nonletters\string!\mathclose\mst@varfam@exclam
\DeclareMathSymbol{?}{\mathclose}{\mst@font@tbu}{"3F}%
\DeclareMathSymbol{\mst@varfam@question}{\mathalpha}{\mst@font@tbu}{"3F}%
\expandafter\mst@addtodo@nonletters\string?\mathclose\mst@varfam@question
\fi
```

`\MTlowerast`
`\mst@doasterisk`
`\mst@@doasterisk`
`\MTnormalasterisk`
`\MTactiveasterisk`

1.12d The `\ast` or `*` is defined in `fontmath.ltx` as a binary operator from the `symbols` font. Usually the asterisk from the text font is in a raised position. Previous versions of **mathastext** did nothing with `\ast` but strangely defined `*` to be the one from the text font, with type `\mathalpha`. The package now leaves by default both `*` and `\ast` untouched, and if passed option `asterisk` replaces both of them with a lowered text asterisk (or the one from the Symbol font), and of type `\mathbin`. A trick is used to optionally get both `*` and `\ast` obey the math alphabets.

The user macro `\MTlowerast` sets the amount of lowering to be applied to the text asterisk.

1.12e Somehow there was a big omission in 1.12d, the command `\MTlowerast` as described in the manual was missing!

1.3i adds `\MTnormalasterisk` and `\MTactiveasterisk`. They do nothing if package is loaded without option `asterisk`.

1.4 uses `\protected` rather than robust commands. And implements the support for the new default of not using `\everymath`.

The first two `\newcommand*` are because the commands were previously defined unconditionally anyhow, in a way making them no-op's without option `asterisk`. Finally using `\let` here with `\empty` with the idea of less impact on TEX internals, but losing LATEX check.

```
\let\MTnormalasterisk\empty
\let\MTactiveasterisk\empty
\ifmst@asterisk\mst@infoline{asterisk: \string\ast\space and *}
  \ifmst@symbolmisc
    \protected\def\mst@bin@ast{\mathbin{\mathchoice
           {\raisebox{-.1\height}{\the\textfont\symmtpsymbol\char42}}%
           {\raisebox{-.1\height}{\the\textfont\symmtpsymbol\char42}}%
           {\raisebox{-.1\height}{\the\scriptfont\symmtpsymbol\char42}}%
           {\raisebox{-.1\height}{\the\scriptscriptfont\symmtpsymbol\char42}}}%
                     }%
  \else
    \protected\def\mst@bin@ast{\mathbin{\mathchoice
           {\raisebox{-\mst@lowerast}{\the\textfont\symmtoperatorfont\char42}}%
           {\raisebox{-\mst@lowerast}{\the\textfont\symmtoperatorfont\char42}}%
           {\raisebox{-\mst@lowerast}{\the\scriptfont\symmtoperatorfont\char42}}%
           {\raisebox{-\mst@lowerast}{\the\scriptscriptfont\symmtoperatorfont\char42}}}%
                     }%
  \fi
  \protected\def\mst@varfam@ast{\ifnum\fam=\m@ne
                      \mst@bin@ast
                      \else
                      \mathbin{\mathchoice
           {\raisebox{-\mst@lowerast}{\the\textfont\fam\char42}}%
           {\raisebox{-\mst@lowerast}{\the\textfont\fam\char42}}%
           {\raisebox{-\mst@lowerast}{\the\scriptfont\fam\char42}}%
           {\raisebox{-\mst@lowerast}{\the\scriptscriptfont\fam\char42}}}%
                      \fi}%
  \let\mst@ast\mst@bin@ast
  \newcommand*\MTlowerast[1]{\def\mst@lowerast{#1}}
  \MTlowerast{.3\height}
```

Arguably **mathastext** should have used the “hard” non-letters affiliation here. Probably too late to change this in 2024... and costly in documentation time.

```
  \mst@do@easynonletters\expandafter{%
      \the\mst@do@easynonletters\let\mst@ast\mst@varfam@ast
  }%
  \ifmst@everymath
     \def\mst@@doasterisk  {\let\ast\mst@ast\mst@mathactivate*{}\mst@ast}%
     \def\MTnormalasterisk {\let\mst@doasterisk\relax}
     \def\MTactiveasterisk {\let\mst@doasterisk\mst@@doasterisk}
     \MTactiveasterisk
     \AtBeginDocument{%
       \everymath\expandafter
              {\the\everymath   \mst@doasterisk \MTnormalasterisk }%
       \everydisplay\expandafter
              {\the\everydisplay\mst@doasterisk \MTnormalasterisk }%
     }%
  \else
     \def\MTnormalasterisk{\AtBeginDocument{\MTnormalasterisk}}
     \def\MTactiveasterisk{\AtBeginDocument{\MTactiveasterisk}}
```

For legacy reasons the responsiveness to math alphabets is made part of the handling of “easy” non letters (probably because it is on per default), and this causes me problems of internal logic and even more annoyingly of documentation. I am leaving this standing because it would be too much of a pain at this stage to document a change and it was already quite annoying to better document actual situation.

```
     \mst@undo@easynonletters\expandafter{%
          \the\mst@do@easynonletters\let\mst@ast\mst@bin@ast
     }%
```

For legacy reasons the action of `\MTactiveasterisk` is not testing if in `subdued` mode.

MEMO: if `subdued` there is `\MTeverymathoff` added to `\begin{document}` near end of package and it will do `\MTnormalasterisk`.

```
     \AtBeginDocument{%
         \let\mst@orig@abd@ast\ast
         \edef\MTnormalasterisk{\noexpand\mst@mathdeactivate*{\the\mst@mathcodenum`\*}%
                                \let\noexpand\ast\noexpand\mst@orig@abd@ast}%
         \def\MTactiveasterisk{\def\ast{\mst@ast}%
                               \mst@mathactivate*{}\mst@ast
                              }%
         \MTactiveasterisk
     }
  \fi
\fi
```

(2011) I renounced to try to do things with all the various dots, they are defined in many different ways, and there is the amsmath also. Dealing with this issue would mean a lot a time for a minuscule result. Better to leave the user use the mathdots package and accept that we can not avoid the default fonts in that case. So here I just treat `.` (in the hope to really lessen by 1 the number of fonts embedded at the end in the PDF).

[(Dec. 2012) should I reexamine these definitive sounding remarks?]

`ncccomma`
`decimalcomma`

`1.3y` of `2022/11/03` adds support for `ncccomma` option.

Some non-obvious hack is needed for compatibility with our home-made mechanism of non-letters obeying math alphabet commands. Alternative would have been to not load at all ncccomma (or since `1.3zb` decimalcomma) and provide the functionality purely by our own means; because here in order to support `\MTnonlettersobeymathxx` we are almost contrived to override quasi entirely the contents of these tiny packages.

`1.3zb` adds support for the `decimalcomma` option. This was mandatory to keep a compatibility layer with frenchmath after its `2.7` release of `2023/12/23`.

Hesitation if I should also make it shadow the `ncccomma` option if both are used at same time, or let the older option have priority. Well, let’s give priority to the new one so that one can do `\PassOptionsToPackage` and recycle old documents compiled via `\input` to tell them to use the new option.

Much ado about these tiny packages!

```
\ifmst@nopunct\else\mst@infoline{punctuation\string: \string, \string. \string:
    \string; and \string\colon}
  \DeclareMathSymbol{,}{\mathpunct}{\mst@font@tbu}{"2C}
  \DeclareMathSymbol{\mst@varfam@comma}{\mathalpha}{\mst@font@tbu}{"2C}
  \ifmst@decimalcomma
    \mst@infoline{loading package decimalomma for `smart comma\string'}
  \RequirePackage{decimalcomma}[2023/12/28]% 1.3 or later
```

Attention that the **BREAKING CHANGE** to `\AtBeginDocument` at October 2020 LaTeX release means that, *taking into account that* **mathastext** *already has employed some* `\AtBeginDocument` *prior to loading* decimalcomma, any code here will be executed **BEFORE** the `\AtBeginDocument` material from decimalcomma!

https://github.com/latex3/latex2e/issues/1226

So we definitely should not do here some ``\mathcode`\,="8000\relax`` in the `\AtBeginDocument`, and by the way I don't even recall why I had this line at some point which ended up causing me some much suffering and pain and lost sleep. It seems to have been a silly copy-paste from the `ncccomma` branch next, and that I started experimenting before having even re-read the code I copied pasted and whether it was needed.

decimalcomma is a rewrite of icomma and it loads the latter for which babel-french has a dectection mechanism, which as a result avoids the bad interactions with numprint plus its `autolanguage` option, which are mentioned below in the ncccomma branch. So we don't need here the workaround done below in the `ncccomma` branch. Notice though that in both cases, the 'intelligent' comma feature will be applied to the whole document, even inside those portions where the user has switched to another language such as English. This is to be expected here as nothing is done in a language specific manner, but if we wanted to do so, we might then be confronted with the babel issue mentioned next in the `ncccomma` branch.

```
  \let\mathcomma\relax
  \DeclareMathSymbol{\mathcomma}{\mathpunct}{\mst@font@tbu}{"2C}
```

Due to package decimalcomma internals, the hack here, which has to do with the "non letters obey math alphabets" optional **mathastext** feature, has to be done differently than the one we apply below for `ncccomma`. One can not really talk of a hack, as we basically have to redo the whole thing to insert an `\aftergroup` trick.

```
  \def\mst@sm@rtcomma{\begingroup\@tfor\@tempa:=0123456789%
                      \do{\expandafter\ifx\@tempa\@let@token
                                                \aftergroup\mathord
                                                \aftergroup\@gobble
                                                \@break@tfor\fi}%
                      \endgroup\mathpunct\mathcomma}
  \mst@do@nonletters\expandafter{\the\mst@do@nonletters
    \let\mathcomma\mst@varfam@comma
    \let\sm@rtcomma\mst@sm@rtcomma
    }
  \ifmst@everymath
  \else
    \edef\mst@tmp{\mathchardef\mathcomma=\the\mathcode`\,\relax}%
    \mst@undo@nonletters\expandafter\expandafter\expandafter
                {\expandafter\mst@tmp\the\mst@undo@nonletters}%
  \fi
\else
```

Work around some bad interaction of ncccomma, numprint with `autolanguage` and babel-french. See

https://github.com/latex3/babel/issues/190

for background. Some hesitation whether I should use the `\noextrasfrench` to work around babel-french code influencing non-French sections in the document. Update: I think the last sentence means I was hesitating at time of 1.3y whether to insert some extra code inside the `\noextrasfrench`.

```
\ifmst@ncccomma
  \mst@infoline{loading package ncccomma for `smart comma\string'}
  \RequirePackage{ncccomma}%
  \AtBeginDocument{%
    \mathcode`\,="8000\relax
    \@ifpackageloaded{babel}{%
               \addto\noextrasfrench{\mathcode`\,="8000\relax}%
               \addto\extrasfrench{\mathcode`\,="8000\relax}%
               }{}%
  }
  \let\mathcomma\relax
  \DeclareMathSymbol{\mathcomma}{\mathpunct}{\mst@font@tbu}{"2C}
```

Complications for compatibility with the `\MTnonlettersobeymathxx` mechanism. No fix done here for usage by ncccomma of `\@tempb` with no restoration of its meaning.

```
  \edef\mst@NCC@comma{\let\noexpand\@empty\mathpunct
                      \unexpanded\expandafter{\NCC@comma}%
                      \let\noexpand\@empty\noexpand\empty}
  \mst@do@nonletters\expandafter{\the\mst@do@nonletters
    \let\mathcomma\mst@varfam@comma
    \let\NCC@comma\mst@NCC@comma
    }
  \ifmst@everymath
  \else
```

Attention that ncccomma (contrarily to icomma loaded by decimalcomma has made the comma math active already when we loaded it. So we don't use `\the\mathcode`\,\relax` here.

```
    \edef\mst@tmp{\mathchardef\mathcomma=\the\mathcomma\relax}%
    \mst@undo@nonletters\expandafter\expandafter\expandafter
                {\expandafter\mst@tmp\the\mst@undo@nonletters}%
  \fi
\else
```

Neither option `ncccomma` nor `decimalcomma`. The 1.4 non-use of `\everymath` and consequences is accounted for automatically by `\mst@addtodo@nonletters`.

```
  \expandafter\mst@addtodo@nonletters\string,\mathpunct\mst@varfam@comma
\fi\fi
```

math dot.

```
\DeclareMathSymbol{.}{\mathord}{\mst@font@tbu}{"2E}
\DeclareMathSymbol{\mst@varfam@dot}{\mathalpha}{\mst@font@tbu}{"2E}
\mst@addtodo@easynonletters\.\mst@varfam@dot
```

math colon.

```
\DeclareMathSymbol{:}{\mathrel}{\mst@font@tbu}{"3A}
\DeclareMathSymbol{\mst@varfam@colon}{\mathalpha}{\mst@font@tbu}{"3A}
\expandafter\mst@addtodo@nonletters\string:\mathrel\mst@varfam@colon
\@ifpackageloaded{amsmath}
```

`\colon` gets defined in amsmath in terms of : with some enlarged explicit spacing. No need to intervene.

```
  {}
```

No amsmath, use standard punctuation spacing.

The reason for `\et\colon\undefined` is if some package has redefined `\colon` which then can not be used in `\DeclareMathSymbol` anymore (we shamelessly overwrite...)

```
  {%
    \let\colon\undefined
    \DeclareMathSymbol{\colon}{\mathpunct}{\mst@font@tbu}{"3A}
    \let\mst@colon\colon
```

`1.3v` uses `\protected` for the (optional) `\colon` redefinition.

```
    \mst@do@nonletters\expandafter{%
        \the\mst@do@nonletters
        \protected\def\colon{\mathpunct{\mst@varfam@colon}}%
    }%
```

`1.4` needs extras.

```
    \ifmst@everymath
    \else
        \mst@undo@nonletters\expandafter{%
            \the\mst@undo@nonletters
            \let\colon\mst@colon
         }%
    \fi
  }
```

Semi-colon. `1.3y` adds `binarysemicolon` option.

```
  \DeclareMathSymbol{\mst@varfam@pointvirgule}{\mathalpha}{\mst@font@tbu}{"3B}
  \ifmst@binarysemicolon
    \mst@infoline{semi-colon set to be of type \string\mathbin}
    \DeclareMathSymbol{;}{\mathbin}{\mst@font@tbu}{"3B}
    \expandafter\mst@addtodo@nonletters\string;\mathbin\mst@varfam@pointvirgule
  \else
    \DeclareMathSymbol{;}{\mathpunct}{\mst@font@tbu}{"3B}
    \expandafter\mst@addtodo@nonletters\string;\mathpunct\mst@varfam@pointvirgule
  \fi
\fi
```

`\relbar` Due to the way = and - are used by LaTeX in arrows, we will have to redefine `\Relbar` and `\relbar` in order for them to preserve their original meanings.

`1.15d`: Oct 13, 2012. Belated amendment of the code to be compatible with Unicode engines in case someone changed the mathcode of -. However, for the time being I can do it in an easy way only for XeTeX, not for LuaLaTeX. Also I do my modifications to `\relbar` in a manner testing for the presence of amsmath.

`1.3v` `2019/09/19`: LaTeX of `2019-10-01` defines `\leftarrowfill` and `\rightarrowfill` as robust macros, so we do the same.

I need to put amsmath under surveillance to check if it decides to robustify `\relbar` at some point, now that the LaTeX team has taken over maintenance.

2019/09/16 Use `\protected` for `\right|leftarrowfill` in the non `\DeclareRobustCommand` branch?

`1.4c` 2024/10/20: I didn't actually put amsmath under surveillance because I had other things to do with my life. But it appears indeed that `\relbar` (and `\Relbar`) are now robust also if amsmath is loaded, so I follow suit here.

MEMO: the minus character used in `\relbar` has its mathcode from the **`mathastext`** loading time. This is not the same as the minus in subdued mode which fetches its mathcode only at begin document. But anyhow the **`mathastext`** documentation insists the package should be loaded last among packages modifying math mode rendering.

```
\ifmst@nominus
\else
  \ifmst@XeOrLua
    \mst@Umathcharnumdef\mst@minus@sign=\mst@Umathcodenum`\-\relax
```

I used this prior to the new `\luatexUmathcodenum`, as available since TL2013:

```
\mathchardef\mst@minus@sign=8704\relax % "2200
```

```
  \else
    \mathchardef\mst@minus@sign=\mathcode`\-\relax
  \fi
  \@ifpackageloaded{amsmath}
    {\DeclareRobustCommand\relbar{\mathrel{\mathpalette\mathsm@sh\mst@minus@sign}}}
    {\DeclareRobustCommand\relbar{\mathrel{\smash\mst@minus@sign}}}
  \DeclareRobustCommand\rightarrowfill{$\m@th\mathord{\relbar}\mkern-7mu%
  \cleaders\hbox{$\mkern-2mu\relbar\mkern-2mu$}\hfill
  \mkern-7mu\mathord\rightarrow$}
  \DeclareRobustCommand\leftarrowfill{$\m@th\mathord\leftarrow\mkern-7mu%
  \cleaders\hbox{$\mkern-2mu\relbar\mkern-2mu$}\hfill
  \mkern-7mu\mathord{\relbar}$}
\fi
```

`endash` `1.1` 2011/01/29: Producing this next piece of code was not a piece of cake for a novice like myself!
`1.11` 2011/02/05: Compatibility with Unicode (via use of fontspec encodings EU1 and EU2)
`1.12` 2011/02/07: Improved dealing of Unicode possibility.
`1.14b` 2011/04/02: Corrected some very irresponsible bug in the Unicode part which caused a problem when 10 or more math families have been allocated.
`1.15` 2012/09/24: Added AtBeginDocument to circumvent some amsmath problem with unicode engines.

`1.3l` 2016/01/29: anticipating TL2016 fontspec's switch to `TU`.

`1.3t` 2018/08/22: fix to very ancient (2012/12/20) bug with `\DeclareMathSymbol` lacking last argument if encoding not T1, OT1 or LY1 when setting up math mode to use the en-dash character as minus sign (PDFTEX engine).

`\mst@subduedminus`
`\mst@nonsubduedminus`
`1.3t` Further, new macros `\mst@subduedminus` and `\mst@nonsubduedminus`, for the good functioning of the subdued option also in case of presence of fontspec. This is the only character for which subdued option works (now) by setting the mathcode on each math version change. Indeed, a typical issue is when the Unicode EN DASH or MINUS is used, but the actual font in subdued normal math version is originally in OT1 or T1 encoding. The only reasonable way to address this is by actually modifying the assigned mathcode at each version change. This means also that `\MTversion` and not `\mathversion` must be used for good functioning.

`1.3u` improves the handling of the minus sign by letting it be compatible with math versions (and not only follow the subdued mechanism) having varying font encodings, even possibly classic 8bit font encoding mixed with TU encoding for Unicode engines. For this it is needed to work around a feature (read: bug) of XeTeX/LuaLaTeX, here is original comment:

> afaict it is impossible to use straightforwardly in extended mathcode assignments a control sequence as created by `\Umathchardef`. This is counter-intuitive and breaks expectations.

A `1.3u` mechanism used a `\mst@UmathchardefWorkAround@i` (a bug which showed under option `noendash`, hence also `symboldelimiters`, with Unicode engines was fixed at `1.3w`). Unfortunately at `1.4c` we still have to use it because a core deficiency with math primitives in XeTeX/LuaTeX is still there (breakage of natural expectations based on TeX original syntax paradigm; search for words such as "painful" in the text next or prior to this location). The macro was renamed `\mst@ParseUmathchardef`.

The next macros are used by `\Mathastext` and `\MTDeclareVersion` to define `\mathchar` or `\Umathchar` control sequences which are named `\mst@minus@mv`⟨*mathversion*⟩ and `\mst@varfam@minus@mv`⟨*mathversion*⟩ which are math version dependent. The `\mst@minus` was added to complement `\mst@varfam@minus` at `1.4c`.

As far as the author can tell now (at `1.4c`) this naming scheme is due to the need to recover the `\textendash` or `\textemdash` slots in the math version dependent text font ("legacy TeX", i.e. non Unicode) encoding, and each math version can have its own font encoding.

Sadly, due to blatant deficiencies of the Unicode engines math primitives, it is only via a detour that we can fetch the extended math code defined by a `\Umathchardef` and we have to know it was not via a `\mathchardef`. Real pain.

The `\mst@subduedminus` is defined only at begin document and fetches there what will be the subdued mathcode of `-`.

`1.4d` avoids here low-level TeX error when attempting to use the en-dash slot in a TeX-legacy non-text encoding such as `OML` (LaTeX has defines no `\textendash` in such encodings). The error was present since changes at `1.3u` which used code adapting to the required font encoding rather than test for it being `T1` or `LY1` etc...

```
\let\mst@subduedminus\empty
\let\mst@nonsubduedminus\empty
\def\mst@dothe@endashstuff#1#2#3#4{%
  \edef\mst@tmp@enc{#3}%
  \if1\mst@OneifUniEnc
    \mst@Umathchardef#1=2 \symmtoperatorfont "\mst@unicodeminus\relax
    \mst@Umathchardef#2=7 \symmtoperatorfont "\mst@unicodeminus\relax
  \else
    \@ifundefined{\mst@tmp@enc\string\textendash}%
        {%
          \PackageWarningNoLine{mathastext}
          {Impossible to let the \string- use an EN-DASH in a math\MessageBreak
           version with \mst@tmp@enc\space encoding, as there is none there.\MessageBreak
           Reverting to default in math version #4}%
          \let#1\relax\let#2\relax
        }%
        {%
          \DeclareMathSymbol{#1}{\mathbin}{mtoperatorfont}
```

```
                            {\csname\mst@tmp@enc\string\textendash\endcsname}%
          \DeclareMathSymbol{#2}{\mathalpha}{mtoperatorfont}
                            {\csname\mst@tmp@enc\string\textendash\endcsname}%
        }%
  \fi
}%
\def\mst@dothe@emdashstuff#1#2#3#4{%
  \edef\mst@tmp@enc{#3}%
  \if1\mst@OneifUniEnc
    \mst@Umathchardef#1=2 \symmtoperatorfont "2014\relax
    \mst@Umathchardef#2=7 \symmtoperatorfont "2014\relax
  \else
    \@ifundefined{\mst@tmp@enc\string\textendash}%
        {%
          \PackageWarningNoLine{mathastext}
          {Impossible to let the \string- use an EM-DASH in a math\MessageBreak
           version with \mst@tmp@enc\space encoding, as there is none there.\MessageBreak
           Reverting to default in math version #4}%
          \let#1\relax\let#2\relax
        }%
        {%
          \DeclareMathSymbol{#1}{\mathbin}{mtoperatorfont}
                            {\csname\mst@tmp@enc\string\textemdash\endcsname}%
          \DeclareMathSymbol{#2}{\mathalpha}{mtoperatorfont}
                            {\csname\mst@tmp@enc\string\textemdash\endcsname}%
        }%
  \fi
}%
\def\mst@dothe@hyphenstuff#1#2{%
  \DeclareMathSymbol{#1}{\mathbin}{\mst@font@tbu}{"2D}%
  \DeclareMathSymbol{#2}{\mathalpha}{\mst@font@tbu}{"2D}%
}%
\def\mst@varfam@minus{\csname mst@varfam@minus@mv\math@version\endcsname}%
\def\mst@minus{\csname mst@minus@mv\math@version\endcsname}%
```

`1.4c` fixes a `1.4` bug: `\MTnonlettersdonotobeymathxx` (i.e. not with `everymath` option) restored the minus to have the mathcode in place at package loading time, which was wrong.

The correct mathcode, which is possibly math version dependent, is stored in some `\mathchar` or `\Umathchar` and can be set via `\mst@nonsubduedminus` execution. We can not use `\mst@addtodo@nonletters` with `-` here (except under `everymath`). We need special coding to use `\mst@nonsubduedminus` or `\mst@subduedminus`.

It is very painful troubles with XeTeX/LuaTeX once we have a `\Umathchardef` to re-assign its extended mathcode to a character token; we can't have syntax common with non-Unicode engine or cases where the control sequence was actually defined via `\mathchardef`. Hence there are some unnatural complications. What a pain.

```
\ifmst@nominus\else
  \ifmst@everymath
    \expandafter\mst@addtodo@nonletters\string-\mathbin\mst@varfam@minus
  \else
```

I have absolutely no idea why I was paranoid about `-` catcode (as for other characters) but I feel even at `1.4c` I have to maintain the tradition.

`1.4d` takes extra precautions relative to **`mathastext`** math versions associated with a font encoding not having an EN-DASH slot.

```
  \def\mst@tmp#1{%
    \mst@do@nonletters\expandafter{\the\mst@do@nonletters
      \@ifundefined{mst@minus@mv\math@version}
          {\mst@nonmathmathactiveminus}
          {\mst@mathactivate#1\mathbin\mst@varfam@minus}%
      }%
```

At `1.4c`, `\mst@mathdeactivate` adds a check to whether its first argument is `-` and then does not do itself the mathcode things (which are the matter of either `\mst@nonsubduedminus` or `\mst@subduedminus`) but only the other stuffs relative to restoration of a previous meaning for the catcode active character.

```
    \mst@undo@nonletters\expandafter{\the\mst@undo@nonletters
       \mst@mathdeactivate#1{}%
       \mst@nonmathactiveminus}%
  }\expandafter\mst@tmp\string-%
```

We have to go through these things because `\MTnonlettersdonotobeymathxx` which executes `\mst@undo@nonletters` is done via `\MTeverymathoff` which is part of the handling of `subdued` normal and bold math.

```
  \def\mst@nonmathactiveminus{%
      \ifmst@subdued
       \ifx\math@version\mst@normalversionname\mst@subduedminus\else
       \ifx\math@version\mst@boldversionname  \mst@subduedminus\else
       \mst@nonsubduedminus\fi\fi
      \else
       \mst@nonsubduedminus
      \fi}%
\fi
```

I have no time to test extensively but it seems in 2024 that we still have this blatant deficiency of LuaTEX/XƎTEX with regards to `\Umathchardef`: we can't use `\Umathcode` or `\Umathcodenum` with them. This is obviously a strong disappointment upon which I lamented already here in 2019 and much earlier certainly as it has plagued my life eversince I added support for Unicode engines to **`mathastext`**.

ATTENTION: Defining `\mst@tmp@enc` is mandatory for `\mst@OneifUniEnc` to work correctly...

If neither `endash` nor `emdash` is true we will have to redefine this macro next.

```
\def\mst@nonsubduedminus{%
  \edef\mst@tmp@enc{\csname mst@encoding@\math@version\endcsname}%
  \if1\mst@OneifUniEnc
   \mst@Umathcode`\-=\expandafter
                     \mst@ParseUmathchardef
                     \csname mst@minus@mv\math@version\endcsname
                     \relax
  \else
   \@ifundefined{mst@minus@mv\mst@tmp@enc}
```

```
        {\mst@subduedminus}
        {\mathcode`\-=\mst@minus}%
    \fi
  }%
  \def\mst@ParseUmathchardef{\expandafter\mst@ParseUmathchardef@i\meaning}%
  \def\mst@ParseUmathchardef@i#1"{"}%
  \ifmst@endash\else\ifmst@emdash\else
    \def\mst@nonsubduedminus{\mathcode`\-=\mst@minus}%
  \fi\fi
\fi
```

`\mst@hbar@mvnormal`
`\mst@ltbar@mvnormal`

2011/01/31, 1.1 I decide to settle the question of the `\hbar`. The LaTeX definition is `\def\hbar{{\mathchar'26\mkern-9muh}}` and its advantage is that `h` is in the correct font. But of course not the macron character (`\=`, `\bar`). And anyway `amsfonts` uses a `\DeclareMathSymbol`. Also there is the kern whose length depends on `cmsy` (18mu=1em and em taken from info in `cmsy`).

I will need an `rlap` adapted to math mode, and this is provided by code from Alexander R. Perlis in his TugBoat article 22 (2001), 350–352, which I found by googling `rlap`. (as an aside, I am only now (April 2, 2011) aware that the package `mathtools` provides the `mathrlap` etc...)

1.3l 2016/01/29: anticipating TL2016 fontspec's switch to `TU`.

1.3u 2019/08/20: encoding (8bits) agnostic construct for hbar, using same method as for `mathaccents` option. I should add some way to adjust the vertical positioning.

On this occasion I replace `h` by `\mst@h` because the mechanism for before and after skips does not interact well with the rlap construct.

1.3v 2019/09/19 adapts to maintain the robustness of `\hbar` which now applies with LaTeX 2019-10-01.

1.3w works around https://github.com/latex3/latex2e/issues/216 via `\mst@DeclareMathAccent`. The upstream bug affected the definition of `\mst@ltbar@mvnormal` and broke usage of `\Mathastext` in preamble.

1.3w also fixes oversight that `\hbar` may have been redefined via `\DeclareMathSymbol` by some package (e.g. `amsfonts`) and with LaTeX 2019-10-01 this means `\hbar<space>` is now undefined. Modifying it changed nothing to `\hbar` behavior in such circumstances. Finally we opt for a `\protected` `\hbar` and choose to ignore completely if there is a `\hbar<space>` or not. To avoid extra steps we do not undefine it if it exists, because we would need to restore it in subdued math versions.

```
\let\mst@subduedhbar\@empty
\let\mst@nonsubduedhbar\@empty
\ifmst@nohbar\else
  \def\mst@subduedhbar{\let\hbar\mst@original@hbar}%
  \def\mst@nonsubduedhbar{\expandafter
       \let\expandafter\hbar\csname mst@hbar@mv\math@version\endcsname
       \ifx\hbar\relax\let\hbar\mst@original@hbar\fi}%
\fi
\def\mst@mathrlap{\mathpalette\mst@mathrlapinternal}
\def\mst@mathrlapinternal#1#2{\rlap{$\mathsurround=0pt#1{#2}$}}
```

ATTENTION! The definition of `\mst@tmp@enc` is mandatory for `\mst@OneifUniEnc` to work correctly...

The 1.3u method which was backported from the one for math accents causes a build crash when attempting to create a **mathastext** math version with `OML` encoding, as there is no `\OML\=` control sequence. Fixed at 1.4d, see also tne definition of `\mst@nonsubduedhbar`.

```
\def\mst@dothe@hbarstuff#1#2#3#4{%
  \edef\mst@tmp@enc{#3}%
  \if1\mst@OneifUniEnc
    \mst@Umathchardef#1="7 \symmtletterfont "0127 \relax %% or 210F?
  \else
    \@ifundefined{\mst@tmp@enc\string\=}%
        {%
          \PackageWarningNoLine{mathastext}
          {Impossible to define the \string\hbar\space using a font with\MessageBreak
           \mst@tmp@enc\space encoding, as there is no \string\= accent.\MessageBreak
           Reverting to default \string\hbar\space in math version #4}%
          \let#1\relax
        }%
        {%
         \begingroup
         \def\@text@composite##1\@text@composite##2{##2}%
         \let\add@accent\@firstoftwo
         \mst@DeclareMathAccent{#2}{\mathalpha}{mtletterfont}%
                   {\csname\mst@tmp@enc\string\=\endcsname{}}%
         \endgroup
         \protected\def#1{\mst@mathrlap{#2{\ }}\MTmathcharletterh}%
        }%
  \fi
}%
```

1.15d: Oct 13, 2012. The `\mathcode` thing with = is (belatedly, sorry!) made Unicode compatible.

`+,=,\Relbar`

```
\ifmst@noplus\else\mst@infoline{\string+ and \string=}
\DeclareMathSymbol{+}{\mathbin}{\mst@font@tbu}{"2B}
\DeclareMathSymbol{\mst@varfam@plus}{\mathalpha}{\mst@font@tbu}{"2B}
\expandafter\mst@addtodo@nonletters\string+\mathbin\mst@varfam@plus
\fi
\ifmst@noequal\else
\ifmst@XeOrLua
    \mst@Umathcharnumdef\mst@equal@sign=\mst@Umathcodenum`\=\relax
\else
    \mathchardef\mst@equal@sign=\mathcode`\=\relax
\fi
```

1.4c makes `\Relbar` robust also with amsmath (the original wasn't robust in the past). We can now use same definition whether or not amsmath is loaded.

MEMO: the character “equal” used in `\Relbar` has its mathcode frozen at **mathastext** loading time.

```
\DeclareRobustCommand\Relbar{\mathrel\mst@equal@sign}
\DeclareMathSymbol{=}{\mathrel}{\mst@font@tbu}{"3D}
```

```
\DeclareMathSymbol{\mst@varfam@equal}{\mathalpha}{\mst@font@tbu}{"3D}
```

`\nfss@catcodes`

2012/12/18: Activating = (only in math mode actually) seems very bad but surprisingly works well. However I had a problem with `eu2lmtt.fd` which should not be loaded with an active =. 2012/12/25: Since then I had switched to only math activation. And in fact the problematic = from `eu2lmtt.fd` end up in `\csname`...`\endcsname` and I have learnt since that TeX does not look at the `mathcode` inside a `\csname`...`\endcsname`. Example:

```
% \mathcode`x="8000
% \begingroup
% \catcode`x=\active
% \global\everymath{\defx{Hello}}
% \endgroup
% \def\foox{World!}
% $x \csname foox\endcsname$
%
```

We need nevertheless to inactivate the =, for the following reason. Imagine someone did `\catcode`==\active\def={\string=}`, or another definition which would not lead to a tragedy in a `\csname`...`\endcsname`. Then the = is active and the re-definition done by `mathastext` will not be compatible with loading `eu2lmtt.fd` (for the first time) from math mode, as this re-definition can not be expanded inside a `\csname`...`\endcsname`.

2012/12/28: to be on the safe side, I add also `;` and `+` and do it without discriminating between engines

```
\mst@infoline{adding \string= \string; and \string+ to \string\nfss@catcodes}
\g@addto@macro\nfss@catcodes{%
  \@makeother\=%
  \@makeother\;%
  \@makeother\+%
}
\expandafter\mst@addtodo@nonletters\string=\mathrel\mst@varfam@equal
\fi
```

`noparenthesis`
`(,),[,],/`

`\lbrack` and `\rbrack` are defined in `latex.ltx` by `\def\lbrack{[}\def\rbrack{]}` so this fits well with what we do here. `\lparen` and `\rparen` are similarly defined in `mathtools`. On the other hand in `latex.ltx` with `\{` and `\}` are defined (in math mode) in terms of the control sequences `\lbrace` and `\rbrace`. Such control sequences can not be simultaneously math symbols and math delimiters, thus, this complicates things for the mathastextification.

```
\ifmst@noparen\else
\mst@infoline{parentheses \string( \string) \string[ \string] and slash \string/}
\ifmst@nosmalldelims
    \DeclareMathSymbol{(}{\mathopen}{\mst@font@tbu}{"28}
    \DeclareMathSymbol{)}{\mathclose}{\mst@font@tbu}{"29}
    \DeclareMathSymbol{[}{\mathopen} {\mst@font@tbu}{"5B}
    \DeclareMathSymbol{]}{\mathclose}{\mst@font@tbu}{"5D}
    \DeclareMathSymbol{/}{\mathord}{\mst@font@tbu}{"2F}
\else
    \DeclareMathDelimiter{(}{\mathopen}{\mst@font@tbu}{"28}{largesymbols}{"00}
    \DeclareMathDelimiter{)}{\mathclose}{\mst@font@tbu}{"29}{largesymbols}{"01}
    \DeclareMathDelimiter{[}{\mathopen} {\mst@font@tbu}{"5B}{largesymbols}{"02}
    \DeclareMathDelimiter{]}{\mathclose}{\mst@font@tbu}{"5D}{largesymbols}{"03}
    \DeclareMathDelimiter{/}{\mathord}{\mst@font@tbu}{"2F}{largesymbols}{"0E}
\fi
\DeclareMathSymbol{\mst@varfam@lparen}{\mathalpha}{\mst@font@tbu}{40}
\DeclareMathSymbol{\mst@varfam@rparen}{\mathalpha}{\mst@font@tbu}{41}
\DeclareMathSymbol{\mst@varfam@lbrack}{\mathalpha}{\mst@font@tbu}{"5B}
\DeclareMathSymbol{\mst@varfam@rbrack}{\mathalpha}{\mst@font@tbu}{"5D}
\DeclareMathSymbol{\mst@varfam@slash}{\mathalpha}{\mst@font@tbu}{"2F}
\expandafter\mst@addtodo@nonletters\string(\mathopen\mst@varfam@lparen
\expandafter\mst@addtodo@nonletters\string)\mathclose\mst@varfam@rparen
\expandafter\mst@addtodo@nonletters\string[\mathopen\mst@varfam@lbrack
\expandafter\mst@addtodo@nonletters\string]\mathclose\mst@varfam@rbrack
\mst@addtodo@easynonletters\/\mst@varfam@slash
\fi
```

`alldelims`
`<,>,\`
`\setminus`
`\backslash`

```
\ifmst@alldelims\mst@infoline{alldelims: \string< \string>
  \string\backslash\space\string\setminus\space\string|
  \string\vert\space\string\mid\space\string\{\space \string\}}
\ifmst@nosmalldelims
```

Dec 18, 2012. We then want `\let\backslash\mst@varfam@backslash` to do nothing when the `\backslash` is used as a delimiter. So here the original definition from `latex.ltx` is copied, generally speaking when people use other math symbol fonts they do respect the encoding of the CM symbols and largesymbols, so this is 90% safe. But in truth I should extract from the meaning of `\backslash` the `delcode`.

At `1.4` I am a bit perplexed at the `\mathalpha` here. Adding a a `\mst@mathord@backslash` for matters of option `noeverymath`.

```
    \DeclareMathDelimiter{\mst@varfam@backslash}
        {\mathalpha}{symbols}{"6E}{largesymbols}{"0F}
    \let\mst@mathord@backslash\backslash
\else
    \DeclareMathDelimiter{<}{\mathopen}{\mst@font@tbu}{"3C}{largesymbols}{"0A}
    \DeclareMathDelimiter{>}{\mathclose}{\mst@font@tbu}{"3E}{largesymbols}{"0B}
```

There is no backslash in the Symbol font hence `mtoperatorfont` here.

```
    \DeclareMathDelimiter{\mst@mathord@backslash}
        {\mathord}{mtoperatorfont}{"5C}{largesymbols}{"0F}
    \let\backslash\mst@mathord@backslash
    \DeclareMathDelimiter{\mst@varfam@backslash}
        {\mathalpha}{mtoperatorfont}{"5C}{largesymbols}{"0F}
\fi
\DeclareMathSymbol{<}{\mathrel}{\mst@font@tbu}{"3C}
\DeclareMathSymbol{>}{\mathrel}{\mst@font@tbu}{"3E}
\DeclareMathSymbol{\mst@varfam@less}{\mathalpha}{\mst@font@tbu}{"3C}
\DeclareMathSymbol{\mst@varfam@more}{\mathalpha}{\mst@font@tbu}{"3E}
\expandafter\mst@addtodo@nonletters\string<\mathrel\mst@varfam@less
\expandafter\mst@addtodo@nonletters\string>\mathrel\mst@varfam@more
\mst@do@easynonletters\expandafter{\the\mst@do@easynonletters
            \let\backslash\mst@varfam@backslash}
```

Extras for `1.4` are needed:

```
\ifmst@everymath
\else
  \mst@undo@easynonletters\expandafter{\the\mst@undo@easynonletters
            \let\backslash\mst@mathord@backslash}
\fi
\DeclareMathSymbol{\setminus}{\mathbin}{mtoperatorfont}{"5C}
\let\mst@setminus\setminus
\DeclareMathSymbol{\mst@varfam@setminus}{\mathalpha}{mtoperatorfont}{"5C}
```

`1.3v` adds a `\protected` here for `\setminus`.

```
\mst@do@nonletters\expandafter{%
     \the\mst@do@nonletters
     \protected\def\setminus{\mathbin{\mst@varfam@setminus}}%
}
```

Extras for `1.4` are needed:

```
\ifmst@everymath
\else
    \mst@undo@nonletters\expandafter{%
         \the\mst@undo@nonletters
         \let\setminus\mst@setminus
    }
\fi
```

`\models` `1.15d`: 13 oct 2012. Before modifying | we must preserve `\models`.

```
\ifmst@XeOrLua
  \mst@Umathcharnumdef\mst@vert@bar=\mst@Umathcodenum`\|\relax
\else
  \mathchardef\mst@vert@bar=\mathcode`\|\relax
\fi
\DeclareRobustCommand\models{\mathrel{\mst@vert@bar}\joinrel\Relbar}
```

`|,\mid,\vert` (2011) I did not do anything then to try to emulate `\Vert` with the vertical bar from the text font... and now (2012) **mathastext** is not as radical as it used to be anyhow, so it is too late. Or not (2019)? maybe I *should* do something here...

`1.3v` 2019/09/19: I discover this rather radical legacy `\def\vert{|}`, which is done here once in the preamble, but I leave it unmodified apart from prefixing it with `\protected`. I also add a `\protected` for the definition of `\mid` (which applies only under `\MTnonlettersobeymathxx` regime).

```
\ifmst@nosmalldelims
    \DeclareMathSymbol{|}{\mathord}{\mst@font@tbu}{124}
\else
    \DeclareMathDelimiter{|}{\mathord}{\mst@font@tbu}{124}{largesymbols}{"0C}
\fi
\protected\def\vert{|}
\DeclareMathSymbol{\mst@varfam@vbar}{\mathalpha}{\mst@font@tbu}{124}
\mst@addtodo@easynonletters\|\mst@varfam@vbar
\let\mid\undefined
\DeclareMathSymbol{\mid}{\mathrel}{\mst@font@tbu}{124}
```

```
\let\mst@mid\mid
\mst@do@nonletters\expandafter{%
    \the\mst@do@nonletters
    \protected\def\mid{\mathrel\mst@varfam@vbar}%
}
```

Extras for 1.4 are needed:

```
\ifmst@everymath
\else
    \mst@undo@nonletters\expandafter{%
        \the\mst@undo@nonletters
        \let\mid\mst@mid
    }
\fi
```

\MTexplicitbracesobeymathxx
\MTexplicitbracesdonotobeymathxx

Braces. With version 1.2, `\{` and `\}` will not be acceptable as delimiters anymore if the redefinitions below in `\mst@dobraces` are enacted. But they will obey math alphabets. Improvements in 1.2a, to preserve robustness.

For 1.3 I make `\lbrace` and `\rbrace` undefined first, else problems may arise with some packages.

1.3e suppresses under option `nosmalldelims` the definitions of `\lbrace` and `\rbrace` as math symbols as this made `\left\lbrace` cause an error, it was a bug.

(obsolete) LaTeX2e defines `\{` and `\}` as robust commands since a long time (I don't know since when). The **mathastext** redefinition is done only if user has executed `\MTexplicitbracesobeymathxx`, and it is done only when entering math mode, but there could be some `\hbox` inside math, hence it has to be careful to be valid in text too.

1.3v maintains strict LaTeX2e robustness for `\{` and `\}`. This assumes no one fiddled with `\{` and `\}` proper (without space in the name).

LaTeX made `\{` and `\}` `\protected`, not robust, at its `2020-02-02` release, so the code used by **mathastext** under `\MTexplicitbracesobeymathxx` toggle remained without effect since as it configured a change to `\{<space>` and `\}<space>`. Fixed at 1.4.

```
  \ifmst@nosmalldelims
  \else
     \let\lbrace\undefined \let\rbrace\undefined
     \DeclareMathDelimiter{\lbrace}
        {\mathopen}{\mst@font@tbu}{123}{largesymbols}{"08}
     \DeclareMathDelimiter{\rbrace}
        {\mathclose}{\mst@font@tbu}{125}{largesymbols}{"09}
  \fi
\DeclareMathSymbol{\mst@varfam@lbrace}{\mathalpha}{\mst@font@tbu}{123}
\DeclareMathSymbol{\mst@varfam@rbrace}{\mathalpha}{\mst@font@tbu}{125}

\protected\def\mst@lbrace{\ifmmode\mathopen\mst@varfam@lbrace\else\textbraceleft\fi}
\protected\def\mst@rbrace{\ifmmode\mathclose\mst@varfam@rbrace\else\textbraceright\fi}
\mst@do@nonletters\expandafter{%
    \the\mst@do@nonletters
    \mst@dobraces{\let\{\mst@lbrace\let\}\mst@rbrace}%
}
\fi
\newcommand*{\MTexplicitbracesobeymathxx}{\let\mst@dobraces\@firstofone}
```

```
\newcommand*{\MTexplicitbracesdonotobeymathxx}{\let\mst@dobraces\@gobble}
```

Extras for `1.4` are needed.

MEMO: The coding for `\MTexplicitbracesobeymathxx` in this non `everymath` (default) branch is lazy, as it causes execution of full `\MTnonlettersobeymathxx`, but it logically maintains the pre-`1.4` situation, which activates the braces only as part of the global `\mst@do@nonletters`. Advantage also is that in `subdued` it is a no-op because `\MTnonlettersobeymathxx` is. But, if issued while in the `subdued` normal math version, if after switching to another math version user employs `\MTnonlettersobeymathxx`, then the effect of `\MTexplicitbracesobeymathxx` will be seen. This is the same behavior as under `everymath` option.

But contrarily to `everymath` situation, in a non-`subdued` math version, it is not needed to explicitly use `\MTnonlettersobeymathxx` prior to `\MTexplicitbracesobeymathxx`, the latter implies the former.

```
\ifmst@everymath
\else
    \renewcommand*\MTexplicitbracesobeymathxx{%
        \let\mst@dobraces\@firstofone
        \MTnonlettersobeymathxx
    }%
    \renewcommand*\MTexplicitbracesdonotobeymathxx{%
        \let\mst@dobraces\@gobble
        \protected\def\{{\ifmmode \lbrace \else \textbraceleft \fi}%
        \protected\def\}{\ifmmode \rbrace \else \textbraceright \fi}%
    }%
    \mst@undo@nonletters\expandafter{\the\mst@undo@nonletters
        \protected\def\{{\ifmmode \lbrace \else \textbraceleft \fi}%
        \protected\def\}{\ifmmode \rbrace \else \textbraceright \fi}%
    }%
\fi
\MTexplicitbracesdonotobeymathxx
```

`specials` `1.14b` 2011/04/02: the redefinitions of #, $, % and & were buggy (this showed up when 10 or more math families had been created).

`1.15f` 2012/10/23: the code, although working, was perhaps a bit insane and had definitions which could surprise other packages. For example, it did:

`\renewcommand{\%}{\ifmmode\mt@mmode@percent\else\char37\relax\fi}`

But it seems this provokes a problem with `microtype`. Perhaps the problem was that the command was not declared robust? For the dollar LaTeX itself does

`\DeclareRobustCommand{\$}{\ifmmode\mathdollar\else\textdollar\fi}`

So here I just modify `\mathdollar`. Then we have in `latex.ltx` the same definitions as in `plain.tex`: `` \chardef\%=`\% ``, `` \chardef\&=`\& ``, and `` \chardef\#=`\# ``. It turns out that we can just adjust the mathcodes of these characters and achieve exactly what is wanted for the corresponding one char control sequences. In math mode the control sequence will use the specified mathcode. So here it is *not* a redefinition of the control sequences, purely an adjustment of mathcodes.

`1.2d` 2013/01/01: previous versions imposed the variable family type. I hereby make it possible to de-activate this feature with the macro `\MTeasynonlettersdonotobeymathxx`. Besides, I have absolutely no idea why I had different looking code depending on the engine XeTeX, LuaTeX or default. Removed.

1.3c 2013/12/14: I have absolutely no idea why I removed the XeTeX and LuaTeX code at the time of 1.2d! the code for tex/pdftex engine could not accomodate more than 16 math families. Code for XeTeX and LuaTeX again added. (and since TL2013 no more problems with `\luatexUmathcode`.)

1.4 has done slight refactoring here to share more code related to math activation across the two branches.

```
\ifmst@nospecials
\else
  \mst@infoline{\string\#\space\string\mathdollar\space
                \string\%\space\string\&\space}
  \ifmst@XeOrLua
    \mst@Umathcode`\#=0 \symmtoperatorfont "23 \relax
    \mst@Umathchardef\mathdollar=0 \symmtoperatorfont "24 \relax
    \mst@Umathcode`\%=0 \symmtoperatorfont "25 \relax
    \mst@Umathcode`\&=0 \symmtoperatorfont "26 \relax
    \mst@Umathchardef\mst@varfam@mathhash      = 7 \symmtoperatorfont "23 \relax
    \mst@Umathchardef\mst@varfam@mathdollar    = 7 \symmtoperatorfont "24 \relax
    \mst@Umathchardef\mst@varfam@mathpercent   = 7 \symmtoperatorfont "25 \relax
    \mst@Umathchardef\mst@varfam@mathampersand = 7 \symmtoperatorfont "26 \relax
    \mst@addtodo@easynonletters@U\#\mst@varfam@mathhash
    \mst@addtodo@easynonletters@U\%\mst@varfam@mathpercent
    \mst@addtodo@easynonletters@U\&\mst@varfam@mathampersand
  \else
    \count@=\symmtoperatorfont
    \multiply\count@ by \@cclvi
    \advance\count@ by 35
        \mathcode`\#\count@
    \advance\count@ by \@ne
        \mathchardef\mathdollar\count@
    \advance\count@ by \@ne
        \mathcode`\%\count@
    \advance\count@ by \@ne
        \mathcode`\&\count@
    \count@=\symmtoperatorfont
    \multiply\count@ by \@cclvi
    \advance\count@ by 28707 % = "7023
        \mathchardef\mst@varfam@mathhash\count@
    \advance\count@ by \@ne
        \mathchardef\mst@varfam@mathdollar\count@
    \advance\count@ by \@ne
        \mathchardef\mst@varfam@mathpercent\count@
    \advance\count@ by \@ne
        \mathchardef\mst@varfam@mathampersand\count@
    \mst@addtodo@easynonletters\#\mst@varfam@mathhash
    \mst@addtodo@easynonletters\%\mst@varfam@mathpercent
    \mst@addtodo@easynonletters\&\mst@varfam@mathampersand
  \fi
```

It is possible to mathematically activate the dollar sign, and to use it in math mode as `\string$`... well, how many documents will have done that? But we do not modify the mathcode of the `$` character anyhow, so why should I add here a `\mst@OnlyIfNotMathActive`? There is no good reason for that.

```
  \mst@do@easynonletters\expandafter{\the\mst@do@easynonletters
    \let\mathdollar\mst@varfam@mathdollar}%
```

Extras for 1.4 are needed. Here too we do not worry about math active dollar character.

```
  \ifmst@everymath\else
      \edef\mst@tmp{\ifmst@XeOrLua\mst@Umathcharnumdef\else\mathchardef\fi
                    \mathdollar=\the\mathdollar\relax}%
      \mst@undo@easynonletters\expandafter\expandafter\expandafter
               {\expandafter\mst@tmp\the\mst@undo@easynonletters}%
  \fi
\fi
```

`symbolmisc` We construct (with some effort) some long arrows from the Symbol glyphs, of almost the same lengths as the standard ones. By the way, I always found the `\iff` to be too wide, but I follow here the default. Also, although there is a `\longmapsto` in standard LaTeX, if I am not mistaken, there is no `\longto`. So I define one here. I could not construct in the same manner `\Longrightarrow` etc… as the $=$ sign from Symbol does not combine easily with the logical arrows, well, I could have done some box manipulations, but well, life is finite.

`\prod`
`\sum`

1.13b: I correct the brutal re-definitions of `\prod` and `\sum` from the earlier versions of the package; most of the time the Symbol glyphs do appear to be too small in display mode. The new redefinitions do have some defects: `$\displaystyle\prod_1^2$` changes the position of limits but not the glyph itself, and `$$\textstyle\prod_1^2$$` change the limits but switches to the CM inline math glyph. So I tried
`\renewcommand{\prod}{\mathchoice{\mst@prod}{\prodpsy}{\prodpsy}{\prodpsy}}`
but this did not go well with subscripts and exponents.

October 2012: maybe I should re-examine what I did?

1.3c (2013/12/14) renames `\defaultprod` to `\MToriginalprod` and `\defaultsum` to `\MToriginalsum`.

1.3v hesitates about making robust here `\prod` and `\sum`. Finally I use `\protected` for them.

1.4 finally replaces all control sequences defines earlier via `\DeclareRobustCommand` to be defined using `\protected\def`. It also removes `\mst@prod` and `\mst@sum` which were aliases of `\MToriginalprod` and `\MToriginalsum`, and having both must have been some left-over of earlier versions in 2012 or 2013 (unchecked).

```
\ifmst@symbolmisc
\mst@infoline{symbolmisc: miscellaneous math symbols from Symbol font}
\let\MToriginalprod\prod
\DeclareMathSymbol{\prodpsy}{\mathop}{mtpsymbol}{213}
\protected\def\prod{\ifinner\prodpsy\else\MToriginalprod\fi}
\let\MToriginalsum\sum
\DeclareMathSymbol{\sumpsy}{\mathop}{mtpsymbol}{229}
\protected\def\sum{\ifinner\sumpsy\else\MToriginalsum\fi}

\DeclareMathSymbol{\mst@implies}{\mathrel}{mtpsymbol}{222}
\protected\def\implies{\;\mst@implies\;}
```

```
\DeclareMathSymbol{\mst@impliedby}{\mathrel}{mtpsymbol}{220}
\protected\def\impliedby{\;\mst@impliedby\;}
\protected\def\iff{\;\mst@impliedby\mathrel{\mkern-3mu}\mst@implies\;}
\DeclareMathSymbol{\mst@iff}{\mathrel}{mtpsymbol}{219}
\protected\def\shortiff{\;\mst@iff\;}
```

1.4 drops defining a `\mst@to` and employs `\to` in subsequent definitions

```
\let\to\relax
\DeclareMathSymbol{\to}{\mathrel}{mtpsymbol}{174}
\DeclareMathSymbol{\mst@trait}{\mathrel}{mtpsymbol}{190}
\protected\def\longto{\mkern2mu\mst@trait\mathrel{\mkern-10mu}\to}
\protected\def\mapsto{\mapstochar\mathrel{\mkern0.2mu}\to}
\protected\def\longmapsto{%
    \mapstochar\mathrel{\mkern2mu}\mst@trait\mathrel{\mkern-10mu}\to
}
\DeclareMathSymbol{\aleph}{\mathord}{mtpsymbol}{192}
\DeclareMathSymbol{\inftypsy}{\mathord}{mtpsymbol}{165}
\DeclareMathSymbol{\emptyset}{\mathord}{mtpsymbol}{198}
\let\varnothing\emptyset
\DeclareMathSymbol{\nabla}{\mathord}{mtpsymbol}{209}
\DeclareMathSymbol{\surd}{\mathop}{mtpsymbol}{214}
\let\angle\undefined
\DeclareMathSymbol{\angle}{\mathord}{mtpsymbol}{208}
\DeclareMathSymbol{\forall}{\mathord}{mtpsymbol}{34}
\DeclareMathSymbol{\exists}{\mathord}{mtpsymbol}{36}
\DeclareMathSymbol{\neg}{\mathord}{mtpsymbol}{216}
\DeclareMathSymbol{\clubsuit}{\mathord}{mtpsymbol}{167}
\DeclareMathSymbol{\diamondsuit}{\mathord}{mtpsymbol}{168}
\DeclareMathSymbol{\heartsuit}{\mathord}{mtpsymbol}{169}
\DeclareMathSymbol{\spadesuit}{\mathord}{mtpsymbol}{170}
\DeclareMathSymbol{\smallint}{\mathop}{mtpsymbol}{242}
\DeclareMathSymbol{\wedge}{\mathbin}{mtpsymbol}{217}
\DeclareMathSymbol{\vee}{\mathbin}{mtpsymbol}{218}
\DeclareMathSymbol{\cap}{\mathbin}{mtpsymbol}{199}
\DeclareMathSymbol{\cup}{\mathbin}{mtpsymbol}{200}
\DeclareMathSymbol{\bullet}{\mathbin}{mtpsymbol}{183}
\DeclareMathSymbol{\div}{\mathbin}{mtpsymbol}{184}
\DeclareMathSymbol{\otimes}{\mathbin}{mtpsymbol}{196}
\DeclareMathSymbol{\oplus}{\mathbin}{mtpsymbol}{197}
\DeclareMathSymbol{\pm}{\mathbin}{mtpsymbol}{177}
\DeclareMathSymbol{\times}{\mathbin}{mtpsymbol}{180}
\DeclareMathSymbol{\proptopsy}{\mathrel}{mtpsymbol}{181}
\DeclareMathSymbol{\mid}{\mathrel}{mtpsymbol}{124}
\DeclareMathSymbol{\leq}{\mathrel}{mtpsymbol}{163}
\DeclareMathSymbol{\geq}{\mathrel}{mtpsymbol}{179}
\DeclareMathSymbol{\approx}{\mathrel}{mtpsymbol}{187}
\DeclareMathSymbol{\supset}{\mathrel}{mtpsymbol}{201}
\DeclareMathSymbol{\subset}{\mathrel}{mtpsymbol}{204}
\DeclareMathSymbol{\supseteq}{\mathrel}{mtpsymbol}{202}
```

```
\DeclareMathSymbol{\subseteq}{\mathrel}{mtpsymbol}{205}
\DeclareMathSymbol{\in}{\mathrel}{mtpsymbol}{206}
\DeclareMathSymbol{\sim}{\mathrel}{mtpsymbol}{126}
\let\cong\undefined
\DeclareMathSymbol{\cong}{\mathrel}{mtpsymbol}{64}
\DeclareMathSymbol{\perp}{\mathrel}{mtpsymbol}{94}
\DeclareMathSymbol{\equiv}{\mathrel}{mtpsymbol}{186}
\let\notin\undefined
\DeclareMathSymbol{\notin}{\mathrel}{mtpsymbol}{207}
\DeclareMathDelimiter{\rangle}
   {\mathclose}{mtpsymbol}{241}{largesymbols}{"0B}
\DeclareMathDelimiter{\langle}
   {\mathopen}{mtpsymbol}{225}{largesymbols}{"0A}
\fi
```

`symbolre` I like the `\Re` and `\Im` from Symbol, so I overwrite the CM ones.

```
\ifmst@symbolre
\mst@infoline{symbolre: \string\Re\space and \string\Im\space from Symbol font}
\DeclareMathSymbol{\Re}{\mathord}{mtpsymbol}{"C2}
\DeclareMathSymbol{\Im}{\mathord}{mtpsymbol}{"C1}
\DeclareMathSymbol{\DotTriangle}{\mathord}{mtpsymbol}{92}
\fi
```

`Greek letters` LGRgreek > selfGreek > eulergreek > symbolgreek

1.11 I correct some bugs on how eulergreek and symbolgreek interacted.

1.12b more bug fixes.

1.13

* Option LGRgreek.

* Also, a behavior has been changed: it regards the selfGreek case, the default shape is now the one for letters, not for operator-names and digits. This complies to the ISO standard.

* bugfix: version 1.12b did not define the `\omicron` in the case when no Greek-related option was passed to the package.

1.13d has new macros `\MTstandardgreek` and `\MTcustomgreek`. And in the subdued case `\MTstandardgreek` is done when switching to the normal or bold math versions (previously something like this was only done in case of `LGRgreek` option.)

```
\let\mst@mathord\mathalpha
\mst@goaheadtrue
\ifmst@selfGreek
    \def\mst@font@tbu{mtselfGreekfont}
\else
    \ifmst@eulergreek
        \def\mst@font@tbu{mteulervm}
    \else
      \ifmst@symbolgreek
        \def\mst@font@tbu{mtpsymbol}
        \let\mst@mathord\mathord
      \else
         \ifmst@LGRgreek
            \mst@goaheadfalse
         \else
```

The `\omicron` requires special treatment. By default we use the o from the (original) normal alphabet, if eulergreek or symbolgreek we adapt. There is also a special adjustment if the package `fourier` was loaded in its `upright` variant: we then take `\omicron` from the (original) rm alphabet.

```
           \mst@goaheadfalse
           \def\mst@omicron {\mst@alph@omicron{o}}
         \fi
      \fi
   \fi
\fi
\ifmst@goahead
  \DeclareMathSymbol{\mst@Alpha}{\mst@mathord}{\mst@font@tbu}{"41}
  \DeclareMathSymbol{\mst@Beta}{\mst@mathord}{\mst@font@tbu}{"42}
  \DeclareMathSymbol{\mst@Epsilon}{\mst@mathord}{\mst@font@tbu}{"45}
  \DeclareMathSymbol{\mst@Zeta}{\mst@mathord}{\mst@font@tbu}{"5A}
  \DeclareMathSymbol{\mst@Eta}{\mst@mathord}{\mst@font@tbu}{"48}
  \DeclareMathSymbol{\mst@Iota}{\mst@mathord}{\mst@font@tbu}{"49}
  \DeclareMathSymbol{\mst@Kappa}{\mst@mathord}{\mst@font@tbu}{"4B}
  \DeclareMathSymbol{\mst@Mu}{\mst@mathord}{\mst@font@tbu}{"4D}
  \DeclareMathSymbol{\mst@Nu}{\mst@mathord}{\mst@font@tbu}{"4E}
  \DeclareMathSymbol{\mst@Omicron}{\mst@mathord}{\mst@font@tbu}{"4F}
  \DeclareMathSymbol{\mst@Rho}{\mst@mathord}{\mst@font@tbu}{"50}
  \DeclareMathSymbol{\mst@Tau}{\mst@mathord}{\mst@font@tbu}{"54}
  \DeclareMathSymbol{\mst@Chi}{\mst@mathord}{\mst@font@tbu}{"58}
```

When we in fact use Symbol, we have to correct `\Rho` and `\Chi`. And `\Digamma` is non-existent in fact (no F in Symbol, F codes a `\Phi`).

```
  \ifx\mst@mathord\mathord
```

symbolgreek but neither eulergreek nor selfGreek
Attention le P de Symbol est un `\Pi` pas un `\Rho`.
Attention le X de Symbol est un `\Xi` pas un `\Chi`.
Attention le F de Symbol est un `\Phi`. Il n'y a pas de `\Digamma`.

```
    \DeclareMathSymbol{\mst@Rho}{\mathord}{mtpsymbol}{"52}
    \DeclareMathSymbol{\mst@Chi}{\mathord}{mtpsymbol}{"43}
    \DeclareMathSymbol{\mst@Gamma}{\mathord}{mtpsymbol}{"47}
    \DeclareMathSymbol{\mst@Delta}{\mathord}{mtpsymbol}{"44}
    \DeclareMathSymbol{\mst@Theta}{\mathord}{mtpsymbol}{"51}
    \DeclareMathSymbol{\mst@Lambda}{\mathord}{mtpsymbol}{"4C}
    \DeclareMathSymbol{\mst@Xi}{\mathord}{mtpsymbol}{"58}
    \DeclareMathSymbol{\mst@Pi}{\mathord}{mtpsymbol}{"50}
    \DeclareMathSymbol{\mst@Sigma}{\mathord}{mtpsymbol}{"53}
    \DeclareMathSymbol{\mst@Upsilon}{\mathord}{mtpsymbol}{"A1}
    \DeclareMathSymbol{\mst@Phi}{\mathord}{mtpsymbol}{"46}
    \DeclareMathSymbol{\mst@Psi}{\mathord}{mtpsymbol}{"59}
    \DeclareMathSymbol{\mst@Omega}{\mathord}{mtpsymbol}{"57}
  \else
```

not symbolgreek but eulergreek or selfGreek. Note `2015/10/31` : apparemment à un moment dans le passé je considérais eulergreek et selfGreek comme pouvant être utilisés simultanément car j'avais ici "or both". Mais je laisse tomber tout effort réel de m'en préoccuper.

```
    \DeclareMathSymbol\mst@Digamma  {\mathalpha}{\mst@font@tbu}{"46}
    \DeclareMathSymbol\mst@Gamma    {\mathalpha}{\mst@font@tbu}{"00}
    \DeclareMathSymbol\mst@Delta    {\mathalpha}{\mst@font@tbu}{"01}
    \DeclareMathSymbol\mst@Theta    {\mathalpha}{\mst@font@tbu}{"02}
    \DeclareMathSymbol\mst@Lambda   {\mathalpha}{\mst@font@tbu}{"03}
    \DeclareMathSymbol\mst@Xi       {\mathalpha}{\mst@font@tbu}{"04}
    \DeclareMathSymbol\mst@Pi       {\mathalpha}{\mst@font@tbu}{"05}
    \DeclareMathSymbol\mst@Sigma    {\mathalpha}{\mst@font@tbu}{"06}
    \DeclareMathSymbol\mst@Upsilon  {\mathalpha}{\mst@font@tbu}{"07}
    \DeclareMathSymbol\mst@Phi      {\mathalpha}{\mst@font@tbu}{"08}
    \DeclareMathSymbol\mst@Psi      {\mathalpha}{\mst@font@tbu}{"09}
    \DeclareMathSymbol\mst@Omega    {\mathalpha}{\mst@font@tbu}{"0A}
  \fi
\fi
```

There are differences regarding Euler and Symbol with respect to the available var-letters. We include one or two things like the `wp` and the `partial`.

The lower case Greek letters in default LaTeX are of type `mathord`. If we use the Euler font it is perhaps better to have them be of type `mathalpha`

```
\ifmst@goahead
  \ifmst@eulergreek
  \DeclareMathSymbol{\mst@alpha}  {\mathalpha}{mteulervm}{"0B}
  \DeclareMathSymbol{\mst@beta}   {\mathalpha}{mteulervm}{"0C}
  \DeclareMathSymbol{\mst@gamma}  {\mathalpha}{mteulervm}{"0D}
  \DeclareMathSymbol{\mst@delta}  {\mathalpha}{mteulervm}{"0E}
  \DeclareMathSymbol{\mst@epsilon}{\mathalpha}{mteulervm}{"0F}
  \DeclareMathSymbol{\mst@zeta}   {\mathalpha}{mteulervm}{"10}
  \DeclareMathSymbol{\mst@eta}    {\mathalpha}{mteulervm}{"11}
  \DeclareMathSymbol{\mst@theta}  {\mathalpha}{mteulervm}{"12}
  \DeclareMathSymbol{\mst@iota}   {\mathalpha}{mteulervm}{"13}
  \DeclareMathSymbol{\mst@kappa}  {\mathalpha}{mteulervm}{"14}
  \DeclareMathSymbol{\mst@lambda} {\mathalpha}{mteulervm}{"15}
  \DeclareMathSymbol{\mst@mu}     {\mathalpha}{mteulervm}{"16}
  \DeclareMathSymbol{\mst@nu}     {\mathalpha}{mteulervm}{"17}
  \DeclareMathSymbol{\mst@xi}     {\mathalpha}{mteulervm}{"18}
  \DeclareMathSymbol{\mst@omicron}{\mathalpha}{mteulervm}{"6F}
  \DeclareMathSymbol{\mst@pi}     {\mathalpha}{mteulervm}{"19}
  \DeclareMathSymbol{\mst@rho}    {\mathalpha}{mteulervm}{"1A}
  \DeclareMathSymbol{\mst@sigma}  {\mathalpha}{mteulervm}{"1B}
  \DeclareMathSymbol{\mst@tau}    {\mathalpha}{mteulervm}{"1C}
  \DeclareMathSymbol{\mst@upsilon}{\mathalpha}{mteulervm}{"1D}
  \DeclareMathSymbol{\mst@phi}    {\mathalpha}{mteulervm}{"1E}
  \DeclareMathSymbol{\mst@chi}    {\mathalpha}{mteulervm}{"1F}
  \DeclareMathSymbol{\mst@psi}    {\mathalpha}{mteulervm}{"20}
  \DeclareMathSymbol{\mst@omega}  {\mathalpha}{mteulervm}{"21}
%
  \DeclareMathSymbol{\mst@varepsilon}{\mathalpha}{mteulervm}{"22}
  \DeclareMathSymbol{\mst@vartheta}{\mathalpha}{mteulervm}{"23}
  \DeclareMathSymbol{\mst@varpi}  {\mathalpha}{mteulervm}{"24}
  \let\mst@varrho=\mst@rho
  \let\mst@varsigma=\mst@sigma
  \DeclareMathSymbol{\mst@varphi} {\mathalpha}{mteulervm}{"27}
%
  \DeclareMathSymbol{\mst@partial}{\mathalpha}{mteulervm}{"40}
  \DeclareMathSymbol{\mst@wp}{\mathalpha}{mteulervm}{"7D}
  \DeclareMathSymbol{\mst@ell}{\mathalpha}{mteulervm}{"60}
 \else
  \ifmst@symbolgreek
  \DeclareMathSymbol{\mst@alpha}{\mathord}{mtpsymbol}{"61}
  \DeclareMathSymbol{\mst@beta}{\mathord}{mtpsymbol}{"62}
  \DeclareMathSymbol{\mst@gamma}{\mathord}{mtpsymbol}{"67}
  \DeclareMathSymbol{\mst@delta}{\mathord}{mtpsymbol}{"64}
  \DeclareMathSymbol{\mst@epsilon}{\mathord}{mtpsymbol}{"65}
  \DeclareMathSymbol{\mst@zeta}{\mathord}{mtpsymbol}{"7A}
  \DeclareMathSymbol{\mst@eta}{\mathord}{mtpsymbol}{"68}
  \DeclareMathSymbol{\mst@theta}{\mathord}{mtpsymbol}{"71}
  \DeclareMathSymbol{\mst@iota}{\mathord}{mtpsymbol}{"69}
  \DeclareMathSymbol{\mst@kappa}{\mathord}{mtpsymbol}{"6B}
  \DeclareMathSymbol{\mst@lambda}{\mathord}{mtpsymbol}{"6C}
  \DeclareMathSymbol{\mst@mu}{\mathord}{mtpsymbol}{"6D}
  \DeclareMathSymbol{\mst@nu}{\mathord}{mtpsymbol}{"6E}
  \DeclareMathSymbol{\mst@xi}{\mathord}{mtpsymbol}{"78}
  \DeclareMathSymbol{\mst@omicron}{\mathord}{mtpsymbol}{"6F}
  \DeclareMathSymbol{\mst@pi}{\mathord}{mtpsymbol}{"70}
  \DeclareMathSymbol{\mst@rho}{\mathord}{mtpsymbol}{"72}
  \DeclareMathSymbol{\mst@sigma}{\mathord}{mtpsymbol}{"73}
  \DeclareMathSymbol{\mst@tau}{\mathord}{mtpsymbol}{"74}
  \DeclareMathSymbol{\mst@upsilon}{\mathord}{mtpsymbol}{"75}
  \DeclareMathSymbol{\mst@phi}{\mathord}{mtpsymbol}{"66}
  \DeclareMathSymbol{\mst@chi}{\mathord}{mtpsymbol}{"63}
  \DeclareMathSymbol{\mst@psi}{\mathord}{mtpsymbol}{"79}
  \DeclareMathSymbol{\mst@omega}{\mathord}{mtpsymbol}{"77}
  \let\mst@varepsilon=\mst@epsilon
  \DeclareMathSymbol{\mst@vartheta}{\mathord}{mtpsymbol}{"4A}
  \DeclareMathSymbol{\mst@varpi}{\mathord}{mtpsymbol}{"76}
  \let\mst@varrho=\mst@rho
  \DeclareMathSymbol{\mst@varsigma}{\mathord}{mtpsymbol}{"56}
  \DeclareMathSymbol{\mst@varphi}{\mathord}{mtpsymbol}{"6A}
  \DeclareMathSymbol{\mst@partial}{\mathord}{mtpsymbol}{"B6}
  \DeclareMathSymbol{\mst@wp}{\mathord}{mtpsymbol}{"C3}
  \fi
 \fi
\fi
```

`\alphaup etc...`

Completely refactored at `1.3y` to define `\Alphaup`, `\Alphait`, `\alphaup`, `\alphait`, etc… and prepare templates `\Alpha`, …, `\alpha`, …, which when activating a math version will be submitted to an `\expanded`, whose behavior will depend on version-specific conditionals.

```
\ifmst@LGRgreek
% cf http://milde.users.sourceforge.net/LGR/lgrxenc.def.html
% et greek.ldf du package babel
  \DeclareMathSymbol{\Alphaup}{\mathalpha}{mtgreekup}{65}
  \DeclareMathSymbol{\Betaup}{\mathalpha}{mtgreekup}{66}
  \DeclareMathSymbol{\Epsilonup}{\mathalpha}{mtgreekup}{69}
  \DeclareMathSymbol{\Zetaup}{\mathalpha}{mtgreekup}{90}
  \DeclareMathSymbol{\Etaup}{\mathalpha}{mtgreekup}{72}
  \DeclareMathSymbol{\Iotaup}{\mathalpha}{mtgreekup}{73}
  \DeclareMathSymbol{\Kappaup}{\mathalpha}{mtgreekup}{75}
  \DeclareMathSymbol{\Muup}{\mathalpha}{mtgreekup}{77}
  \DeclareMathSymbol{\Nuup}{\mathalpha}{mtgreekup}{78}
  \DeclareMathSymbol{\Omicronup}{\mathalpha}{mtgreekup}{79}
  \DeclareMathSymbol{\Rhoup}{\mathalpha}{mtgreekup}{82}
  \DeclareMathSymbol{\Tauup}{\mathalpha}{mtgreekup}{84}
  \DeclareMathSymbol{\Chiup}{\mathalpha}{mtgreekup}{81}
  %
  \DeclareMathSymbol{\Alphait}{\mathalpha}{mtgreekit}{65}
  \DeclareMathSymbol{\Betait}{\mathalpha}{mtgreekit}{66}
  \DeclareMathSymbol{\Epsilonit}{\mathalpha}{mtgreekit}{69}
  \DeclareMathSymbol{\Zetait}{\mathalpha}{mtgreekit}{90}
  \DeclareMathSymbol{\Etait}{\mathalpha}{mtgreekit}{72}
  \DeclareMathSymbol{\Iotait}{\mathalpha}{mtgreekit}{73}
  \DeclareMathSymbol{\Kappait}{\mathalpha}{mtgreekit}{75}
  \DeclareMathSymbol{\Muit}{\mathalpha}{mtgreekit}{77}
  \DeclareMathSymbol{\Nuit}{\mathalpha}{mtgreekit}{78}
  \DeclareMathSymbol{\Omicronit}{\mathalpha}{mtgreekit}{79}
  \DeclareMathSymbol{\Rhoit}{\mathalpha}{mtgreekit}{82}
  \DeclareMathSymbol{\Tauit}{\mathalpha}{mtgreekit}{84}
  \DeclareMathSymbol{\Chiit}{\mathalpha}{mtgreekit}{81}
```

1.3w and earlier had a bug regarding Digamma which was set up to use same font shape as for lowercase digamma.

```
  \DeclareMathSymbol{\Digammaup}{\mathalpha}{mtgreekup}{195}
  \DeclareMathSymbol{\Digammait}{\mathalpha}{mtgreekit}{195}
  %
  \DeclareMathSymbol{\Gammaup}{\mathalpha}{mtgreekup}{71}
  \DeclareMathSymbol{\Deltaup}{\mathalpha}{mtgreekup}{68}
  \DeclareMathSymbol{\Thetaup}{\mathalpha}{mtgreekup}{74}
  \DeclareMathSymbol{\Lambdaup}{\mathalpha}{mtgreekup}{76}
  \DeclareMathSymbol{\Xiup}{\mathalpha}{mtgreekup}{88}
  \DeclareMathSymbol{\Piup}{\mathalpha}{mtgreekup}{80}
  \DeclareMathSymbol{\Sigmaup}{\mathalpha}{mtgreekup}{83}
  \DeclareMathSymbol{\Upsilonup}{\mathalpha}{mtgreekup}{85}
  \DeclareMathSymbol{\Phiup}{\mathalpha}{mtgreekup}{70}
  \DeclareMathSymbol{\Psiup}{\mathalpha}{mtgreekup}{89}
  \DeclareMathSymbol{\Omegaup}{\mathalpha}{mtgreekup}{87}
  %
```

```
\DeclareMathSymbol{\Gammait}{\mathalpha}{mtgreekit}{71}
\DeclareMathSymbol{\Deltait}{\mathalpha}{mtgreekit}{68}
\DeclareMathSymbol{\Thetait}{\mathalpha}{mtgreekit}{74}
\DeclareMathSymbol{\Lambdait}{\mathalpha}{mtgreekit}{76}
\DeclareMathSymbol{\Xiit}{\mathalpha}{mtgreekit}{88}
\DeclareMathSymbol{\Piit}{\mathalpha}{mtgreekit}{80}
\DeclareMathSymbol{\Sigmait}{\mathalpha}{mtgreekit}{83}
\DeclareMathSymbol{\Upsilonit}{\mathalpha}{mtgreekit}{85}
\DeclareMathSymbol{\Phiit}{\mathalpha}{mtgreekit}{70}
\DeclareMathSymbol{\Psiit}{\mathalpha}{mtgreekit}{89}
\DeclareMathSymbol{\Omegait}{\mathalpha}{mtgreekit}{87}
%
\def\mst@Alpha{\ifmst@greek@upper@up\Alphaup\else\Alphait\fi}%
\def\mst@Beta{\ifmst@greek@upper@up\Betaup\else\Betait\fi}%
\def\mst@Epsilon{\ifmst@greek@upper@up\Epsilonup\else\Epsilonit\fi}%
\def\mst@Zeta{\ifmst@greek@upper@up\Zetaup\else\Zetait\fi}%
\def\mst@Eta{\ifmst@greek@upper@up\Etaup\else\Etait\fi}%
\def\mst@Iota{\ifmst@greek@upper@up\Iotaup\else\Iotait\fi}%
\def\mst@Kappa{\ifmst@greek@upper@up\Kappaup\else\Kappait\fi}%
\def\mst@Mu{\ifmst@greek@upper@up\Muup\else\Muit\fi}%
\def\mst@Nu{\ifmst@greek@upper@up\Nuup\else\Nuit\fi}%
\def\mst@Omicron{\ifmst@greek@upper@up\Omicronup\else\Omicronit\fi}%
\def\mst@Rho{\ifmst@greek@upper@up\Rhoup\else\Rhoit\fi}%
\def\mst@Tau{\ifmst@greek@upper@up\Tauup\else\Tauit\fi}%
\def\mst@Chi{\ifmst@greek@upper@up\Chiup\else\Chiit\fi}%
%
\def\mst@Digamma{\ifmst@greek@upper@up\Digammaup\else\Digammait\fi}%
%
\def\mst@Gamma{\ifmst@greek@upper@up\Gammaup\else\Gammait\fi}%
\def\mst@Delta{\ifmst@greek@upper@up\Deltaup\else\Deltait\fi}%
\def\mst@Theta{\ifmst@greek@upper@up\Thetaup\else\Thetait\fi}%
\def\mst@Lambda{\ifmst@greek@upper@up\Lambdaup\else\Lambdait\fi}%
\def\mst@Xi{\ifmst@greek@upper@up\Xiup\else\Xiit\fi}%
\def\mst@Pi{\ifmst@greek@upper@up\Piup\else\Piit\fi}%
\def\mst@Sigma{\ifmst@greek@upper@up\Sigmaup\else\Sigmait\fi}%
\def\mst@Upsilon{\ifmst@greek@upper@up\Upsilonup\else\Upsilonit\fi}%
\def\mst@Phi{\ifmst@greek@upper@up\Phiup\else\Phiit\fi}%
\def\mst@Psi{\ifmst@greek@upper@up\Psiup\else\Psiit\fi}%
\def\mst@Omega{\ifmst@greek@upper@up\Omegaup\else\Omegait\fi}%
%
\DeclareMathSymbol{\alphaup}{\mathalpha}{mtgreekup}{97}
\DeclareMathSymbol{\betaup}{\mathalpha}{mtgreekup}{98}
\DeclareMathSymbol{\gammaup}{\mathalpha}{mtgreekup}{103}
\DeclareMathSymbol{\deltaup}{\mathalpha}{mtgreekup}{100}
\DeclareMathSymbol{\epsilonup}{\mathalpha}{mtgreekup}{101}
\DeclareMathSymbol{\zetaup}{\mathalpha}{mtgreekup}{122}
\DeclareMathSymbol{\etaup}{\mathalpha}{mtgreekup}{104}
\DeclareMathSymbol{\thetaup}{\mathalpha}{mtgreekup}{106}
\DeclareMathSymbol{\iotaup}{\mathalpha}{mtgreekup}{105}
```

```
\DeclareMathSymbol{\kappaup}{\mathalpha}{mtgreekup}{107}
\DeclareMathSymbol{\lambdaup}{\mathalpha}{mtgreekup}{108}
\DeclareMathSymbol{\muup}{\mathalpha}{mtgreekup}{109}
\DeclareMathSymbol{\nuup}{\mathalpha}{mtgreekup}{110}
\DeclareMathSymbol{\xiup}{\mathalpha}{mtgreekup}{120}
\DeclareMathSymbol{\omicronup}{\mathalpha}{mtgreekup}{111}
\DeclareMathSymbol{\piup}{\mathalpha}{mtgreekup}{112}
\DeclareMathSymbol{\rhoup}{\mathalpha}{mtgreekup}{114}
\DeclareMathSymbol{\sigmaup}{\mathalpha}{mtgreekup}{115}
\DeclareMathSymbol{\tauup}{\mathalpha}{mtgreekup}{116}
\DeclareMathSymbol{\upsilonup}{\mathalpha}{mtgreekup}{117}
\DeclareMathSymbol{\phiup}{\mathalpha}{mtgreekup}{102}
\DeclareMathSymbol{\chiup}{\mathalpha}{mtgreekup}{113}
\DeclareMathSymbol{\psiup}{\mathalpha}{mtgreekup}{121}
\DeclareMathSymbol{\omegaup}{\mathalpha}{mtgreekup}{119}
%
\DeclareMathSymbol{\digammaup}{\mathalpha}{mtgreekup}{147}
```

Only varsigma defined (I should check this again).

```
\DeclareMathSymbol{\varsigmaup}{\mathalpha}{mtgreekup}{99}
%
\DeclareMathSymbol{\alphait}{\mathalpha}{mtgreekit}{97}
\DeclareMathSymbol{\betait}{\mathalpha}{mtgreekit}{98}
\DeclareMathSymbol{\gammait}{\mathalpha}{mtgreekit}{103}
\DeclareMathSymbol{\deltait}{\mathalpha}{mtgreekit}{100}
\DeclareMathSymbol{\epsilonit}{\mathalpha}{mtgreekit}{101}
\DeclareMathSymbol{\zetait}{\mathalpha}{mtgreekit}{122}
\DeclareMathSymbol{\etait}{\mathalpha}{mtgreekit}{104}
\DeclareMathSymbol{\thetait}{\mathalpha}{mtgreekit}{106}
\DeclareMathSymbol{\iotait}{\mathalpha}{mtgreekit}{105}
\DeclareMathSymbol{\kappait}{\mathalpha}{mtgreekit}{107}
\DeclareMathSymbol{\lambdait}{\mathalpha}{mtgreekit}{108}
\DeclareMathSymbol{\muit}{\mathalpha}{mtgreekit}{109}
\DeclareMathSymbol{\nuit}{\mathalpha}{mtgreekit}{110}
\DeclareMathSymbol{\xiit}{\mathalpha}{mtgreekit}{120}
\DeclareMathSymbol{\omicronit}{\mathalpha}{mtgreekit}{111}
\DeclareMathSymbol{\piit}{\mathalpha}{mtgreekit}{112}
\DeclareMathSymbol{\rhoit}{\mathalpha}{mtgreekit}{114}
\DeclareMathSymbol{\sigmait}{\mathalpha}{mtgreekit}{115}
\DeclareMathSymbol{\tauit}{\mathalpha}{mtgreekit}{116}
\DeclareMathSymbol{\upsilonit}{\mathalpha}{mtgreekit}{117}
\DeclareMathSymbol{\phiit}{\mathalpha}{mtgreekit}{102}
\DeclareMathSymbol{\chiit}{\mathalpha}{mtgreekit}{113}
\DeclareMathSymbol{\psiit}{\mathalpha}{mtgreekit}{121}
\DeclareMathSymbol{\omegait}{\mathalpha}{mtgreekit}{119}
%
\DeclareMathSymbol{\digammait}{\mathalpha}{mtgreekit}{147}
\DeclareMathSymbol{\varsigmait}{\mathalpha}{mtgreekit}{99}
%
\def\mst@alpha{\ifmst@greek@lower@up\alphaup\else\alphait\fi}%
```

```
  \def\mst@beta{\ifmst@greek@lower@up\betaup\else\betait\fi}%
  \def\mst@gamma{\ifmst@greek@lower@up\gammaup\else\gammait\fi}%
  \def\mst@delta{\ifmst@greek@lower@up\deltaup\else\deltait\fi}%
  \def\mst@epsilon{\ifmst@greek@lower@up\epsilonup\else\epsilonit\fi}%
  \def\mst@zeta{\ifmst@greek@lower@up\zetaup\else\zetait\fi}%
  \def\mst@eta{\ifmst@greek@lower@up\etaup\else\etait\fi}%
  \def\mst@theta{\ifmst@greek@lower@up\thetaup\else\thetait\fi}%
  \def\mst@iota{\ifmst@greek@lower@up\iotaup\else\iotait\fi}%
  \def\mst@kappa{\ifmst@greek@lower@up\kappaup\else\kappait\fi}%
  \def\mst@lambda{\ifmst@greek@lower@up\lambdaup\else\lambdait\fi}%
  \def\mst@mu{\ifmst@greek@lower@up\muup\else\muit\fi}%
  \def\mst@nu{\ifmst@greek@lower@up\nuup\else\nuit\fi}%
  \def\mst@xi{\ifmst@greek@lower@up\xiup\else\xiit\fi}%
  \def\mst@omicron{\ifmst@greek@lower@up\omicronup\else\omicronit\fi}%
  \def\mst@pi{\ifmst@greek@lower@up\piup\else\piit\fi}%
  \def\mst@rho{\ifmst@greek@lower@up\rhoup\else\rhoit\fi}%
  \def\mst@sigma{\ifmst@greek@lower@up\sigmaup\else\sigmait\fi}%
  \def\mst@tau{\ifmst@greek@lower@up\tauup\else\tauit\fi}%
  \def\mst@upsilon{\ifmst@greek@lower@up\upsilonup\else\upsilonit\fi}%
  \def\mst@phi{\ifmst@greek@lower@up\phiup\else\phiit\fi}%
  \def\mst@chi{\ifmst@greek@lower@up\chiup\else\chiit\fi}%
  \def\mst@psi{\ifmst@greek@lower@up\psiup\else\psiit\fi}%
  \def\mst@omega{\ifmst@greek@lower@up\omegaup\else\omegait\fi}%
    %
  \def\mst@digamma{\ifmst@greek@lower@up\digammaup\else\digammait\fi}%
  \def\mst@varsigma{\ifmst@greek@lower@up\varsigmaup\else\varsigmait\fi}%
\fi
```

`\MTstandardgreek`
`\MTcustomgreek`
`MTrecordstandardgreek`

1.3d 2014/05/23 defines the commands `\MTstandardgreek` and `\MTcustomgreek` for package and user. I leave `\MTrecordstandardgreek` undocumented as I don't want to encourage people to load math packages after **mathastext**.

1.3h 2015/10/31: corrected `\MTcustomgreek` as it caused `\ell` to become undefined under option `symbolgreek` and, much more catastrophic, caused `\alpha`, etc.. to become undefined under option `selfGreek` !

```
\newcommand*{\MTstandardgreek}{}
\newcommand*{\MTcustomgreek}{}
\newcommand*{\MTrecordstandardgreek}{}
\ifmst@customgreek
 \renewcommand*{\MTrecordstandardgreek}{%
    \let\mst@origAlpha\Alpha
    \let\mst@origBeta\Beta
    \let\mst@origGamma\Gamma
    \let\mst@origDelta\Delta
    \let\mst@origEpsilon\Epsilon
    \let\mst@origZeta\Zeta
    \let\mst@origEta\Eta
    \let\mst@origTheta\Theta
    \let\mst@origIota\Iota
    \let\mst@origKappa\Kappa
```

```
  \let\mst@origLambda\Lambda
  \let\mst@origMu\Mu
  \let\mst@origNu\Nu
  \let\mst@origXi\Xi
  \let\mst@origOmicron\Omicron
  \let\mst@origPi\Pi
  \let\mst@origRho\Rho
  \let\mst@origSigma\Sigma
  \let\mst@origTau\Tau
  \let\mst@origUpsilon\Upsilon
  \let\mst@origPhi\Phi
  \let\mst@origChi\Chi
  \let\mst@origPsi\Psi
  \let\mst@origOmega\Omega
%
  \let\mst@origalpha\alpha
  \let\mst@origbeta\beta
  \let\mst@origgamma\gamma
  \let\mst@origdelta\delta
  \let\mst@origepsilon\epsilon
  \let\mst@origvarepsilon\varepsilon
  \let\mst@origzeta\zeta
  \let\mst@origeta\eta
  \let\mst@origtheta\theta
  \let\mst@origvartheta\vartheta
  \let\mst@origiota\iota
  \let\mst@origkappa\kappa
  \let\mst@origlambda\lambda
  \let\mst@origmu\mu
  \let\mst@orignu\nu
  \let\mst@origxi\xi
  \let\mst@origomicron\omicron
  \let\mst@origpi\pi
  \let\mst@origvarpi\varpi
  \let\mst@origrho\rho
  \let\mst@origvarrho\varrho
  \let\mst@origsigma\sigma
  \let\mst@origvarsigma\varsigma
  \let\mst@origtau\tau
  \let\mst@origupsilon\upsilon
  \let\mst@origphi\phi
  \let\mst@origvarphi\varphi
  \let\mst@origchi\chi
  \let\mst@origpsi\psi
  \let\mst@origomega\omega
  \let\mst@origDigamma\Digamma
  \let\mst@origdigamma\digamma
%
  \let\mst@origpartial\partial
```

```
    \let\mst@origwp\wp
    \let\mst@origell\ell }%
\MTrecordstandardgreek
\renewcommand*{\MTstandardgreek}{%
    \let\Alpha\mst@origAlpha
    \let\Beta\mst@origBeta
    \let\Gamma\mst@origGamma
    \let\Delta\mst@origDelta
    \let\Epsilon\mst@origEpsilon
    \let\Zeta\mst@origZeta
    \let\Eta\mst@origEta
    \let\Theta\mst@origTheta
    \let\Iota\mst@origIota
    \let\Kappa\mst@origKappa
    \let\Lambda\mst@origLambda
    \let\Mu\mst@origMu
    \let\Nu\mst@origNu
    \let\Xi\mst@origXi
    \let\Omicron\mst@origOmicron
    \let\Pi\mst@origPi
    \let\Rho\mst@origRho
    \let\Sigma\mst@origSigma
    \let\Tau\mst@origTau
    \let\Upsilon\mst@origUpsilon
    \let\Phi\mst@origPhi
    \let\Chi\mst@origChi
    \let\Psi\mst@origPsi
    \let\Omega\mst@origOmega
%
    \let\alpha\mst@origalpha
    \let\beta\mst@origbeta
    \let\gamma\mst@origgamma
    \let\delta\mst@origdelta
    \let\epsilon\mst@origepsilon
    \let\varepsilon\mst@origvarepsilon
    \let\zeta\mst@origzeta
    \let\eta\mst@origeta
    \let\theta\mst@origtheta
    \let\vartheta\mst@origvartheta
    \let\iota\mst@origiota
    \let\kappa\mst@origkappa
    \let\lambda\mst@origlambda
    \let\mu\mst@origmu
    \let\nu\mst@orignu
    \let\xi\mst@origxi
    \let\omicron\mst@origomicron
    \let\pi\mst@origpi
    \let\varpi\mst@origvarpi
    \let\rho\mst@origrho
    \let\varrho\mst@origvarrho
    \let\sigma\mst@origsigma
    \let\varsigma\mst@origvarsigma
    \let\tau\mst@origtau
    \let\upsilon\mst@origupsilon
    \let\phi\mst@origphi
    \let\varphi\mst@origvarphi
    \let\chi\mst@origchi
    \let\psi\mst@origpsi
    \let\omega\mst@origomega
    \let\Digamma\mst@origDigamma
    \let\digamma\mst@origdigamma
%
    \let\partial\mst@origpartial
    \let\wp\mst@origwp
    \let\ell\mst@origell
}%
\ifmst@greekplus
```

1.3za implementation of `LGRgreek+` option. It is not exactly clear what we should do for `\mathnormal` and `\mathnormalbold`.

This definition allows usage of `\alpha` for example in numerical context. To be completely clean perhaps we should get rid of final `\fi`, but old-fashioned LaTeX does not have built-in conveniences, were it not for the nested if's simple `\expandafter` would do, but here we would need three in four places. Or simply wrap the whole in `\expanded`. Anyway, not really important.

```
\def\mst@define@lowergreekletter#1#2{%
    \protected\def#1{\ifcase\mst@mathalph
        \ifmst@greek@lower@up\mathgreekup{#2}\else\mathgreekit{#2}\fi
      \or % rm
        \mathgreekup{#2}%
      \or % bf
        \mathgreekupbold{#2}%
      \or % it
        \mathgreekit{#2}%
      \or % normalbold
        \ifmst@greek@lower@up\mathgreekupbold{#2}\else\mathgreekitbold{#2}\fi
      \else #2\fi}%
    }
\def\mst@define@uppergreekletter#1#2{%
    \protected\def#1{\ifcase\mst@mathalph
        \ifmst@greek@upper@up\mathgreekup{#2}\else\mathgreekit{#2}\fi
      \or % rm
        \mathgreekup{#2}%
      \or % bf
        \mathgreekupbold{#2}%
      \or % it
        \mathgreekit{#2}%
      \or % mathnormalbold
        \ifmst@greek@upper@up\mathgreekupbold{#2}\else\mathgreekitbold{#2}\fi
      \else #2\fi}%
    }
\renewcommand*{\MTcustomgreek}{%
    \mst@define@uppergreekletter\Alpha\mst@Alpha
    \mst@define@uppergreekletter\Beta\mst@Beta
    \mst@define@uppergreekletter\Epsilon\mst@Epsilon
    \mst@define@uppergreekletter\Zeta\mst@Zeta
    \mst@define@uppergreekletter\Eta\mst@Eta
    \mst@define@uppergreekletter\Iota\mst@Iota
    \mst@define@uppergreekletter\Kappa\mst@Kappa
    \mst@define@uppergreekletter\Mu\mst@Mu
    \mst@define@uppergreekletter\Nu\mst@Nu
    \mst@define@uppergreekletter\Omicron\mst@Omicron
    \mst@define@uppergreekletter\Rho\mst@Rho
    \mst@define@uppergreekletter\Tau\mst@Tau
    \mst@define@uppergreekletter\Chi\mst@Chi
    \mst@define@uppergreekletter\Digamma\mst@Digamma
    \mst@define@uppergreekletter\Gamma\mst@Gamma
    \mst@define@uppergreekletter\Delta\mst@Delta
    \mst@define@uppergreekletter\Theta\mst@Theta
    \mst@define@uppergreekletter\Lambda\mst@Lambda
    \mst@define@uppergreekletter\Xi\mst@Xi
    \mst@define@uppergreekletter\Pi\mst@Pi
    \mst@define@uppergreekletter\Sigma\mst@Sigma
    \mst@define@uppergreekletter\Upsilon\mst@Upsilon
    \mst@define@uppergreekletter\Phi\mst@Phi
    \mst@define@uppergreekletter\Psi\mst@Psi
    \mst@define@uppergreekletter\Omega\mst@Omega
    \mst@define@lowergreekletter\alpha\mst@alpha
    \mst@define@lowergreekletter\beta\mst@beta
    \mst@define@lowergreekletter\gamma\mst@gamma
    \mst@define@lowergreekletter\delta\mst@delta
    \mst@define@lowergreekletter\epsilon\mst@epsilon
    \mst@define@lowergreekletter\zeta\mst@zeta
    \mst@define@lowergreekletter\eta\mst@eta
    \mst@define@lowergreekletter\theta\mst@theta
    \mst@define@lowergreekletter\iota\mst@iota
    \mst@define@lowergreekletter\kappa\mst@kappa
    \mst@define@lowergreekletter\lambda\mst@lambda
    \mst@define@lowergreekletter\mu\mst@mu
    \mst@define@lowergreekletter\nu\mst@nu
    \mst@define@lowergreekletter\xi\mst@xi
    \mst@define@lowergreekletter\omicron\mst@omicron
    \mst@define@lowergreekletter\pi\mst@pi
    \mst@define@lowergreekletter\rho\mst@rho
    \mst@define@lowergreekletter\sigma\mst@sigma
    \mst@define@lowergreekletter\tau\mst@tau
    \mst@define@lowergreekletter\upsilon\mst@upsilon
    \mst@define@lowergreekletter\phi\mst@phi
    \mst@define@lowergreekletter\chi\mst@chi
    \mst@define@lowergreekletter\psi\mst@psi
    \mst@define@lowergreekletter\omega\mst@omega
    \mst@define@lowergreekletter\varsigma\mst@varsigma
    \mst@define@lowergreekletter\digamma\mst@digamma
}%
\else
```

Under `selfGreek` or other Greek option but not `LGRgreek`, these Greek letter control sequences are already `\mathchar`'s, but under `LGRgreek` they need (well not really, but I feel it is cleaner) expansion which will react to the Boolean saying if using 'upright' or 'italic'. This Boolean setting is recorded when declaring a math version and reenacted when `\MTversion` is encountered in the document body. We must be careful not to contaminate things in the principal mode from math version declarations but I think my (now quite old) code is globally designed to achieve this protection see how `\MTDeclareVersion` is done. The `\MTcustomgreek` will always be executed in preamble at least once, except under `subdued` option.

The `\expanded`'s act on unexpanding tokens if not used under `LGRgreek` regimen.

```
\renewcommand*{\MTcustomgreek}{%
  \expanded{%
    \let\noexpand\Alpha\mst@Alpha
    \let\noexpand\Beta\mst@Beta
    \let\noexpand\Epsilon\mst@Epsilon
    \let\noexpand\Zeta\mst@Zeta
    \let\noexpand\Eta\mst@Eta
    \let\noexpand\Iota\mst@Iota
    \let\noexpand\Kappa\mst@Kappa
    \let\noexpand\Mu\mst@Mu
    \let\noexpand\Nu\mst@Nu
    \let\noexpand\Omicron\mst@Omicron
    \let\noexpand\Rho\mst@Rho
    \let\noexpand\Tau\mst@Tau
    \let\noexpand\Chi\mst@Chi
  }%
```

1.3h: `\mst@Digamma` not defined if `symbolgreek` option.

```
    \ifmst@symbolgreek\else
        \expanded{\let\noexpand\Digamma\mst@Digamma}%
    \fi
  \expanded{%
    \let\noexpand\Gamma\mst@Gamma
    \let\noexpand\Delta\mst@Delta
    \let\noexpand\Theta\mst@Theta
    \let\noexpand\Lambda\mst@Lambda
    \let\noexpand\Xi\mst@Xi
    \let\noexpand\Pi\mst@Pi
    \let\noexpand\Sigma\mst@Sigma
    \let\noexpand\Upsilon\mst@Upsilon
    \let\noexpand\Phi\mst@Phi
    \let\noexpand\Psi\mst@Psi
    \let\noexpand\Omega\mst@Omega
  }%
```

`1.3h` `2015/10/31` adds this conditional to correct the bad bug in `1.3d` `2014/05/23` which caused `\alpha` etc... to become undefined under option `selfGreek`.

```
\ifmst@selfGreek\else
 \expanded{%
   \let\noexpand\alpha\mst@alpha
   \let\noexpand\beta\mst@beta
   \let\noexpand\gamma\mst@gamma
   \let\noexpand\delta\mst@delta
   \let\noexpand\epsilon\mst@epsilon
   \let\noexpand\zeta\mst@zeta
   \let\noexpand\eta\mst@eta
   \let\noexpand\theta\mst@theta
   \let\noexpand\iota\mst@iota
   \let\noexpand\kappa\mst@kappa
   \let\noexpand\lambda\mst@lambda
   \let\noexpand\mu\mst@mu
   \let\noexpand\nu\mst@nu
   \let\noexpand\xi\mst@xi
   \let\noexpand\omicron\mst@omicron
   \let\noexpand\pi\mst@pi
   \let\noexpand\rho\mst@rho
   \let\noexpand\sigma\mst@sigma
   \let\noexpand\tau\mst@tau
   \let\noexpand\upsilon\mst@upsilon
   \let\noexpand\phi\mst@phi
   \let\noexpand\chi\mst@chi
   \let\noexpand\psi\mst@psi
   \let\noexpand\omega\mst@omega
   \let\noexpand\varsigma\mst@varsigma
 }%
```

`1.3h`: digamma only defined with option `LGRgreek`.

```
   \ifmst@LGRgreek
       \expanded{\let\noexpand\digamma\mst@digamma}%
   \fi
```

Conditional added `1.3h` `2015/10/31`.

```
   \ifmst@LGRgreek\else
     \let\varepsilon\mst@varepsilon
     \let\vartheta\mst@vartheta
     \let\varpi\mst@varpi
     \let\varrho\mst@varrho
     \let\varphi\mst@varphi
     \let\partial\mst@partial
     \let\wp\mst@wp
```

`1.3h`: no `\mst@ell` if `symbolgreek` (bugfix `1.3h` `2015/10/31`).

```
     \ifmst@symbolgreek\else\let\ell\mst@ell\fi
   \fi
\fi
```

```
}%
\fi
\fi
\let\Mathastextstandardgreek\MTstandardgreek
\let\Mathastextcustomgreek\MTcustomgreek
\ifmst@subdued\else\MTcustomgreek\fi
```

`\inodot`
`\jnodot`

In 1.0, I had them of type `mathord`, here I choose `mathalpha`. If I used `\i` and `\j` from the text font the problem would be with the fontsize, if in scriptstyle. The amsmath `\text` would do the trick.

1.14b 2011/04/02: again this bug in the EU1/EU2 encoding part, as in the code redefining $ etc in math mode (see above). Fixed.

1.3l 2016/01/29: anticipating TL2016 fontspec's switch to `TU`.

1.3t 2018/08/22 removes the definitions done of `\i` and `\j` since 1.12 (as robust commands usable both in text and math mode).

1.3u lets the `\imath` and `\jmath` react to the font encoding at each math version.

1.3v lets the redefined `\imath` and `\jmath` be `\protected`.

The 1.3u change is the cause a crash when attempting to define math version in some encodings such as `OML`. This is same problem as with `\hbar` and minus as endash. Fixed at 1.4d.

1.4d lets the option `noletters` prevent definition of `\inodot` and `\jnodot`.

```
\let\mst@subduedinodot\@empty
\let\mst@nonsubduedinodot\@empty
\ifmst@noletters
\else
\def\mst@subduedinodot{%
   \let\inodot\mst@original@imath
   \let\jnodot\mst@original@jmath
}%
\def\mst@nonsubduedinodot{%
   \expandafter\let\expandafter\inodot
               \csname mst@inodot@mv\math@version\endcsname
   \expandafter\let\expandafter\jnodot
               \csname mst@jnodot@mv\math@version\endcsname
   \ifx\inodot\relax\let\inodot\mst@original@imath\fi
   \ifx\jnodot\relax\let\jnodot\mst@original@jmath\fi
}%
\fi
\def\mst@dothe@inodotstuff#1#2#3#4{%
  \edef\mst@tmp@enc{#3}%
  \if1\mst@OneifUniEnc
    \mst@Umathchardef#1="7 \symmtletterfont "0131 \relax
    \mst@Umathchardef#2="7 \symmtletterfont "0237 \relax
  \else
    \@ifundefined{\mst@tmp@enc\string\i}%
        {%
          \PackageWarningNoLine{mathastext}
          {Impossible to define the \string\inodot\space using a font with\MessageBreak
           \mst@tmp@enc\space encoding, as there is no \string\i.\MessageBreak
           Reverting to default \string\imath\space in math version #4}%
          \let#1\relax
        }%
        {%
          \DeclareMathSymbol{#1}{\mathalpha}{mtletterfont}
                            {\csname\mst@tmp@enc\string\i\endcsname}%
        }%
    \@ifundefined{\mst@tmp@enc\string\j}%
        {%
          \PackageWarningNoLine{mathastext}
          {Impossible to define the \string\jnodot\space using a font with\MessageBreak
           \mst@tmp@enc\space encoding, as there is no \string\j.\MessageBreak
           Reverting to default \string\jmath\space in math version #4}%
          \let#2\relax
        }%
        {%
          \DeclareMathSymbol{#2}{\mathalpha}{mtletterfont}
                            {\csname\mst@tmp@enc\string\j\endcsname}%
        }%
  \fi
}%
\ifmst@defaultimath\else\mst@infoline{\string\imath\space and \string\jmath\space}
    \AtEndOfPackage{\AtBeginDocument{%
        \protected\def\imath{\inodot}%
        \protected\def\jmath{\jnodot}%
    }}%
\fi
```

`math accents` *Obsolete comments relative to the 2011 code:*

> I don't know how to get from the encoding to the slot positions of the accents (apart from going to look at all possible encodings definition files and putting this info here). In standard LaTeX, the math accents are taken from the 'operators' font. So we do the same here. Of course there is the problem that the user can define math versions with different encodings. Here I take T1 if it was the default at the time of loading the package, else OT1. `1.12b`: I add LY1 which is quasi like OT1.

At `1.3u` `2019/08/20` I decide to remove the hard-coded slot positions for OT1, T1 and LY1, and replace them with some hack which assumes LaTeX2e way of handling text accents got executed by the encoding definition file. If not, some breakage on package loading could occur, but this whole thing is conditional on the `mathaccents` option anyway, which per default is not executed.

Added note at `1.4d` `2024/10/26`: the above mentioned breakage definitely occurs with non-text encoding such as `OML`, if an attempt is made via `\Mathastext` or `\MTDeclareVersion` to set it up for use in a **`mathastext`** math version. This happens only with `mathaccents` option, and was shadowed by a similar breakage occurring without any option due to similar code added at `1.3u` for the `\hbar`, `\imath`, `\jmath` and even the `-` which tries to use the EN-DASH slot in an 8bit TeX font encoding. Fixed at `1.4` which emits warnings in such exotic contexts.

The `\vec` accent is not considered here because it has no suitable available glyph in a standard 8bits text font encodings.

Also at `1.3u` the math accents adapt to the font encoding at each math version.

(obsolete at `1.4`) `1.3v` adapts to LaTeX `2019-10-01` which now comes with robust math accent macros. The «original»-named macros are without the robustifying space (NOT true anymore, see `1.3w` next), as they only serve as meaning holders.

(obsolete at `1.4`) On the other hand the macros indexed by math version names are (in the pdflatex branch) always defined via `\DeclareMathAccent` hence they will be robust with `2019-10-01` or later and we must use the `\mst@robustifyingspace` with them to access their real meaning (this thus differs from the situation with `\hbar`).

`1.3w` The above was a bit optimistic as `amsmath` for example modifies LaTeX internals and handles math accents differently.

We thus needed to double our `\let`'s as, if `amsmath` is loaded, the cs with space will exist but not be paired in expected way with the original cs. This breaks things by the way if some math accent is written to an external file under a certain context and executed in another context. The new context will be probably ignored if `amsmath` is loaded, as the external file will have an already expanded-once meaning.

Some macros with space in name might thus be created as `\relax`. Should I rather create `\protected` macros for the math accents with Unicode engines? Anyway, the construct does give good result with the few OpenType text fonts I tested.

```
\let\mst@subduedmathaccents\@empty
\let\mst@nonsubduedmathaccents\@empty
\ifmst@mathaccents
\def\mst@subduedmathaccents{%
  \@tfor\@tempa:={grave}{acute}{check}{breve}{bar}%
                 {dot}{ddot}{mathring}{hat}{tilde}%
  \do
  {\expandafter\let\csname\@tempa\expandafter\endcsname
                   \csname mst@original@\@tempa\endcsname
   \expandafter\let\csname\@tempa\space\expandafter\endcsname
                   \csname mst@original@\@tempa\space\endcsname
  }%
}%
```

Modified at `1.4d` to catch math versions trying to use a non-text encoding such as `OML`.

```
\def\mst@nonsubduedmathaccents{%
 \@ifundefined{mst@grave@mv\math@version}
 {\mst@subduedmathaccents}
 {%
  \@tfor\@tempa:={grave}{acute}{check}{breve}{bar}%
                 {dot}{ddot}{mathring}{hat}{tilde}%
  \do
  {\expandafter\let\csname\@tempa\expandafter\endcsname
                   \csname mst@\@tempa @mv\math@version\endcsname
   \expandafter\let\csname\@tempa\space\expandafter\endcsname
                   \csname mst@\@tempa @mv\math@version\space\endcsname
  }%
 }%
}%
\def\mst@dothe@mathaccentsstuff#1#2{%
  \begingroup
  \edef\mst@tmp@enc{#2}%
```

```
\def\@text@composite##1\@text@composite##2{##2}%
\let\add@accent\@firstoftwo
\let\add@unicode@accent\@firstoftwo
\if1\mst@OneifUniEnc
```

Incredibly 1.4 removed the definition of a certain `\mst@robustifyingspace` but kept on using it here. The author add probably searched for a related conditional, but not for the macro itself... and there was a test file which would have shown the regression but it was not compiled.

```
  \ifmst@unimathaccents
    \expandafter\xdef\csname mst@grave@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\`\endcsname{}\relax}%
    \expandafter\xdef\csname mst@acute@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\'\endcsname{}\relax}%
    \expandafter\xdef\csname mst@check@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\v\endcsname{}\relax}%
    \expandafter\xdef\csname mst@breve@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\u\endcsname{}\relax}%
    \expandafter\xdef\csname mst@bar@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\=\endcsname{}\relax}%
    \expandafter\xdef\csname mst@dot@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\.\endcsname{}\relax}%
    \expandafter\xdef\csname mst@ddot@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\"\endcsname{}\relax}%
    \expandafter\xdef\csname mst@mathring@mv#1 \endcsname
     {\mst@Umathaccent
      7
      \number\symmtoperatorfont\space
      \csname#2\string\r\endcsname{}\relax}%
    \expandafter\xdef\csname mst@hat@mv#1 \endcsname
```

```
       {\mst@Umathaccent
        7
        \number\symmtoperatorfont\space
        \csname#2\string\^\endcsname{}\relax}%
      \expandafter\xdef\csname mst@tilde@mv#1 \endcsname
       {\mst@Umathaccent
        7
        \number\symmtoperatorfont\space
        \csname#2\string\~\endcsname{}\relax}%
    \else
```

1.3u used some `\def` but this made the accent macro meanings look slightly different depending on whether the math version being set-up was with an 8bit encoding or TU encoding.

For the sake of uniform treatment we modify this at 1.3v, but this is a bit complicated regarding timing: we need, in absence of `unimathaccents` option, in math versions with an OpenType font, to let the `\acute` etc... acquire back some prior non-**mathastext** meanings. To allow maximal flexibility, these original meaning get stored at begin document only. But `\mst@nonsubduedmathaccents` assigns to `\acute` etc... (in the robust sense with LaTeX `2019-10-01` or later) the meaning stored in the macros with the math version in their names. Such `\mst@acute@mvnormal` etc... must thus be ready before `\mst@nonsubduedmathaccents` (or at least before the last such) execution: the code here must get executed after the definition of the «original»-named macros but prior to the (last one, if multiple) `\mst@nonsubduedmathaccents`.

Hence 1.3v delayed a bit the initial execution of this macro (see further down in the code) compared to what happened in 1.3u.

We are in a group but `\AtEndOfPackage` does the right thing.

```
      \AtEndOfPackage{\AtBeginDocument{%
      \@tfor\@tempa:={grave}{acute}{check}{breve}{bar}%
                     {dot}{ddot}{mathring}{hat}{tilde}%
      \do
      {\expandafter\let
          \csname mst@\@tempa @mv#1\expandafter\endcsname
          \csname mst@original@\@tempa\endcsname
       \expandafter\let
          \csname mst@\@tempa @mv#1\space\expandafter\endcsname
          \csname mst@original@\@tempa\space\endcsname}%
      }}%
    \fi
```

This is needed because the pdflatex engine branch will use `\DeclareMathAccent` and it creates robust macros with LaTeX `2019-10-01` or later. As we want elsewhere in the package code not to have to check if under Unicode engine or not, we need to handle here also some definition of robust macros.

But wouldn't it be simpler to manage `\protected` macros?

```
      \@tfor\@tempa:={grave}{acute}{check}{breve}{bar}%
                     {dot}{ddot}{mathring}{hat}{tilde}%
      \do
      {\expandafter\xdef\csname mst@\@tempa @mv#1\endcsname
         {\noexpand\protect
          \expandafter\noexpand\csname mst@\@tempa @mv#1 \endcsname}%
      }%
```

```
  \else
```

`\DeclareMathAccent` works `\globally`. And with LaTeX `2019-10-01` or later it creates robust macros.

`\mst@DeclareMathAccent` to work around https://github.com/latex3/latex2e/issues/216

`1.4d` adds steps to avoid a crash if user is trying to set up a `mathastext` math version with an encoding such as `OML`.

```
   \@ifundefined{#2\string\`}
      {\PackageWarningNoLine{mathastext}
         {Impossible to pick up text accents from an #2\MessageBreak
          encoded font.  The mathaccent option will be\MessageBreak
          ignored for the math version #1}%
       \let\mst@tempa0}
      {\let\mst@tempa1}%
   \if\mst@tempa1%
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@grave@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\`\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@acute@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\'\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@check@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\v\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@breve@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\u\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@bar@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\=\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@dot@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\.\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@ddot@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\"\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@mathring@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\r\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@hat@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\^\endcsname{}}
    \expandafter\mst@DeclareMathAccent\expandafter
               {\csname mst@tilde@mv#1\endcsname}{\mathalpha}{mtoperatorfont}%
               {\csname#2\string\~\endcsname{}}
   \else
```

The math version can be declared multiple times we can not simply do some `\@nameuse` here.

```
    \global\expandafter\let\csname mst@grave@mv#1\endcsname\relax
```

```
   \fi
  \fi
  \endgroup
}%
\fi
```

\MTDeclareVersion The `\MTDeclareVersion` command is to be used in the preamble to declare a math version. A more complicated variant would also specify a choice of series for the Euler and Symbol font: anyhow Symbol only has the medium series, and Euler has medium and bold, so what is lacking is the possibility to create a version with a bold Euler. There is already one such version: the default **`bold`** one. And there is always the possibility to add to the preamble **`\SetSymbolFont{mteulervm}{versionname}{U}{zeur}{bx}{n}`** if one wants to have a math version with bold Euler characters.

For version `1.1` we add an optional parameter specifying the shape to be used for letters.

Note: (2012/10/24) I really should check whether the user attempts to redefine the 'normal' and 'bold' versions and issue a warning in that case! Finally done at `1.3w` `2019/11/16`! Better late than never...

`1.3c` (2013/12/14) adds an extra optional parameter after all previous ones, to inherit the settings from another version. Typically to be used with [bold]. I take this opportunity to sanitize a bit some line endings to avoid generating (in the preamble, document macros were already careful of course) too many space tokens, at least inside macros. And I modifiy (correct? perhaps it was on purpose) the strange way I used **`\@onlypreamble`** in earlier version.

`1.3u` adds storage of macros holding the needed meanings for **`\imath`**, **`\hbar`**, math accents, and the minus symbol, version wise.

`1.3w` adds the check to forbid **`normal`** and **`bold`** as version names.

```
\newcommand*\MTDeclareVersion[6][]{%
  \edef\mst@declareversionargs{{#1}{#2}{#3}{#4}{#5}{#6}}%
  \edef\mst@version{#2}%
  \if0\ifx\mst@version\mst@normalversionname0\else
      \ifx\mst@version\mst@boldversionname0\else
      1\fi\fi
   \expandafter\expandafter\expandafter
   \MTDoNotDeclareVersion@\expandafter\@gobblefour
  \fi
  \relax\DeclareMathVersion{\mst@version}\MTDeclareVersion@
}%
\newcommand*\MTDoNotDeclareVersion@[1][]{%
   \PackageWarningNoLine{mathastext}{You asked to declare a version with name
   `\mst@version'.^^J%
   \@spaces Sorry but you are not allowed to do that.^^J%
   \@spaces \ifmst@subdued Anyway the `subdued' option is in force\else
         Use rather \string\Mathastext\space macro (with no optional argument)\fi
}}%
\newcommand*\MTDeclareVersion@[1][]{%
    \edef\mst@tmp{#1}%
    \ifx\mst@tmp\empty\else
      \global\expandafter\let\csname mv@\mst@version\expandafter\endcsname
                             \csname mv@#1\endcsname
      \PackageInfo{mathastext}
```

```
                  {Math version `\mst@version\string'\MessageBreak
                   declared\on@line\MessageBreak
                   inherits from `#1\string'\@gobble}%
    \fi
    \expandafter\MTDeclareVersion@@\mst@declareversionargs
}%
\newcommand*\MTDeclareVersion@@[6]{%
  \expandafter\edef\csname mst@encoding@\mst@version\endcsname{#3}%
  \expandafter\edef\csname mst@family@\mst@version\endcsname{#4}%
  \expandafter\edef\csname mst@series@\mst@version\endcsname{#5}%
  \expandafter\edef\csname mst@shape@\mst@version\endcsname{#6}%
  \expandafter\edef\csname mst@boldvariant@\mst@version\endcsname{\mst@bold}%
  \expandafter\edef\csname mst@itdefault@\mst@version\endcsname{\itdefault}%
  \expandafter\edef\csname mst@rmdefault@\mst@version\endcsname{\rmdefault}%
  \expandafter\edef\csname mst@sfdefault@\mst@version\endcsname{\sfdefault}%
  \expandafter\edef\csname mst@ttdefault@\mst@version\endcsname{\ttdefault}%
  \expandafter\edef\csname mst@exists@skip@\mst@version\endcsname
        {\mst@exists@skip}%
  \expandafter\edef\csname mst@forall@skip@\mst@version\endcsname
        {\mst@forall@skip}%
  \expandafter\edef\csname mst@prime@skip@\mst@version\endcsname
        {\mst@prime@skip}%
  \def\mst@tmp{#1}%
  \ifx\mst@tmp\empty
    \ifmst@italic
      \SetSymbolFont{mtletterfont}{#2}{#3}{#4}{#5}{\mst@ltsh}%
      \immediate\write\m@ne{}%
      \PackageInfo{mathastext}%
                        {\ifmst@noletters
                         \else
                         Latin letters in math version `#2\string'\MessageBreak
                         declared\on@line\MessageBreak
                         will use the font #3/#4/#5/\mst@ltsh
                         \ifmst@frenchmath\space(uppercase: #6)\fi\MessageBreak
                         \fi
                         Other characters (digits, ...) and \protect\log-like names\Messa
                         \ifmst@noletters
                         will use the font #3/#4/#5/#6%
                         \else
                         will be in `#6\string' shape%
                         \fi\@gobble}%
      \immediate\write\m@ne{}%
      \expandafter\edef\csname mst@ltshape@\mst@version\endcsname{\mst@ltsh}%
    \else
      \SetSymbolFont{mtletterfont}{#2}{#3}{#4}{#5}{#6}%
      \ifmst@noletters
      \else
      \immediate\write\m@ne{}%
      \PackageInfo{mathastext}%
```

```
                      {Latin letters in math version `#2\string'\MessageBreak
                       declared\on@line\MessageBreak
                       will use the fonts #3/#4/#5(\mst@bold)/#6\@gobble}%
    \fi
    \immediate\write\m@ne{}%
    \expandafter\edef\csname mst@ltshape@\mst@version\endcsname{#6}%
  \fi
\else
```

#1 not empty.

```
  \SetSymbolFont{mtletterfont}{#2}{#3}{#4}{#5}{#1}%
  \immediate\write\m@ne{}%
  \PackageInfo{mathastext}%
                      {\ifmst@noletters
                       \else
                       Latin letters in math version `#2\string'\MessageBreak
                       declared\on@line\MessageBreak
                       will use the font #3/#4/#5/#1%
                       \ifmst@frenchmath\space(uppercase: #6)\fi\MessageBreak
                       \fi
                       Other characters (digits, ...) and
                       \protect\log-like names\MessageBreak
                       \ifmst@noletters
                       will use the font #3/#4/#5/#6%
                       \else
                       will be in `#6\string' shape%
                       \fi\@gobble}%
  \immediate\write\m@ne{}%
  \expandafter\edef\csname mst@ltshape@\mst@version\endcsname{#1}%
\fi
```

Here and elsewhere 1.3za has removed an `\ifmst@nonormalbold` conditional.

```
     \SetMathAlphabet{\Mathnormalbold}{#2}{#3}{#4}{\mst@bold}%
                 {\csname mst@ltshape@\mst@version\endcsname}%
\SetSymbolFont{mtoperatorfont}{#2}{#3}{#4}{#5}{#6}%
```

Since 1.3za (and prior to 1.15f) these math alphabet commands are always defined.

```
\SetMathAlphabet{\Mathbf}{#2}{#3}{#4}{\mst@bold}{#6}
\SetMathAlphabet{\Mathit}{#2}{#3}{#4}{#5}{\itdefault}
\SetMathAlphabet{\Mathsf}{#2}{#3}{\sfdefault}{#5}{#6}
\SetMathAlphabet{\Mathtt}{#2}{#3}{\ttdefault}{#5}{#6}
\ifmst@needeuler
    \SetMathAlphabet{\MathEulerBold}{#2}{U}{zeur}{\mst@bold}{n}%
\fi
```

`LGRgreeks` `selfGreeks` In the case of option LGRgreeks (selfGreeks), it is expected that the fonts used in each math versions exist in LGR (OT1) encoding. We first recalculate the shapes to be used for lowercase and uppercase Greek letters depending on the frenchmath and [it/up][g/G]reek options as well as on the (local to this version) shapes for letters and digits.

1.3y replaces `\updefault` by `\MTgreekupdefault` and `\itdefault` by `\MTgreekitdefault`. It also prepares to store two Boolean settings saying whether lowercase respectively uppercase

Greek letters will use ‘upright’ or ‘italic’ (`LGRgreek(s)` only).

The 1.3y refactoring of `LGRgreek` is done via a decoupling, thus things are done here under `selfGreek` or other Greek options which ultimately serve nothing and conversely things are done here for `LGRgreek` which are superfluous.

At 1.4e, the code previously used here was replaced with a call to a slightly modified `\mst@update@greeksh` to share code. Formerly some `\mst@greek@ush@loc` and `\mst@greek@lsh@loc` were defined here but this appears to have been due only to extra cautious coding to avoid overwriting something else. On closer examination (after a great many years) this swas not needed at all.

```
  \mst@update@greeksh
     {\csname mst@ltshape@\mst@version\endcsname}%
     {\csname mst@shape@\mst@version\endcsname}%
```

1.3za refactoring to reduce code duplication; I briefly considered trying to enhance `\MTgreekfont` to work also with `LGRgreeks` and `selfGreeks` but I have dropped the idea for now.

```
    \ifmst@LGRgreeks \def\mst@greekfont{#4}\fi
    \ifmst@selfGreeks\def\mst@greekfont{#4}\fi
    \ifmst@LGRgreek
      \SetSymbolFont{mtgreekup}{#2}{LGR}{\mst@greekfont}{#5}{\MTgreekupdefault}%
      \SetSymbolFont{mtgreekit}{#2}{LGR}{\mst@greekfont}{#5}{\MTgreekitdefault}%
      \SetMathAlphabet{\mathgreekupbold}{#2}{LGR}{\mst@greekfont}
                                                 {\mst@bold}{\MTgreekupdefault}%
      \SetMathAlphabet{\mathgreekitbold}{#2}{LGR}{\mst@greekfont}
                                                 {\mst@bold}{\MTgreekitdefault}%
```

This is where the shape of uppercase/lowercase Greek letters is recorded, for `\MTversion`’s triggered `\MTcustomgreek` to do the right thing.

```
      \expandafter\let\csname ifmst@greek@\mst@version @upper@up\expandafter\endcsname
                      \csname ifmst@greek@upper@up\endcsname
      \expandafter\let\csname ifmst@greek@\mst@version @lower@up\expandafter\endcsname
                      \csname ifmst@greek@lower@up\endcsname
\immediate\write\m@ne{}%
\PackageInfo{mathastext}{Greek letters (upper:
      `\ifmst@greek@upper@up\MTgreekupdefault\else\MTgreekitdefault\fi\string',
                                 lower:
      `\ifmst@greek@lower@up\MTgreekupdefault\else\MTgreekitdefault\fi\string')\MessageBr
                         will use font family `\mst@greekfont\string' (LGR)\MessageBreak
                         in mathastext
                         math version `\mst@version\string'\MessageBreak
                         declared}%
\immediate\write\m@ne{}%
    \else
     \ifmst@selfGreek
       \SetSymbolFont{mtselfGreekfont}{#2}{OT1}{\mst@greekfont}{#5}{\mst@greek@ush}%
\immediate\write\m@ne{}%
\PackageInfo{mathastext}{Capital Greek letters (shape `\mst@greek@ush\string'
                           will use the font\MessageBreak
                         family `\mst@greekfont\string' (OT1) in mathastext\MessageBreak
                         math version `\mst@version\string' declared}%
\immediate\write\m@ne{}%
```

```
     \fi
    \fi
```

1.4d adds here a check for the `nohbar` option. Presumably this was not done before because anyhow the package does

`\let\mst@subduedhbar\@empty\let\mst@nonsubduedhbar\@empty`

But it is done in `\Mathastext@`. And besides, for the `-` the code does check the `endash` et al. options.

```
  \ifmst@nohbar\else
    \edef\mst@tmp{\expandafter\noexpand\csname mst@hbar@mv#2\endcsname
                  \expandafter\noexpand\csname mst@ltbar@mv#2\endcsname}%
    \expandafter\mst@dothe@hbarstuff\mst@tmp{#3}{#3}%
  \fi
  \edef\mst@tmp{\expandafter\noexpand\csname mst@inodot@mv#2\endcsname
                \expandafter\noexpand\csname mst@jnodot@mv#2\endcsname}%
```

1.4d lets the option `noletters` prevent definition of `\inodot` and `\jnodot`.

```
  \ifmst@noletters\else
    \expandafter\mst@dothe@inodotstuff\mst@tmp{#3}{#2}%
  \fi
  \ifmst@mathaccents
    \mst@dothe@mathaccentsstuff{#2}{#3}%
  \fi
  \edef\mst@tmp{\expandafter\noexpand\csname mst@minus@mv#2\endcsname
                \expandafter\noexpand\csname mst@varfam@minus@mv#2\endcsname}%
  \ifmst@endash
    \expandafter\mst@dothe@endashstuff\mst@tmp{#3}{#2}%
  \else
    \ifmst@emdash
      \expandafter\mst@dothe@emdashstuff\mst@tmp{#3}{#2}%
    \else
      \expandafter\mst@dothe@hyphenstuff\mst@tmp
    \fi
  \fi
}%
\let\MathastextDeclareVersion\MTDeclareVersion
```

`\MTversion`
`\MTversion@`
`\MTversion@s`
`\MTversion@@`

This is a wrapper around LaTeX's `\mathversion`: here we have an optional argument allowing a quick and easy change of the text fonts additionally to the math fonts. Present already in the initial version of the package (January 2011.)

1.15: some modifications for the subdued option vs LGRgreek and for the math muskips after `\exists` and `\forall`.

1.2: with the subdued option sets the math alphabets in the normal and bold math versions do not apply to operator names and non-alphabetical symbols. The switch for braces is left as it is.

1.2b: with the subdued option, the italic corrections are not added. Else, we check the shape of letters in this version. Also, there was a bug since 1.15: the values of the math skips were taken not from the settings for the math version (#2) but from those of the optional argument (#1), if present...

1.3: activation of italic corrections is now separated from actual math activation of letters.

1.3c: a starred variant is added which does not modify the text fonts, only the math set-tup.

1.3d: replaced in `\MTversion@` things like `\edef\mst@encoding{...}` and

`\renewcommand{\encodingdefault}{\mst@@encoding}`

by `\edef\encodingdefault{...}` etc... All those `\mst@@...` things were useless. I also redefine `\seriesdefault` rather than `\mddefault`.

1.3d: mechanism of restoration of Greek in subdued normal and bold versions has been to all cases, and not only for the `LGRgreek` option.

1.3u: version savvy (i.e. font-encoding savvy) minus sign, `\hbar`, `\imath`, math accents.

1.3y: Booleans recovered from stored data in the math version will configure the things `\MTcustomgreek` do, under `LGRgreek` option.

```
\newcommand*\MTversion {\@ifstar\MTversion@s\MTversion@}
\newcommand*\MTversion@s [1]{\mathversion{#1}\MTversion@@}
\newcommand*\MTversion@ [2][]{%
```

This defines `\math@version` as expanded `#2`.

```
    \mathversion{#2}%
    \edef\mst@tmpa{#1}%
    \ifx\mst@tmpa\empty
        \let\mst@tmp\math@version
    \else
        \let\mst@tmp\mst@tmpa
    \fi
    \edef\encodingdefault {\csname mst@encoding@\mst@tmp\endcsname}%
    \edef\familydefault   {\csname mst@family@\mst@tmp\endcsname}%
    \edef\seriesdefault   {\csname mst@series@\mst@tmp\endcsname}%
    \edef\shapedefault    {\csname mst@shape@\mst@tmp\endcsname}%
    \edef\bfdefault {\csname mst@boldvariant@\mst@tmp\endcsname}%
    \edef\itdefault {\csname mst@itdefault@\mst@tmp\endcsname}%
    \edef\rmdefault {\csname mst@rmdefault@\mst@tmp\endcsname}%
    \edef\sfdefault {\csname mst@sfdefault@\mst@tmp\endcsname}%
    \edef\ttdefault {\csname mst@ttdefault@\mst@tmp\endcsname}%
    \usefont{\encodingdefault}{\familydefault}{\seriesdefault}{\shapedefault}%
    \MTversion@@
}%
```

1.3j has a stronger subdued which does `\MTnormalprime`, `\MTnormalexists`, `\MTnormalforall` rather than setting the skips to `0mu`. Hence `\MTversion` by default should do `\MTprimedoesskip`, `\MTexistsdoesskip`, `\MTforalldoesskip`.

1.3u drops the argument, as the info is in `\math@version` from LaTeX2e code.

```
\newcommand*\MTversion@@ {%
    \MTexistsdoesskip
    \MTforalldoesskip
    \MTprimedoesskip
```

`v1.15e`: muskips.

```
     \mst@exists@muskip\csname mst@exists@skip@\math@version\endcsname\relax
     \mst@forall@muskip\csname mst@forall@skip@\math@version\endcsname\relax
```

`v1.2`: muskip for `\prime`.

```
     \mst@prime@muskip\csname mst@prime@skip@\math@version\endcsname\relax
```

`v1.2b`: italic corrections except for italic/slanted (sic) letters, and of course except in the subdued normal and bold math versions.

`v1.3`: by default, letters are made mathematically active, even if italic corrections are not used, to allow the action of `\MTsetmathskips`.

```
    \edef\mst@tmpa{\csname mst@ltshape@\math@version\endcsname}%
    \edef\mst@tmpb{\csname mst@shape@\math@version\endcsname}%
```

`1.15c`: extending subdued to LGRgreek.

`1.15f`: subdueing math alphabets in a simpler way than in `1.15e`.

`1.2b`: subdueing the activation of characters in math mode.

`1.2d`: special treatment of the asterisk.

`1.3d`: extended LGRgreek mechanism of activation/restoration of Greek to all cases.

`1.3j`: use of `\MTeverymathdefault`, which includes `\MTicinmath`, but must be corrected then according to shape of letters and presence or absence of option `frenchmath`. We do only `\def\mst@ITcorr{\ifnum\fam=\m@ne\/\fi}` and not `\MTICinmath` to not overwrite some user-defined `\MTeverymathdefault`. Code for italic corrections or not according to letter shape is executed after `\MTeverymathdefault` which limits a bit user customizing possibilities, but if I moved it later, I would possibly have to put inside the `\MTicinmath` the check for `it` ot `sl`. Similary the `\MTcustomgreek` always executed (if not `subdued`).

MEMO: `\MTeverymathdefault` is executed *also* if in `subdued` mode but there is a `\MTeverymathoff` done next in the *normal* and *bold* version. It does `\MTicinmath` and `\MTmathactivedigits` which however are no-ops (only partly for the former, and for the latter always anyhow if no option `activedigits`) in subdued mode.

```
    \MTmathoperatorsobeymathxx
    \MTeverymathdefault
    \MTcustomizenewmcodes
    \@for\mst@tmpc:=it,sl\do{\ifx\mst@tmpc\mst@tmpa\MTnoicinmath\fi}%
    \ifmst@frenchmath
      \def\mst@ITcorr{\ifnum\fam=\m@ne\/\fi}%
      \@for\mst@tmpc:=it,sl\do{\ifx\mst@tmpc\mst@tmpb\MTnoICinmath\fi}%
    \fi
```

`1.3j` has a stronger subdued which does `\MTnormalprime`, `\MTnormalexists`, `\MTnormalforall` rather than simply setting the skips to `0mu`. Note: `\MTnormalprime` is done as part of `\MTeverymathoff`.

The `subdued` mode does *not* undo the effect of the `frenchmath` option on uppercase Latin letters: they will use the same shape as digits and operator names! (This should have been made more prominent in user manual more than ten years ago, but is done only today 2023/12/28...).

```
    \ifmst@subdued
      \ifx\math@version\mst@normalversionname
        \mst@restorealphabets
        \MTstandardgreek
        \MTmathoperatorsdonotobeymathxx
        \MTnormalexists
        \MTnormalforall
```

`1.4` has kept `\MTmathstandardletters` inside `\MTeverymathoff` but its action is now quite different from earlier situation as it resets mathcodes from active to normal status on the spot.

```
        \MTeverymathoff
        \MTresetnewmcodes
```

1.3t adds better compatibility with `subdued` mode for `\imath/\jmath` and perfect compatibility for the minus sign.

1.3u extends this further to allow per-math-version meanings for them.

1.4c removes usage here of `\mst@subduedminus` (which anyhow is empty under `everymath`) because `\MTeverymathoff` does `\MTnonlettersdonotobeymathxx` which executes the `\mst@undo@nonletters` `\toks` hence ends up executing `\mst@subduedminus`. On second thoughts not so sure good idea to have removed it because documentation mentions `\MTeverymathoff` and sadly suggests redefining it for some effects.

```
  \mst@subduedhbar
  \mst@subduedinodot
  \mst@subduedmathaccents
 \else
  \ifx\math@version\mst@boldversionname
   \mst@restorealphabets
   \MTstandardgreek
   \MTmathoperatorsdonotobeymathxx
   \MTnormalexists
   \MTnormalforall
   \MTeverymathoff
   \MTresetnewmcodes
   \mst@subduedhbar
   \mst@subduedinodot
   \mst@subduedmathaccents
  \else
   \mst@setalphabets
```

1.3y addition for `\MTcustomgreek` under `LGRgreeks` option.

MEMO: the needed mathematical re-activation of letters when switching from *normal* or *bold* to a non-`subdued` math version has already been done above from the `\MTicinmath` which is part of `\MTeverymathdefault`.

MEMO: idem for digits under option `activedigits`.

```
\expandafter\let\csname ifmst@greek@upper@up\expandafter\endcsname
        \csname ifmst@greek@\math@version @upper@up\endcsname
\expandafter\let\csname ifmst@greek@lower@up\expandafter\endcsname
        \csname ifmst@greek@\math@version @lower@up\endcsname
   \MTcustomgreek
   \mst@nonsubduedhbar
   \mst@nonsubduedinodot
   \mst@nonsubduedmathaccents
   \mst@nonsubduedminus
  \fi
 \fi
\else
```

1.3y addition for `\MTcustomgreek` under `LGRgreek` option.

MEMO: the mathematical activation of letters happened above from `\MTeverymathdefault`. Idem if `activedigits` for digits.

Sadly the 1.3y addition broke usage of (non-`subdued` `\MTversion{normal}` under option `LGRgreek`, as the `\ifmst@greek@normal@upper@up` used here never got any definition, hence

`\ifmst@greek@upper@up` got set to `\relax` which caused down the road an `Extra \else` error in `\MTcustomgreek`.

```
    \expandafter\let\csname ifmst@greek@upper@up\expandafter\endcsname
                    \csname ifmst@greek@\math@version @upper@up\endcsname
    \expandafter\let\csname ifmst@greek@lower@up\expandafter\endcsname
                    \csname ifmst@greek@\math@version @lower@up\endcsname
      \MTcustomgreek
      \mst@nonsubduedhbar
      \mst@nonsubduedinodot
      \mst@nonsubduedmathaccents
      \mst@nonsubduedminus
    \fi
}%
\let\MathastextVersion\MTversion
\let\Mathastextversion\MTversion
\let\MTVersion\MTversion
\let\mathastextversion\MTversion
```

`\MTWillUse` This is a preamble-only command, which can be used more than once, only the latest one counts. Sets up the math fonts in the normal and bold versions, as does `\Mathastext`.

```
\newcommand*\MTWillUse[5][]{
  \MTencoding{#2}
  \MTfamily{#3}
  \MTseries{#4}
  \MTshape{#5}
    \ifmst@italic\MTlettershape{\itdefault}\fi
  \edef\mst@tmp{#1}
  \ifx\mst@tmp\empty\else\MTlettershape{#1}\fi
  \Mathastext}
\let\MathastextWillUse\MTWillUse
\let\Mathastextwilluse\MTWillUse
```

`\Mathastext` The command `\Mathastext` can be used anywhere in the preamble and any number of time, the last one is the one that counts.

In version 1.1 we have two fonts: they only differ in shape. The `mtletterfont` is for letters, and the `mtoperatorfont` for digits and log-like operator names. The default is that both are upright.

Starting with version 1.12, an optional argument makes `\Mathastext` act as the declaration of a math version, to be later used in the document.

Versions 1.15x brought some adaptations related to the subdued option.

1.3c adds a second optional parameter to inherit previous settings from another version; mostly done to inherit the bold version fonts for symbols and large symbols. This is done in `\MTDeclareVersion`.

1.3j moves the code related to `\MTicinmath` from `\Mathastext@` to `\AtBeginDocument` (code depending on whether `subdued` option in use). But we omit for this from `\MTicinmath` the `\MTmathactiveletters` and issue the latter during loading of package, hence allowing `\MTmathstandardletters` to be effective in the preamble.

I forgot to document that under `subdued` option the `\Mathastext` command without optional parameter does not any `\SetSymbolFont` etc... but it has a few other tasks to complete nevertheless.

`1.3u` fixes some long-standing bug that `\Mathastext` did not repeat some font-encoding dependent things: they got done only once during package loading (things regarding the `\hbar`, `\imath`, the math accents and the minus sign). They are now part of the contents of `\Mathastext` macro itself (which is executed during package loading).

`1.3y` has refactored the `LGRgreek` associated math fonts.

```
\def\Mathastext {\@ifnextchar[\Mathastext@declare\Mathastext@}% ]
\def\Mathastext@declare [#1]{%
  \edef\mst@tmp{#1}%
  \ifx\mst@tmp\empty
       \expandafter\@firstoftwo
  \else\expandafter\@secondoftwo
  \fi
  \Mathastext@
  {\MTDeclareVersion[\mst@ltsh]{#1}{\mst@enc}{\mst@fam}{\mst@ser}{\mst@opsh}}%
}%
```

This is thus the macro which is triggered by usage of `\Mathastext` in the preamble without optional argument. The package will execute it once after its definition.

```
\def\Mathastext@ {%
```

This `\mst@update@greeksh` redefines `\mst@greek@ush` (used below in a `\SetSymbolFont` executed if option `selfGreek`) and sets the `\ifmst@greek@upper@up` and `\ifmst@greek@lower@up` conditionals.

```
  \mst@update@greeksh {\mst@ltsh}{\mst@opsh}%
```

`1.4e` fixes belatedly a bug introduced at `1.3y`, which had broken usage of `\MTversion{normal}` in documents using `LGRgreek` or `LGRgreeks` (but not if `subdued`).

```
      \expandafter\let\csname ifmst@greek@normal@upper@up\expandafter\endcsname
                      \csname ifmst@greek@upper@up\endcsname
      \expandafter\let\csname ifmst@greek@normal@lower@up\expandafter\endcsname
                      \csname ifmst@greek@lower@up\endcsname
      \expandafter\let\csname ifmst@greek@bold@upper@up\expandafter\endcsname
                      \csname ifmst@greek@upper@up\endcsname
      \expandafter\let\csname ifmst@greek@bold@lower@up\expandafter\endcsname
                      \csname ifmst@greek@lower@up\endcsname
  \edef\mst@encoding@normal{\mst@enc}%
  \edef\mst@family@normal{\mst@fam}%
  \edef\mst@series@normal{\mst@ser}%
  \edef\mst@shape@normal{\mst@opsh}%
  \edef\mst@ltshape@normal{\mst@ltsh}%
  \edef\mst@itdefault@normal{\itdefault}%
  \edef\mst@rmdefault@normal{\rmdefault}%
  \edef\mst@sfdefault@normal{\sfdefault}%
  \edef\mst@ttdefault@normal{\ttdefault}%
  \edef\mst@boldvariant@normal{\mst@bold}%
  \edef\mst@exists@skip@normal{\mst@exists@skip}%
  \edef\mst@forall@skip@normal{\mst@forall@skip}%
  \edef\mst@prime@skip@normal{\mst@prime@skip}%
```

```
\edef\mst@encoding@bold{\mst@enc}%
\edef\mst@family@bold{\mst@fam}%
\edef\mst@series@bold{\mst@bold}%
\edef\mst@shape@bold{\mst@opsh}%
\edef\mst@ltshape@bold{\mst@ltsh}%
\edef\mst@boldvariant@bold{\mst@bold}%
\edef\mst@itdefault@bold{\itdefault}%
\edef\mst@rmdefault@bold{\rmdefault}%
\edef\mst@sfdefault@bold{\sfdefault}%
\edef\mst@ttdefault@bold{\ttdefault}%
\edef\mst@exists@skip@bold{\mst@exists@skip}%
\edef\mst@forall@skip@bold{\mst@forall@skip}%
\edef\mst@prime@skip@bold{\mst@prime@skip}%
\ifmst@subdued
```

Since 1.3j this branch is actually almost superfluous, as entering `normal` or `bold` with `\MTversion` does `\MTnormalexists`, `\MTnormalforall`, and `\MTnormalprime`. But some default values are needed if the user insists on issuing `\MTexistsdoesskip`, etc... nevertheless.

```
  \def\mst@exists@skip@normal{0mu}%
  \def\mst@forall@skip@normal{0mu}%
  \def\mst@prime@skip@normal{0mu}%
  \def\mst@exists@skip@bold{0mu}%
  \def\mst@forall@skip@bold{0mu}%
  \def\mst@prime@skip@bold{0mu}%
\else
  \ifmst@italic
    \ifmst@frenchmath
      \mst@exists@muskip\mst@exists@skip\relax
      \mst@forall@muskip\mst@forall@skip\relax
      \mst@prime@muskip\mst@prime@skip\relax
    \else
     \def\mst@exists@skip@normal{0mu}%
     \def\mst@forall@skip@normal{0mu}%
     \def\mst@prime@skip@normal{0mu}%
     \def\mst@exists@skip@bold{0mu}%
     \def\mst@forall@skip@bold{0mu}%
     \def\mst@prime@skip@bold{0mu}%
    \fi
  \else
      \mst@exists@muskip\mst@exists@skip\relax
      \mst@forall@muskip\mst@forall@skip\relax
      \mst@prime@muskip\mst@prime@skip\relax
  \fi
\fi
```

Here and elsewhere 1.3za has removed usage of an `\ifmst@nonormalbold` conditional which was added at 1.15f.

```
 \SetMathAlphabet{\Mathnormalbold}{normal}{\mst@encoding@normal}%
                               {\mst@family@normal}%
                               {\mst@boldvariant@normal}%
                               {\mst@ltshape@normal}%
 \SetMathAlphabet{\Mathnormalbold}{bold}{\mst@encoding@bold}%
                               {\mst@family@bold}%
                               {\mst@boldvariant@bold}%
                               {\mst@ltshape@bold}%
\ifmst@subdued\else
 \SetSymbolFont{mtletterfont}{normal}{\mst@encoding@normal}%
                                     {\mst@family@normal}%
                                     {\mst@series@normal}%
                                     {\mst@ltshape@normal}%
 \SetSymbolFont{mtletterfont}{bold}  {\mst@encoding@bold}%
                                     {\mst@family@bold}%
                                     {\mst@series@bold}%
                                     {\mst@ltshape@bold}%
 \SetSymbolFont{mtoperatorfont}{normal}{\mst@encoding@normal}%
                                     {\mst@family@normal}%
                                     {\mst@series@normal}%
                                     {\mst@shape@normal}%
 \SetSymbolFont{mtoperatorfont}{bold} {\mst@encoding@bold}%
                                     {\mst@family@bold}%
                                     {\mst@series@bold}%
                                     {\mst@shape@bold}%
```

`1.3za` removes the `1.15f` added conditional checks.

```
  \SetMathAlphabet{\Mathbf}{normal}{\mst@encoding@normal}%
                               {\mst@family@normal}%
                               {\mst@series@bold}%
                               {\mst@shape@normal}%
  \SetMathAlphabet{\Mathbf}{bold}{\mst@encoding@bold}%
                               {\mst@family@bold}%
                               {\mst@series@bold}%
                               {\mst@shape@bold}%
  \SetMathAlphabet{\Mathit}{normal}{\mst@encoding@normal}%
                               {\mst@family@normal}%
                               {\mst@series@normal}%
                               {\mst@itdefault@normal}%
  \SetMathAlphabet{\Mathit}{bold}{\mst@encoding@bold}%
                               {\mst@family@bold}%
                               {\mst@series@bold}%
                               {\mst@itdefault@bold}%
  \SetMathAlphabet{\Mathsf}{normal}{\mst@encoding@normal}%
                               {\mst@sfdefault@normal}%
                               {\mst@series@normal}%
                               {\mst@shape@normal}%
  \SetMathAlphabet{\Mathsf}{bold}{\mst@encoding@bold}%
                               {\mst@sfdefault@bold}%
                               {\mst@series@bold}%
                               {\mst@shape@bold}%
  \SetMathAlphabet{\Mathtt}{normal}{\mst@encoding@normal}%
                               {\mst@ttdefault@normal}%
                               {\mst@series@normal}%
                               {\mst@shape@normal}%
  \SetMathAlphabet{\Mathtt}{bold}{\mst@encoding@bold}%
                               {\mst@ttdefault@bold}%
                               {\mst@series@bold}%
                               {\mst@shape@bold}%
\fi
```

`\MathEulerBold`

1.14c: We reset `mteulervm` and `\MathEulerBold` here as the variant for bold may have been changed by the user via `\Mathastextboldvariant{m}`; and we should keep this local to math versions.

```
\ifmst@needeuler
  \SetSymbolFont{mteulervm}{bold}{U}{zeur}{\mst@boldvariant@normal}{n}%
  \SetMathAlphabet{\MathEulerBold}{normal}%
             {U}{zeur}{\mst@boldvariant@normal}{n}%
  \SetMathAlphabet{\MathEulerBold}{bold}%
             {U}{zeur}{\mst@boldvariant@bold}{n}%
\fi
```

```
\ifmst@needsymbol\SetSymbolFont{mtpsymbol}{bold}%
                   {U}{psy}{\mst@boldvariant@bold}{n}%
\fi
```

`LGRgreek*`
`selfGreek*`

LGRgreek, LGRgreeks, selfGreek, and selfGreeks options.

1.3y has refactored the `LGRgreek` associated math fonts.

1.3za adds the math alphabets `\mathgreekitbold` and `\mathgreekupbold`. And it executes this code also in `subdued` mode, because anyhow the symbolfonts `mtgreekup` and `mtgreekit` and associated alphabets have been declared also, at time of loading the package, so not doing it here means that effect of `\MTgreekfont` would be ignored; which was probably a bug. And by the way, documentation says `\MTgreekfont` has no effect under `LGRgreeks` and `selfGreeks` option so we need to enforce it here (for time being).

```
 \ifmst@LGRgreeks \edef\mst@greekfont{\mst@fam}\fi
 \ifmst@selfGreeks\edef\mst@greekfont{\mst@fam}\fi
 \ifmst@LGRgreek
  \SetSymbolFont{mtgreekup}{normal}{LGR}%
          {\mst@greekfont}{\mst@series@normal}{\MTgreekupdefault}%
  \SetSymbolFont{mtgreekup}{bold}{LGR}%
          {\mst@greekfont}{\mst@boldvariant@bold}{\MTgreekupdefault}%
  \SetSymbolFont{mtgreekit}{normal}{LGR}%
          {\mst@greekfont}{\mst@series@normal}{\MTgreekitdefault}%
  \SetSymbolFont{mtgreekit}{bold}{LGR}%
          {\mst@greekfont}{\mst@boldvariant@bold}{\MTgreekitdefault}%
  \SetMathAlphabet{\mathgreekupbold}{normal}{LGR}%
          {\mst@greekfont}{\mst@boldvariant@normal}{\MTgreekupdefault}%
  \SetMathAlphabet{\mathgreekupbold}{bold}{LGR}%
          {\mst@greekfont}{\mst@boldvariant@bold}{\MTgreekupdefault}%
  \SetMathAlphabet{\mathgreekitbold}{normal}{LGR}%
          {\mst@greekfont}{\mst@boldvariant@normal}{\MTgreekitdefault}%
  \SetMathAlphabet{\mathgreekitbold}{bold}{LGR}%
```

```
          {\mst@greekfont}{\mst@boldvariant@bold}{\MTgreekitdefault}%
 \else
  \ifmst@selfGreek
   \SetSymbolFont{mtselfGreekfont}{normal}{OT1}%
          {\mst@greekfont}{\mst@series@normal}{\mst@greek@ush}%
   \SetSymbolFont{mtselfGreekfont}{bold}{OT1}%
          {\mst@greekfont}{\mst@boldvariant@bold}{\mst@greek@ush}%
  \fi
 \fi
```

1.3za adds the log message in case of `selfGreek` option.

```
\ifmst@subdued
\else
 \mst@infoline{Latin letters in the `normal\string', resp. `bold\string',}%
 \mst@infoline{math versions are now set up to use the fonts}%
 \mst@infoline{\mst@encoding@normal/\mst@family@normal/\mst@series@normal
               /\mst@ltshape@normal, resp.
               \mst@encoding@normal/\mst@family@normal/\mst@boldvariant@normal
               /\mst@ltshape@normal.}%
\ifmst@frenchmath\mst@infoline{(uppercase: \mst@shape@normal)}\fi
\ifmst@LGRgreek
 \mst@infoline{Greek letters (upper:
    `\ifmst@greek@upper@up\MTgreekupdefault\else\MTgreekitdefault\fi\string',
                                           lower:
    `\ifmst@greek@lower@up\MTgreekupdefault\else\MTgreekitdefault\fi\string')
    will use font}%
 \mst@infoline{family `\mst@greekfont\string' (LGR).}%
 \else
   \ifmst@selfGreek
     \mst@infoline{Capital Greek letters (shape `\mst@greek@ush\string') will use font}
     \mst@infoline{family `\mst@greekfont\string' (OT1).}%
   \fi
 \fi
 \ifmst@nodigits\else
   \mst@infoline{Other characters (digits, ...) and \string\log-like names will be}%
   \mst@infoline{typeset with the \mst@shape@normal\space shape.}%
 \fi
\fi % not subdued
\ifmst@nohbar\else
   \mst@infoline{\string\hbar}%
   \mst@dothe@hbarstuff
       \mst@hbar@mvnormal\mst@ltbar@mvnormal\mst@encoding@normal{normal}%
   \let\mst@hbar@mvbold\mst@hbar@mvnormal
\fi
```

1.4d lets the option `noletters` prevent definition of `\inodot` and `\jnodot`.

```
\ifmst@noletters\else
  \mst@dothe@inodotstuff\inodot\jnodot\mst@encoding@normal{normal}%
  \let\mst@inodot@mvnormal\inodot
  \let\mst@inodot@mvbold\inodot
  \let\mst@jnodot@mvnormal\jnodot
  \let\mst@jnodot@mvbold\jnodot
\fi
\ifmst@mathaccents
  \mst@infoline{math accents}%
  \mst@dothe@mathaccentsstuff{normal}\mst@encoding@normal
\fi
\ifmst@nominus\else
\ifmst@subdued
 \mst@infoline{minus is kept as is in subdued normal and bold math versions}%
\else
 \mst@infoline{minus either as endash, Unicode minus, emdash or hyphen}%
```

If `subdued` the `\mst@minus@mvnormal` etc... as defined via the `\mst@dothe@endashstuff` et al. would never end up being used in their respective *normal* and *bold* versions, because the actual mathcode assignment to `-` will proceed via `\mst@subduedminus` which simply uses the mathcode in place at `\begin{document}`. And `\MTnonlettersobeymathxx` is a no-op in `subdued` math versions. And (at `1.4c`) `\MTnonlettersdonotobeymathxx` (which is executed via `\MTeverymathoff`) will do (via `\mst@undo@nonletters`) `\mst@subduedminus` in these math versions (see `\mst@nonmathactiveminus`).

With `everymath` option the mathcode assignment is done similarly on entering the math version (either `\mst@subduedminus` or `\mst@nonsubduedminus`) or when using mathematically active character via `\mst@the\mst@do@nonletters` done at `\everymath`. And `\mst@the` is set to `\@gobble` in subdued math versions.

So at `1.4c` the execution of this code was made conditional on not being in `subdued` mode.

```
 \ifmst@endash
  \mst@dothe@endashstuff\mst@minus@mvnormal\mst@varfam@minus@mvnormal
                        \mst@encoding@normal {normal}%
  \mst@dothe@endashstuff\mst@minus@mvbold\mst@varfam@minus@mvbold
                        \mst@encoding@normal {bold}%
 \else
  \ifmst@emdash
    \mst@dothe@emdashstuff\mst@minus@mvnormal\mst@varfam@minus@mvnormal
                          \mst@encoding@normal {normal}%
    \mst@dothe@emdashstuff\mst@minus@mvbold\mst@varfam@minus@mvbold
                          \mst@encoding@normal {bold}%
  \else
    \mst@dothe@hyphenstuff\mst@minus@mvnormal\mst@varfam@minus@mvnormal
    \let\mst@minus@mvbold\mst@minus@mvnormal
    \let\mst@varfam@minus@mvbold\mst@varfam@minus@mvnormal
  \fi
 \fi
\fi % not subdued
\fi % not nominus
```

`1.3zb` moves this info line last and also explicitly mentions `italic` or `frenchmath` (here and at some other locations above).

```
\ifmst@subdued
  \mst@infoline{Subdued `normal\string' and `bold\string' math versions.}%
\fi
```

```
  \ifmst@italic
    \mst@infoline{The \ifmst@frenchmath frenchmath \else
                                          italic \fi option is in effect.}%
  \fi
}%
\let\mathastext\Mathastext
\Mathastext
```

Additional appropriate messages to the terminal and the log.

```
\ifmst@eulergreek
      \mst@infoline{Greek letters will use the Euler font.}%
      \mst@infoline{Use \string\MathastextEulerScale{<factor>} to scale the
               font.}%
      \ifmst@subdued
          \mst@infoline{(subdued mode: `normal\string' and `bold\string' math versions}%
          \mst@infoline{\space keep the default Greek letters).}%
      \fi
\else
\ifmst@symbolgreek
       \mst@infoline{Greek letters will use the PostScript Symbol font.}%
       \mst@infoline{Use \string\MathastextSymbolScale{<factor>} to scale the font.}%
      \ifmst@subdued
          \mst@infoline{(subdued mode: `normal\string' and `bold\string' math versions}%
          \mst@infoline{\space keep the default Greek letters).}%
      \fi
\fi\fi
```

`Math sizes` I took the code for \Huge and \HUGE from the moresize package of Christian CORNELSSEN

```
\ifmst@defaultsizes\else
\providecommand\@xxxpt{29.86}
\providecommand\@xxxvipt{35.83}
\ifmst@twelve
  \def\Huge{\@setfontsize\Huge\@xxxpt{36}}
  \def\HUGE{\@setfontsize\HUGE\@xxxvipt{43}}
\mst@infoline{\string\Huge\space and \string\HUGE\space have been (re)-defined.}
\else
  \def\HUGE{\@setfontsize\HUGE\@xxxpt{36}}
\mst@infoline{\string\HUGE\space has been (re)-defined.}
\fi
```

I choose rather big subscripts.

```
\def\defaultscriptratio{.8333}
\def\defaultscriptscriptratio{.7}
\DeclareMathSizes{9}{9}{7}{5}
\DeclareMathSizes{\@xpt}{\@xpt}{8}{6}
\DeclareMathSizes{\@xipt}{\@xipt}{9}{7}
\DeclareMathSizes{\@xiipt}{\@xiipt}{10}{8}
\DeclareMathSizes{\@xivpt}{\@xivpt}{\@xiipt}{10}
```

```
\DeclareMathSizes{\@xviipt}{\@xviipt}{\@xivpt}{\@xiipt}
\DeclareMathSizes{\@xxpt}{\@xxpt}{\@xviipt}{\@xivpt}
\DeclareMathSizes{\@xxvpt}{\@xxvpt}{\@xxpt}{\@xviipt}
\DeclareMathSizes{\@xxxpt}{\@xxxpt}{\@xxvpt}{\@xxpt}
\DeclareMathSizes{\@xxxvipt}{\@xxxvipt}{\@xxxpt}{\@xxvpt}
\mst@infoline{mathastext has declared larger sizes for subscripts.}
\mst@infoline{To keep LaTeX defaults, use option `defaultmathsizes\string'.}
\fi
```

`\MTeverymathoff` 1.3i 2016/01/06 Compatibility patch with `\url` from `url.sty` and `\url`/`\nolinkurl` from `hyperref.sty`.

1.3j 2016/01/15 renamed the macro from `\MTactivemathoff` to `\MTeverymathoff`, as it is not exclusively a matter of math active characters due to `\MTeasynonlettersdonotobeymathxx`.

1.3o 2016/05/03 adds `\MTdonotfixfonts`. Operant with LuaLaTeX only.

1.4 2024/07/20 keeps the `\MTmathstandardletters` as a component of `\MTeverymathoff`. It was checked that url and hyperref do not change mathcodes of ascii letters prior to location where `\MTeverymathoff` gets executed, so nothing is overwritten, despite the new mode of action of `\MTmathstandardletters`.

1.4 has `\MTmathstandarddigits` and inserts it into `\MTeverymathoff`.

```
\newcommand*\MTeverymathoff {%
    \MTnormalasterisk
    \MTnormalprime
    \MTnonlettersdonotobeymathxx
    \MTeasynonlettersdonotobeymathxx
    \MTmathstandardletters
    \MTmathstandarddigits
    \MTdonotfixfonts
}%
\AtBeginDocument {%
   \@ifpackageloaded{hyperref}
   {\def\Hurl{\begingroup\MTeverymathoff\Url}}
   {\@ifpackageloaded{url}{\DeclareUrlCommand\url{\MTeverymathoff}}{}}%
}%
```

`\MTeverymathdefault` 1.3j 2016/01/15 Customizable command which gets executed by `\MTversion` except when switching to `normal/bold` if option `subdued`. The included `\MTicinmath` does `\MTmathactiveletters` which will also activate the math skips around letters.

The `\MTeverymathdefault` does not include `\MTmathoperatorsobeymathxx` as the latter does not correspond to something done during execution of `\the\everymath`.

Should I put `\let\newmcodes@\mst@newmcodes@` here too ? No, it is not done at everymath.

During the loading, the (non subdued) package does `\MTactiveasterisk` (if option asterisk), `\MTprimedoesskip`, `\MTeasynonlettersobeymathxx` and `\MTmathactiveletters`. There is some code at begin document for decisions about italic corrections, this code does not emit again `\MTmathactiveletters`, hence a `\MTmathstandardletters` in the preamble is not overruled. Furthermore the at begin document code will not overrule user emitted `\MTnoicinmath` etc... commands in the preamble.

And user can employ `\MTnormalexists`, etc..., from inside the preamble, it will not be overruled (as it is delayed at begin document to after **mathastext** dealings).

1.3o 2016/05/03 adds `\MTfixfonts`. Operant with LuaLaTeX only.

\MTmathactivedigits of 1.4 is a no-op except under option activedigits.

```
\newcommand*\MTeverymathdefault {%
    \MTactiveasterisk
    \MTprimedoesskip
    \MTeasynonlettersobeymathxx
    \MTicinmath
    \MTmathactivedigits
    \MTfixfonts
}%
```

Things to do last "at begin document"

1.4 has a significant change here that mathematical activation of ascii letters is now not incorporated into the \everymath and \everydisplay.

```
\ifmst@everymath
  \AtBeginDocument{%
    \everymath\expandafter{\the\everymath
        \mst@the\mst@do@nonletters \let\mst@the\@gobble
        \mst@theeasy\mst@do@easynonletters  \let\mst@theeasy\@gobble
    }%
    \everydisplay\expandafter{\the\everydisplay
        \mst@the\mst@do@nonletters \let\mst@the\@gobble
        \mst@theeasy\mst@do@easynonletters  \let\mst@theeasy\@gobble
    }%
  }
\fi
```

1.3j: moved here to be executed at begin document (and not from inside \Mathastext@.) The \MTeverymathoff does: \MTnormalasterisk, \MTnormalprime, \MTnonlettersdonotobeymathxx, \MTeasynonlettersdonotobeymathxx, \MTmathstandardletters.

1.3m: doing \MTmathactiveletters in subdued mode immediately after \begin{document} resulted in errors because \mst@itcorr had been left undefined. We thus add \MTnoicinmath to the subdued initialization.

Since 1.3n there is \MTresetnewmcodes which needs \mst@originalnewmcodes@, itself defined at begin document. Thus we have wrapped the whole thing in \AtEndOfPackage (at 1.3u whole code directly moved at end of package).

And 1.3p adds here \MTcustomizenewmcodes which had been regrettably forgotten by 1.3n.

1.3t adds some extras to handle correctly the minus sign and dotless i and j in subdued mode, even in case of usage with fontspec.

1.3u similarly lets math accents be correctly subdued.

1.3v adapts to \hbar and math accents now being robust with LaTeX 2019-10-01 or later.

1.3w pays attention to the fact that \hbar may well be a \mathchar and not a robust macro! And no need to worry about \hbar<space> finally in revised code.

```
\AtBeginDocument{%
  \MTcustomizenewmcodes
  \let\mst@original@hbar\hbar
  \let\mst@original@imath\imath
  \let\mst@original@jmath\jmath
  \@tfor\@tempa:={grave}{acute}{check}{breve}{bar}%
                 {dot}{ddot}{mathring}{hat}{tilde}%
  \do
  {\expandafter\let\csname mst@original@\@tempa\expandafter\endcsname
                  \csname \@tempa\endcsname
   \expandafter\let\csname mst@original@\@tempa\space\expandafter\endcsname
                  \csname \@tempa\space\endcsname
  }%
```

`1.4c` has `\mst@mathcodenum` which is either alias of `\Umathcodenum` or of `\mathcode`.

```
  \edef\mst@subduedminus{\mst@mathcodenum`\noexpand\-=\the\mst@mathcodenum`\-\relax}%
  \ifmst@subdued
```

This `\MTeverymathoff` does `\MTnonlettersdonotobeymathxx`, which has been already set to its final definition.

```
    \MTeverymathoff
    \MTresetnewmcodes
    \MTnoicinmath
    \MTmathoperatorsdonotobeymathxx
    \let\inodot\imath
    \let\jnodot\jmath
    \mst@subduedminus
  \else
     \mst@nonsubduedhbar
```

`1.3v` needs `\mst@nonsubduedmathaccents` to get executed later (see code comments for `\mst@dothe@mathaccentsstuff`).

```
     \mst@nonsubduedminus
```

`1.3j`: an earlier version of this code was earlier part of `\Mathastext@`. As we are now in `\AtBeginDocument` we try to be careful not to overwrite `\MTicinmath`, `\MTnoicinmath`, `\MTicalsoinmathxx`, ... if issued by the user in the preamble, though. And we do not execute `\MTmathactiveletters`, it is issued by **mathastext** at loading time in order to allow user to cancel it if desired from inside the preamble.

```
     \ifx\mst@itcorr\@undefined
         \def\mst@itcorr{\ifnum\fam=\m@ne\/\fi}%
         \@for\mst@tmp:=it,sl\do
           {\ifx\mst@tmp\mst@ltshape@normal\let\mst@itcorr\@empty\fi }%
     \fi
     \ifx\mst@ITcorr\@undefined
         \let\mst@ITcorr\mst@itcorr
         \ifmst@frenchmath
           \def\mst@ITcorr{\ifnum\fam=\m@ne\/\fi}%
           \@for\mst@tmp:=it,sl\do
               {\ifx\mst@tmp\mst@shape@normal\let\mst@ITcorr\@empty\fi }%
         \fi
     \fi
  \fi
}%
\AtEndOfPackage{\AtBeginDocument{\ifmst@subdued\else\mst@nonsubduedmathaccents\fi}}%
```

`subdued` `1.15`: The subdued code was initiated in May 2011. I returned to `mathastext` on Sep 24, 2012, and decided to complete what I had started then, but in the mean time I had forgotten almost all of the little I knew about LaTeX macro programming.

The point was to extract the data about how are ‘letters’ and ‘operators’ in the normal and bold versions, through obtaining the math families of ‘a’ and ‘1’, respectively[1]. Due to the reassignements done for characters by `mathastext` I also had decided in 2011 that the OT1 encoding, if detected, should be replaced by T1

[1]but the *euler* package for example assigns the digits to the *letters* symbol font...

`1.15d`: Oct 13, 2012. The `\mathcode` thing has to be used with care under Unicode engines. Unfortunately the `\luatexUmathcode` macro is helpless as it is not possible to know if it will return a legacy mathcode or a Unicode mathcode. On the other hand the much saner `\XeTeXmathcodenum` always return a Unicode mathcode.

UPDATE for **mathastext** 1.3 (2013/09/02): since the release of lualatex as included in TL2013, `\luatexUmathcodenum` behaves as `\XeTeXmathcodenum` so **mathastext** 1.3 treats identically under both unicode engines the equal and minus signs (and the vertical bar).

`1.15e`: Oct 22, 2012. I add the necessary things to also subdue the `\mathbf`, `\mathit`, `\mathsf` and `\mathtt` macros (previous version only took care of the symbol alphabets `\mathnormal` and `\mathrm`.) [update: `1.15f` does that in a completely different and much simpler way] Notice that the package defines a `\mathnormalbold` macro, but it will not be subdued in the normal and bold math versions.

`1.15f`: Oct 23, 2012. The previous version of the code queried the math family of a, respectively 1, to guess and then extract the fonts to be reassigned to mtletterfont and mtoperatorfont (which is done at the end of this .sty file). The present code simply directly uses letters and operators (so mathastext could not subdue itself... if it was somehow cloned), but obtains indeed the corresponding font specifications in normal and bold in a cleaner manner. But it is so much shorter (and avoids the LuaLaTeX problem with `\luatexUmathcode`). Anyhow, for example the euler package puts the digits in the letters math family! so the previous method was also error prone. In fact there is no way to do this subdued mechanism on the basis of the legacy code of mathastext. The only way is to rewrite entirely the package to query all mathcodes of things it changes in order to be able to revert these changes (and one would have to do even more hacking for `\mathversion{normal}` and not only `\MTversion{normal}` to work).

`1.15f`: and also I take this opportunity to do the subdued math alphabets things in a much much easier way, see below.

`1.3s` 2018/08/21: I have half-forgotten the reasons for modifiying the font encoding to current `\encodingdefault`, but at any rate this should not be done in a fontspec context, encoding default being (now) `TU` it is very unlikely modifying from `TU` or to `TU` from something else will do any good. I add workaround here for case of fontspec being detected via the `\encodingdefault` setting.

`1.3t` 2018/08/22: the `1.3s` fix erroneously removed the `OT1->T1` replacement in `TU` context.

`1.3u`: the whole thing will only get executed At Begin Document.

I realize extremely late (2023/12/28) I never said explicitly anywhere it seems in the code comments that the `frenchmath` option effect is *not* subdued: the uppercase Latin letters `\mathcode`’s are not changed back to their defaults at start of a subdued document or when going to the subdued normal math version! Time to do so before the package enters resolutely dormant maintenance status soon... and I end up really forgetting enything and having wrong expectations on what is the behavior of the package.

```
\ifmst@subdued
  \AtBeginDocument{%
  \def\mst@reserved#1\getanddefine@fonts\symletters#2#3\@nil{%
        \def\mst@normalmv@letter{#2}}%
  \expandafter\mst@reserved\mv@normal\@nil
  \def\mst@reserved#1\getanddefine@fonts\symletters#2#3\@nil{%
```

```
        \def\mst@boldmv@letter{#2}}%
  \expandafter\mst@reserved\mv@bold\@nil
  \def\mst@reserved#1\getanddefine@fonts\symoperators#2#3\@nil{%
        \def\mst@normalmv@operator{#2}}%
  \expandafter\mst@reserved\mv@normal\@nil
  \def\mst@reserved#1\getanddefine@fonts\symoperators#2#3\@nil{%
        \def\mst@boldmv@operator{#2}}%
  \expandafter\mst@reserved\mv@bold\@nil
  \edef\mst@tmp@enc{\mst@encoding@normal}%
  \def\mst@reserved#1/#2/#3/#4/{\gdef\mst@debut{#1}\gdef\mst@reste{#2/#3/#4}}%
  \begingroup\escapechar\m@ne
      \xdef\mst@funnyoti{\expandafter\string\csname OT1\endcsname}%
      \expandafter\expandafter\expandafter
          \mst@reserved\expandafter\string\mst@normalmv@operator/%
  \endgroup
  \ifx\mst@debut\mst@funnyoti\ifx\mst@tmp@enc\mst@oti\def\mst@tmp@enc{T1}\fi\fi
  \edef\mst@normalmv@operator{\expandafter\noexpand\csname
       \if1\mst@OneifUniEnc
         \ifx\mst@debut\mst@funnyoti T1\else\mst@debut\fi
       \else
         \mst@tmp@enc
       \fi/\mst@reste\endcsname}%
  \edef\mst@tmp@enc{\mst@encoding@bold}%
  \begingroup\escapechar\m@ne
      \expandafter\expandafter\expandafter
          \mst@reserved\expandafter\string\mst@boldmv@operator/%
  \endgroup
  \ifx\mst@debut\mst@funnyoti\ifx\mst@tmp@enc\mst@oti\def\mst@tmp@enc{T1}\fi\fi
  \edef\mst@boldmv@operator{\expandafter\noexpand\csname
       \if1\mst@OneifUniEnc
         \ifx\mst@debut\mst@funnyoti T1\else\mst@debut\fi
       \else
         \mst@tmp@enc
       \fi/\mst@reste\endcsname}%
  \expandafter\SetSymbolFont@
    \expandafter\mv@normal\mst@normalmv@letter\symmtletterfont
  \expandafter\SetSymbolFont@
    \expandafter\mv@bold\mst@boldmv@letter\symmtletterfont
  \expandafter\SetSymbolFont@
    \expandafter\mv@normal\mst@normalmv@operator\symmtoperatorfont
  \expandafter\SetSymbolFont@
    \expandafter\mv@bold\mst@boldmv@operator\symmtoperatorfont
  \immediate\write\m@ne{}%
  \PackageInfo{mathastext}{...entering subdued mode...\MessageBreak ...done}%
  \immediate\write\m@ne{}%
  }%
\fi
```

`Preamble-only...` “Only preamble” restrictions. I was way too much obedient back in 2011, particularly taking into account how much of a pain it has been and still is that things such as `\DeclareMathSymbol` or `\DeclareMathAccent` are preamble-only. But keeping this for time being, however not using `\@onlypreamble` which breaks one's heart when tracing to see how much place it takes, so we do it in one go.

```
\expandafter \gdef \expandafter \@preamblecmds \expandafter {\@preamblecmds
\do\MTitgreek
\do\MTupgreek
\do\MTitGreek
\do\MTitGreek
\do\Mathastextitgreek
\do\Mathastextupgreek
\do\MathastextitGreek
\do\MathastextitGreek
\do\MTgreekfont
\do\Mathastextgreekfont
\do\MTgreekupdefault
\do\MTgreekitdefault
\do\MTDeclareVersion
\do\MathastextDeclareVersion
\do\MTWillUse
\do\MathastextWillUse
\do\Mathastextwilluse
\do\Mathastext
\do\mathastext
}
\immediate\write\m@ne{}
\PackageInfo{mathastext}{Loading is complete. \space You can now use \string\Mathastext
                         \space to\MessageBreak
                         modify the normal and bold math versions. \space
                         Use it\MessageBreak
                         with optional argument or use \string\MTDeclareVersion\space
                         to\MessageBreak
                         declare additional math versions\@gobble}
\endinput
```