

The `lualatex-math` package*

Philipp Stephani
p.stephani2@gmail.com

2020/09/25

Contents

1	Introduction	1
2	Interface	2
3	Implementation of the $\LaTeX 2_\epsilon$ package	2
3.1	Requirements	2
3.2	Messages	3
3.3	Initialization	3
3.4	Patching	3
3.5	$\LaTeX 2_\epsilon$ kernel	5
3.6	<code>amsmath</code>	5
3.7	<code>mathtools</code>	8
3.8	<code>icomma</code>	9
4	Implementation of the <code>Lua\LaTeX</code> module	10

1 Introduction

`Lua \TeX` brings major improvements to all areas of \TeX typesetting and programming. They are made available through new primitives or the embedded Lua interpreter, and combining them with existing $\LaTeX 2_\epsilon$ packages is not a task the average \LaTeX user should have to care about. Therefore a multitude of $\LaTeX 2_\epsilon$ packages have been written to bridge the gap between documents and the new features. The `lualatex-math` package focuses on the additional possibilities for mathematical typesetting. The most eminent of the new features is the ability to use Unicode and OpenType fonts, as provided by Will Robertson's `unicode-math` package. However, there is a smaller group of changes unrelated to Unicode: these are to be dealt with in this package. While in principle most \TeX documents written for traditional engines should work just fine with `Lua \TeX` , there is a small number of breaking changes that require the attention of package authors. The `lualatex-math` package tries to fix some of the issues encountered while porting traditional macro packages to `Lua \LaTeX` .

The decision to write patches for existing macro packages should not be made lightly: monkey patching done by somebody different from the original package author ties the patching package to the implementation details of the patched functionality and breaks all rules of encapsulation. However, due to the lack of

*This document corresponds to `lualatex-math` v1.9, dated 2020/09/25.

alternatives, it has become an accepted way of providing new functionality in L^AT_EX. To keep the negative impact as small as possible, the lualatex-math package patches only the L^AT_EX 2_ε kernel and a small number of popular packages. In general, this package should be regarded as a temporary kludge that should be removed once the math-related packages are updated to be usable with LuaT_EX. By its very nature, the package is likely to cause problems; in such cases, please refer to the issue tracker¹.

2 Interface

The lualatex-math package can be loaded with `\usepackage` or `\RequirePackage`, as usual. It has no options and no public interface; the patching is always done when the package is loaded and cannot be controlled. As a matter of course, the lualatex-math package needs LuaL^AT_EX to function; it will produce error messages and refuse to load under other engines and formats. The package depends on the expl3 bundle, the etoolbox package and the filehook package. The lualatex-math package is independent of the unicode-math package; the fixes provided here are valid for both Unicode and legacy math typesetting.

Currently patches for the L^AT_EX 2_ε kernel and the amsmath, mathtools and icomma packages are provided. It is not relevant whether you load these packages before or after lualatex-math. They should work as expected (and ideally you shouldn't notice anything), but if you load other packages that by themselves overwrite commands patched by this package, bad things may happen, as it is usual with L^AT_EX.

`\mathstyle` One user-visible change is that the new `\mathstyle` primitive should work in all cases after the lualatex-math package has been loaded, provided you use the high-level macros `\frac`, `\binom`, and `\genfrac`. `\frac`, `\binom`, `\genfrac` The fraction-like T_EX primitives like `\over` or `\atopwithdelims` and the plain T_EX leftovers like `\brack` or `\choose` cannot be patched, and you shouldn't use them.

3 Implementation of the L^AT_EX 2_ε package

3.1 Requirements

```

1 (*package)
2 (@@=lltxmath)
3 \NeedsTeXFormat{LaTeX2e}[2020/02/02]
4 \RequirePackage{expl3}[2018/06/18]
5 \ProvidesExplPackage{lualatex-math}{2020/09/25}{1.9}%
6   {Patches for mathematics typesetting with LuaLaTeX}
7 \RequirePackage { etoolbox } [ 2007/10/08 ]
8 \cs_if_exist:NF \newluabytecode
9   { \RequirePackage { luatexbase } [ 2010/05/27 ] }
10 \directlua{require("lualatex-math")}

```

`\@@_restore_catcode:N` Executing the exhaustive expansion of `\@@_restore_catcode:N`*<character token>* restores the category code of the *<character token>* to its current value.

```

11 \cs_new_nopar:Npn \@@_restore_catcode:N #1 {
12   \char_set_catcode:nn { \int_eval:n { `#1 } }
13   { \char_value_catcode:n { `#1 } }
14 }

```

¹<https://github.com/phst/lualatex-math/issues>

We use the macro defined above to restore the category code of the dollar sign. There are packages that make the dollar sign active; hopefully they get loaded after the packages we are trying to patch.

```

15 \exp_args:Nx \AtEndOfPackage {
16   \@@_restore_catcode:N \$
17 }
18 \char_set_catcode_math_toggle:N \$

```

3.2 Messages

- luatex-required Issued when not running under LuaTeX.
- ```

19 \msg_new:nnn { luatex-math } { luatex-required } {
20 The~ luatex-math~ package~ requires~ LuaTeX. \\
21 I~ will~ stop~ loading~ now.
22 }

```
- macro-expected Issued when trying to patch a non-macro. The first argument must be the detokenized macro name.
- ```

23 \msg_new:nnn { luatex-math } { macro-expected } {
24   I've~ expected~ that~ #1~ is~ a~ macro,~ but~ it~ isn't.
25 }

```
- wrong-meaning Issued when trying to patch a macro with an unexpected meaning. The first argument must be the detokenized macro name; the second argument must be the actual detokenized meaning; and the third argument must be the expected detokenized meaning.
- ```

26 \msg_new:nnn { luatex-math } { wrong-meaning } {
27 I've~ expected~ #1~ to~ have~ the~ meaning \\
28 #3, \\
29 but~ it~ has~ the~ meaning \\
30 #2.
31 }

```
- patch-macro Issued when a macro is patched. The first argument must be the detokenized macro name.
- ```

32 \msg_new:nnn { luatex-math } { patch-macro } {
33   I'm~ going~ to~ patch~ macro~ #1.
34 }

```

3.3 Initialization

Unless we are running under LuaTeX, we issue an error and quit immediately.

```

35 \sys_if_engine_luatex:F {
36   \msg_error:nn { luatex-math } { luatex-required }
37   \endinput
38 }

```

3.4 Patching

\@@_temp:w A scratch macro.

```

39 \cs_new_eq:NN \@@_temp:w \prg_do_nothing:

```

\@@_patch:NNnnn The auxiliary macro \@@_patch:NNnnn⟨*command*⟩⟨*factory command*⟩⟨*parameter text*⟩⟨*expected replacement text*⟩⟨*new replacement text*⟩ tries to patch ⟨*command*⟩. If ⟨*command*⟩ is undefined, do nothing. Otherwise it must be a macro with the given ⟨*parameter text*⟩ and ⟨*expected replacement text*⟩, created by the

given $\langle factory command \rangle$ or equivalent. In this case it will be overwritten using the $\langle parameter text \rangle$ and the $\langle new replacement text \rangle$. Otherwise issue a warning and don't overwrite.

```

40 \cs_new_protected_nopar:Npn \@@_patch:NNnnn #1 #2 #3 #4 #5 {
41   \cs_if_exist:NT #1 {
42     \token_if_macro:NTF #1 {
43       \group_begin:
44       #2 \@@_temp:w #3 { #4 }
45       \cs_if_eq:NNTF #1 \@@_temp:w {
46         \msg_info:nxx { lualatex-math } { patch-macro }
47         { \token_to_str:N #1 }
48       \group_end:
49       #2 #1 #3 { #5 }
50     } {
51       \msg_warning:nxxxx { lualatex-math } { wrong-meaning }
52       { \token_to_str:N #1 } { \token_to_meaning:N #1 }
53       { \token_to_meaning:N \@@_temp:w }
54     \group_end:
55   }
56 } {
57   \msg_warning:nxx { lualatex-math } { macro-expected }
58   { \token_to_str:N #1 }
59 }
60 }
61 }
62 \cs_generate_variant:Nn \@@_patch:NNnnn { c }

```

$\backslash\@_set_mathchar:NN$ The macro $\backslash\@_set_mathchar:NN\langle control sequence \rangle\langle token \rangle$ defines the $\langle control sequence \rangle$ as an extended mathematical character shorthand whose mathematical code is given by the mathematical code of the character $\langle token \rangle$. We cannot use the $\backslash\Umathcharnumdef$ primitive here since we would then rely on the $\backslash\Umathcodenum$ primitive which is currently broken.²

```

63 \cs_new_protected_nopar:Npn \@@_set_mathchar:NN #1 #2 {
64   \Umathchardef #1
65   \lua_now:e {
66     lualatex.math.print_class_fam_slot( \int_eval:n { `#2 } )
67   }
68   \scan_stop:
69 }

```

$\backslash\@_before_package:nn$ The macro $\backslash\@_before_package:nn\{\langle package \rangle\}\{\langle code \rangle\}$ executes the $\langle code \rangle$ before the $\langle package \rangle$ is loaded. Accordingly, $\backslash\@_after_package:nn\{\langle package \rangle\}\{\langle code \rangle\}$ executes the $\langle code \rangle$ after the $\langle package \rangle$ is loaded. If the $\langle package \rangle$ is already loaded, nothing happens. We prefer using native L^AT_EX 2_ε hooks if possible.

```

70 \ifl@t@r \fmtversion { 2020/10/01 } {
71   \cs_new_protected_nopar:Npn \@@_before_package:nn #1 #2 {
72     \AddToHook { package/before/#1 } { #2 }
73   }
74   \cs_new_protected_nopar:Npn \@@_after_package:nn #1 #2 {
75     \AddToHook { package/after/#1 } { #2 }
76   }
77 } {
78   \RequirePackage { filehook } [ 2011/03/09 ]
79   \cs_new_protected_nopar:Npn \@@_before_package:nn #1 #2 {
80     \AtBeginOfPackageFile { #1 } { #2 }
81   }

```

²<http://tug.org/pipermail/luatex/2012-October/003794.html>

```

82 \cs_new_protected_nopar:Npn \@@_after_package:nn #1 #2 {
83   \AtEndOfPackageFile { #1 } { #2 }
84 }
85 }

```

`\@@_after_package_or_now:nn` The macro `\@@_after_package_or_now:nn{<package>}{<code>}` executes the `<code>` after the `<package>` is loaded. If the `<package>` is already loaded, the `<code>` is executed immediately.

```

86 \cs_new_protected_nopar:Npn \@@_after_package_or_now:nn #1 #2 {
87   \ifpackageloaded { #1 } { #2 } { \@@_after_package:nn { #1 } { #2 } }
88 }

```

3.5 L^AT_EX 2_ε kernel

LuaT_EX enables access to the current mathematical style via the `\mathstyle` primitive. For this to work, fraction-like constructs (e.g., `<numerator> \over <denominator>`) have to be enclosed in a `\Ustack` group. `\frac` can be patched to do this, but the plain T_EX remnants `\choose`, `\brack` and `\brace` should be discouraged.

`\frac` Here we assume that nobody except `amsmath` redefines `\frac`. This is obviously not the case, but we ignore other packages (e.g., `nath`) for the moment. We only patch the L^AT_EX 2_ε kernel definition if the `amsmath` package is not loaded; the corresponding patch for `amsmath` follows below. Since `\frac` is declared by `\DeclareRobustCommand`, we must patch the macro `\frac␣`.

```

89 \AtEndPreamble {
90   \ifpackageloaded { amsmath } { } {
91     \@@_patch:cNnnn { frac~ } \cs_set:Npn { #1 #2 } {
92       {
93         \begingroup #1 \endgroup \over #2
94       }
95     } {

```

To do: do we need the additional set of braces around `\Ustack`?

```

96     {
97       \Ustack { \group_begin: #1 \group_end: \over #2 }
98     }
99   }
100 }
101 }

```

3.6 amsmath

The popular `amsmath` package is subject to three LuaT_EX-related problems:

- The `\mathcode` primitive is used several times, which fails for Unicode math characters. `\Umathcode` should be used instead.
- Legacy font dimensions are used for constructing stacks in the `\substack` command and the `subarray` environment. This doesn't work if a Unicode math font is selected.
- The fraction commands `\frac` and `\genfrac` don't use the `\Ustack` primitive.

These problems have been fixed in version 2.17i of `amsmath`, so we don't attempt to patch it if that version is loaded.

`\c_@@_std_minus_mathcode_int` `\c_@@_std_equal_mathcode_int` These constants contain the standard \TeX mathematical codes for the minus and the equal signs. We temporarily set the math codes to these constants before loading the `amsmath` package so that it can request the legacy math code without error.

```
102 \int_const:Nn \c_@@_std_minus_mathcode_int { "2200 }
103 \int_const:Nn \c_@@_std_equal_mathcode_int { "303D }
```

`\l_@@_minus_mathchar` `\l_@@_equal_mathchar` These mathematical characters are saved before `amsmath` is loaded so that we can temporarily assign the \TeX values to the mathematical codes of the minus and equals signs. The `amsmath` package queries these codes, and if they represent Unicode characters, the package loading will fail. If `amsmath` has already been loaded, there is nothing we can do, therefore we use the non-starred version of `\AtBeginOfPackageFile`.

```
104 \tl_new:N \l_@@_minus_mathchar
105 \tl_new:N \l_@@_equal_mathchar
106 \@@_before_package:nn { amsmath } {
107   \ifpackagelater { amsmath } { 2020/08/24 } { } {
108     \@@_set_mathchar:NN \l_@@_minus_mathchar \-
109     \@@_set_mathchar:NN \l_@@_equal_mathchar \=
110   }
```

Now we temporarily reset the mathematical codes.

```
110   \char_set_mathcode:nn { \- } { \c_@@_std_minus_mathcode_int }
111   \char_set_mathcode:nn { \= } { \c_@@_std_equal_mathcode_int }
112   \@@_after_package:nn { amsmath } { }
```

`\std@minus` `\std@equals` The `amsmath` package defines the control sequences `\std@minus` and `\std@equal` as mathematical character shorthands while loading, but uses our restored mathematical codes, which must be fixed.

```
113   \cs_set_eq:NN \std@minus \l_@@_minus_mathchar
114   \cs_set_eq:NN \std@equal \l_@@_equal_mathchar
```

Finally, we restore the original mathematical codes of the two signs.

```
115   \Umathcodenum `\- \l_@@_minus_mathchar
116   \Umathcodenum `\= \l_@@_equal_mathchar
117   }
118   }
119 }
```

All of the following fixes work even if `amsmath` is already loaded.

`\@begindocumenthook` `amsmath` repeats the definition of `\std@minus` and `\std@equal` at the beginning of the document, so we also have to patch the internal kernel macro `\@begindocumenthook` which contains the hook code.

```
120 \@@_after_package_or_now:nn { amsmath } {
121   \ifpackagelater { amsmath } { 2020/08/24 } { } {
122     \tl_replace_once:Nnn \@begindocumenthook {
123       \mathchardef \std@minus \mathcode `\- \relax
124       \mathchardef \std@equal \mathcode `\= \relax
125     } {
126       \@@_set_mathchar:NN \std@minus \-
127       \@@_set_mathchar:NN \std@equal \=
128     }
129   }
```

`subarray` The `subarray` environment uses legacy font dimensions. We simply patch it to use \LaTeX font parameters (and \LaTeX expressions instead of \TeX arithmetic). Since subscript arrays are conceptually vertical stacks, we use the sum of top and bottom

shift for the default vertical baseline distance (`\baselineskip`) and the minimum vertical gap for stack for the minimum baseline distance (`\lineskip`).

```

130 \ifpackagelater { amsmath } { 2020/09/23 } { } {
131   \@@_patch:NNnnn \subarray \cs_set:Npn { #1 } {
132     \vcenter
133     \bgroup
134     \Let@
135     \restore@math@cr
136     \default@tag
137     \baselineskip \fontdimen 10~ \scriptfont \tw@
138     \advance \baselineskip \fontdimen 12~ \scriptfont \tw@
139 \@@=}
140   \lineskip \thr@@ \fontdimen 8~ \scriptfont \thr@@
141 \@@=||txmath)
142   \lineskiplimit \lineskip
143   \ialign
144   \bgroup
145   \ifx c #1 \hfil \fi
146   $ \m@th \scriptstyle ## $
147   \hfil
148   \crcr
149 } {
150   \vcenter
151   \c_group_begin_token
152   \Let@
153   \restore@math@cr
154   \default@tag
155   \skip_set:Nn \baselineskip {
156     \Umathstacknumup \scriptstyle
157     + \Umathstackdenomdown \scriptstyle
158   }
159   \lineskip \Umathstackvgap \scriptstyle
160   \lineskiplimit \lineskip
161   \ialign
162   \c_group_begin_token
163   \token_if_eq_meaning:NNT c #1 { \hfil }
164   \Ustartmath
165   \m@th
166   \scriptstyle
167   \alignmark \alignmark
168   \Ustopmath
169   \hfil
170   \crcr
171 }

```

`\frac` Since `\frac` is declared by `\DeclareRobustCommand`, we must patch the macro `\frac_`.

```

172 \@@_patch:cNnnn { frac~ } \cs_set:Npn { #1 #2 } {
173   {
174 \@@=}
175     \begingroup #1 \endgroup \@@over #2
176   }
177 } {
178   {
179     \Ustack { \group_begin: #1 \group_end: \@@over #2 }
180 \@@=||txmath)
181   }
182 }

```

`\genfrac` Generalized fractions are typeset by the `\genfrac` command. Since `\genfrac` is declared by `\DeclareRobustCommand`, we have to patch the macro `\genfrac_U`.

```

183 \@@_patch:cNnnn { genfrac~ } \cs_set:Npn {
184   #1 #2 #3 #4 #5 #6
185 } {
186   {
187     \@mathstyle { #4 }
188     \genfrac@choice o { #1 }
189     {
190       \begingroup #5 \endgroup
191 <@@=)
192     \ifx @ #3 @ \@@over \else \@@above \fi #3 \relax
193     #6
194   }
195   \genfrac@choice c { #2 }
196 }
197 } {
198   {
199     \@mathstyle { #4 }
200     \genfrac@choice o { #1 }
201     {
202       \Ustack {
203         \group_begin: #5 \group_end:
204         \tl_if_empty:nTF { #3 } {
205           \@@over
206         } {
207           \@@above #3 \scan_stop:
208         }
209 <@@=lltxmath)
210         #6
211       }
212     }
213     \genfrac@choice c { #2 }
214   }
215 }
216 }
217 }

```

3.7 mathtools

`mathtools`' `\cramped` command and others that make use of its internal version use a hack involving a null radical. LuaTeX has primitives for setting material in cramped mode, so we make use of them.

`\MT_cramped_internal:Nn` The macro `\MT_cramped_internal:Nn<style>{<expression>}` typesets the *<expression>* in the cramped style corresponding to the given *<style>* (`\displaystyle` etc.); all we have to do in LuaTeX is to select the correct primitive. Rewriting the user-level `\cramped` command and employing `\mathstyle` would be possible as well, but we avoid this way since we want to patch only a single command.

```

218 \@@_after_package_or_now:nn { mathtools } {
219   \@@_patch:NNnnn \MT_cramped_internal:Nn
220   \cs_set_nopar:Npn { #1 #2 } {
221     \sbox \z@ {
222       $
223       \m@th
224       #1

```



```

225     \nulldelimiterspace = \z@
226     \radical \z@ { #2 }
227     $
228   }
229   \ifx #1 \displaystyle
230     \dimen@ = \fontdimen 8 \textfont 3
231     \advance \dimen@ .25 \fontdimen 5 \textfont 2
232   \else
233     \dimen@ = 1.25 \fontdimen 8
234     \ifx #1 \textstyle
235       \textfont
236     \else
237       \ifx #1 \scriptstyle
238         \scriptfont
239       \else
240         \scriptscriptfont
241       \fi
242     \fi
243     3
244   \fi
245   \advance \dimen@ -\ht\z@
246   \ht\z@ = -\dimen@
247   \box\z@
248 } {

```

Here the additional set of braces is absolutely necessary, otherwise the changed mathematical style would be applied to the material after the `\mathchoice` construct. As the original command works in both text and math mode, we use `\ensuremath` here.

```

249   {
250     \ensuremath {
251       \use:c { cramped \cs_to_str:N #1 } #2
252     }
253   }
254 }
255 }

```

3.8 icomma

The `icomma` package uses `\mathchardef` to save the mathematical code of the comma character. This breaks for Unicode fonts. The incompatibility was noticed by Peter Breitfeld.³

`\mathcomma` `icomma` defines the mathematical character shorthand `\icomma` at the beginning of the document, therefore we again patch `\@begindocumenthook`.

```

256 \@@_after_package_or_now:nn { icomma } {
257   \tl_replace_once:Nnn \@begindocumenthook {
258     \mathchardef \mathcomma \mathcode `\",
259   } {
260     \@@_set_mathchar:NN \mathcomma \,
261   }
262 }
263 \</package>

```

³<https://groups.google.com/forum/#!topic/de.comp.text.tex/Cputk-AJS5I/discussion>

4 Implementation of the Lua^AT_EX module

For the Lua module, we use the standard `luatexbase-modutils` template.

```

264  $\langle *lua \rangle$ 
265 luaAtex = luatex or {}
266 luaAtex.math = luatex.math or {}
267 luatexbase.provides_module({
268   name = "luaAtex-math",
269   date = "2013/08/03",
270   version = 1.3,
271   description = "Patches for mathematics typesetting with LuaLaTeX",
272   author = "Philipp Stephani",
273   licence = "LPPL v1.3+"
274 })

```

`unpack` The function `unpack` needs to be treated specially as it got moved around in Lua 5.2.

```

275 local unpack = unpack or table.unpack

276 local cctb = luatexbase.catcodetables or
277   {string = luatexbase.registernumber("catcodetable@string")}

```

`print_class_fam_slot` The function `print_class_fam_slot` takes one argument which must be a number. It interprets the argument as a Unicode code point whose mathematical code is printed in the form $\langle class \rangle_{\langle family \rangle}_{\langle slot \rangle}$, suitable for the right-hand side of `\Umathchardef`.

```

278 function luatex.math.print_class_fam_slot(char)
279   local code = tex.getmathcode(char)
280   local class, family, slot = unpack(code)
281   local result = string.format("%i %i %i ", class, family, slot)
282   tex.sprint(cctb.string, result)
283 end

284 return luatex.math
285  $\langle /lua \rangle$ 

```

Change History

v0.1	
General: Initial version	1
v0.2	
General: Added patch for the <code>icomma</code> package	9
v0.3	
General: Patched math group allocation to gain access to all families	5
v0.3a	
General: Updated for changes in <code>l3kernel</code>	1
v0.3b	
<code>\@begindocumenthook</code> : Another update for a change in <code>l3kernel</code>	6
v0.3c	
<code>\@@_set_mathchar:NN</code> : <code>l3kernel</code> renamed <code>\lua_now:x</code> to <code>\lua_now_x:n</code>	4
v1.0	
General: Switched to <code>l3docstrip</code>	1
v1.1	
<code>\@@_set_mathchar:NN</code> : Update reasoning why <code>\Umathcharnumdef</code> is not used here	4
General: Add fix and unit test for <code>amsopn</code>	8

v1.2	<code>\l_@@_equal_mathchar</code> : Replace removed macro <code>\chk_if_free_cs:N</code>	6
v1.3	General: Stop using the deprecated <code>module</code> function	10
v1.3a	<code>@@_set_mathchar:NN</code> : l3kernel has (currently) dropped <code>\lua_now_x:n</code>	4
v1.4	<code>\MT_cramped_internal:Nn</code> : Added <code>\ensuremath</code> to work around issue 11	9
	General: Removed patch for math group allocation; the kernel itself now supports all available math families	5
v1.4a	<code>@@_set_mathchar:NN</code> : <code>\lua_now_x:n</code> is back	4
	General: Avoid <code>\RequireLuaModule</code>	2
	Load <code>luatexbase</code> only if required	2
	Load all of <code>luatexbase</code>	10
	Pick up new name for string catcode table where available	10
	Use <code>expl3</code> versions of LuaTeX math primitives	2
v1.5	General: Removed patch for <code>\Mathstrutbox@</code> ; <code>amsmath</code> now has a definition usable in LuaTeX	6
	Removed unused helper macro <code>@@_char_dim:NN</code>	6
	Removed unused Lua function <code>print_fam_slot</code>	10
v1.6	General: Removed patch for <code>\newmcodes@</code> ; <code>amsmath</code> now has a definition usable in LuaTeX	8
v1.7	<code>\genfrac</code> : Adapt patch to changes in <code>amsmath</code>	8
v1.8	<code>@@_set_mathchar:NN</code> : <code>\lua_now_x:n</code> is now called <code>\lua_now:e</code>	4
	Stop using <code>\...:D</code> control sequences	4
	<code>\frac</code> : Stop using <code>\...:D</code> control sequences	5, 7
	<code>\genfrac</code> : Stop using <code>\...:D</code> control sequences	8
	General: Stop using <code>\...:D</code> control sequences	6
	<code>subarray</code> : Stop using <code>\...:D</code> control sequences	7
v1.9	<code>@begindocumenthook</code> : Don't patch newer versions of <code>amsmath</code>	6
	<code>\MT_cramped_internal:Nn</code> : Stop using <code>\...:D</code> control sequences	9
	<code>\frac</code> : Adapt to changes in LaTeX 2 _ε kernel	5
	<code>\l_@@_equal_mathchar</code> : Don't patch newer versions of <code>amsmath</code>	6
	General: Require 2020 version of LaTeX 2 _ε	2
	Use builtin LaTeX 2 _ε hooks if available	2
	<code>subarray</code> : Don't patch newer versions of <code>amsmath</code>	7
	Stop using <code>\...:D</code> control sequences	7

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols

<code>\\$</code>	16, 18
<code>\,</code>	258, 260
<code>\-</code>	108, 110, 115, 123, 126
<code>\=</code>	109, 111, 116, 124, 127
<code>@@_after_package:nn</code>	70, 87, 112
<code>@@_after_package_or_now:nn</code>	<u>86</u> , 120, 218, 256

<code>\@@_before_package:mn</code>	70, 106
<code>\@@_patch:N\Nnn</code>	40, 131, 219
<code>\@@_patch:c\Nnn</code>	40, 91, 172, 183
<code>\@@_restore_catcode:N</code>	11, 16
<code>\@@_set_mathchar:MN</code>	63, 108, 109, 126, 127, 260
<code>\@@_temp:w</code>	39, 44, 45, 53
<code>\@above</code>	192, 207
<code>\@over</code>	175, 179, 192, 205
<code>\@begindocumenthook</code>	120, 257
<code>\@ifl@t@r</code>	70
<code>\@ifpackagelater</code>	107, 121, 130
<code>\@ifpackageloaded</code>	87, 90
<code>\@mathstyle</code>	187, 199
<code>\@</code>	20, 27, 28, 29

A

<code>\AddToHook</code>	72, 75
<code>\advance</code>	138, 231, 245
<code>\alignmark</code>	167
<code>amsmath</code> (package)	1, 2, 5, 6, 11
<code>amsopn</code> (package)	10
<code>\AtBeginOfPackageFile</code>	80
<code>\AtEndOfPackage</code>	15
<code>\AtEndOfPackageFile</code>	83
<code>\AtEndPreamble</code>	89

B

<code>\baselineskip</code>	137, 138, 155
<code>\begingroup</code>	93, 175, 190
<code>\bgroup</code>	133, 144
<code>\binom</code>	2
<code>\box</code>	247
Breitfeld, Peter	9

C

<code>\c_@@_std_equal_mathcode_int</code>	102, 111
<code>\c_@@_std_minus_mathcode_int</code>	102, 110
<code>\c_group_begin_token</code>	151, 162
<code>\char_set_catcode:mn</code>	12
<code>\char_set_catcode_math_toggle:N</code>	18
<code>\char_set_mathcode:mn</code>	110, 111
<code>\char_value_catcode:n</code>	13
<code>\crrc</code>	148, 170
<code>\cs_generate_variant:Nn</code>	62
<code>\cs_if_eq:MNTF</code>	45
<code>\cs_if_exist:NF</code>	8
<code>\cs_if_exist:NT</code>	41
<code>\cs_new_eq:MN</code>	39
<code>\cs_new_nopar:Npn</code>	11
<code>\cs_new_protected_nopar:Npn</code>	40, 63, 71, 74, 79, 82, 86
<code>\cs_set:Npn</code>	91, 131, 172, 183
<code>\cs_set_eq:NN</code>	113, 114
<code>\cs_set_nopar:Npn</code>	220
<code>\cs_to_str:N</code>	251

D

<code>\default@tag</code>	136, 154
<code>\dimen@</code>	230, 231, 233, 245, 246
<code>\directlua</code>	10

`\displaystyle` 229

E

`\else` 192, 232, 236, 239
`\endgroup` 93, 175, 190
`\endinput` 37
`\ensuremath` 250
environments:
 `subarray` [130](#)
`etoolbox` (package) [2](#)
`\exp_args:Nx` 15
`expl3` (package) [2](#), [11](#)

F

`\fi` 145, 192, 241, 242, 244
`filehook` (package) [2](#)
`\fmtversion` 70
`\fontdimen` 137, 138, 140, 230, 231, 233
`\frac` [2](#), [89](#), [172](#)
functions:
 `module` [11](#)
 `print_class_fam_slot` [10](#), [278](#)
 `print_fam_slot` [11](#)
 `unpack` [10](#), [275](#)

G

`\genfrac` [2](#), [183](#)
`\genfrac@choice` 188, 195, 200, 213
`\group_begin:` 43, 97, 179, 203
`\group_end:` 48, 54, 97, 179, 203

H

`\hfil` 145, 147, 163, 169
`\ht` 245, 246

I

`\ialign` 143, 161
`icomma` (package) [1](#), [2](#), [9](#), [10](#)
`\ifx` 145, 192, 229, 234, 237
`\int_const:Nn` 102, 103
`\int_eval:n` 12, 66

L

`l3docstrip` (package) [10](#)
`l3kernel` (package) [10](#), [11](#)
`\l_@_equal_mathchar` [104](#), 114, 116
`\l_@_minus_mathchar` [104](#), 113, 115
`\Let@` 134, 152
`\lineskip` 140, 142, 159, 160
`\lineskiplimit` 142, 160
`\lua_now:e` 65
`luatex-required` (message) [19](#)
`luatexbase` (package) [11](#)
`luatexbase-modutils` (package) [10](#)

M

`\m@th` 146, 165, 223
`macro-expected` (message) [23](#)
`\mathchardef` 123, 124, 258

<code>\mathcode</code>	123, 124, 258
<code>\mathcomma</code>	256
<code>\mathstyle</code>	2
mathtools (package)	1, 2, 8
messages:	
<code>luatex-required</code>	19
<code>macro-expected</code>	23
<code>patch-macro</code>	32
<code>wrong-meaning</code>	26
module (function)	11
<code>\msg_error:mn</code>	36
<code>\msg_info:nxx</code>	46
<code>\msg_new:nnn</code>	19, 23, 26, 32
<code>\msg_warning:nxx</code>	57
<code>\msg_warning:nnxxx</code>	51
<code>\MT_cramped_internal:Nn</code>	218

N

nath (package)	5
<code>\NeedsTeXFormat</code>	3
<code>\newluabytecode</code>	8
<code>\nulldelimiterspace</code>	225

O

<code>\over</code>	93, 97
--------------------------	--------

P

packages:	
amsmath	1, 2, 5, 6, 11
amsopn	10
etoolbox	2
expl3	2, 11
filehook	2
icomma	1, 2, 9, 10
l3docstrip	10
l3kernel	10, 11
luatexbase	11
luatexbase-modutils	10
mathtools	1, 2, 8
nath	5
unicode-math	1, 2
<code>patch-macro</code> (message)	32
<code>\prg_do_nothing:</code>	39
<code>print_class_fam_slot</code> (function)	10, 278
<code>print_fam_slot</code> (function)	11
<code>\ProvidesExplPackage</code>	5

R

<code>\radical</code>	226
<code>\relax</code>	123, 124, 192
<code>\RequirePackage</code>	4, 7, 9, 78
<code>\restore@math@cr</code>	135, 153
Robertson, Will	1

S

<code>\sbox</code>	221
<code>\scan_stop:</code>	68, 207
<code>\scriptfont</code>	137, 138, 140, 238
<code>\scriptscriptfont</code>	240

<code>\scriptstyle</code>	146, 156, 157, 159, 166, 237
<code>\skip_set:Mn</code>	155
<code>\std@equal</code>	114, 124, 127
<code>\std@equals</code>	113
<code>\std@minus</code>	113, 123, 126
<code>\subarray</code>	131
<code>subarray (environment)</code>	130
<code>\sys_if_engine_luatex:F</code>	35

T

<code>\textfont</code>	230, 231, 235
<code>\textstyle</code>	234
<code>\thr@@</code>	140
<code>\tl_if_empty:NTF</code>	204
<code>\tl_new:N</code>	104, 105
<code>\tl_replace_once:Nmn</code>	122, 257
<code>\token_if_eq_meaning:NNT</code>	163
<code>\token_if_macro:NTF</code>	42
<code>\token_to_meaning:N</code>	52, 53
<code>\token_to_str:N</code>	47, 52, 58
<code>\tw@</code>	137, 138

U

<code>\Umathchardef</code>	64
<code>\Umathcodenum</code>	115, 116
<code>\Umathstackdenomdown</code>	157
<code>\Umathstacknumup</code>	156
<code>\Umathstackvgap</code>	159
<code>unicode-math (package)</code>	1, 2
<code>unpack (function)</code>	10, 275
<code>\use:c</code>	251
<code>\Ustack</code>	97, 179, 202
<code>\Ustartmath</code>	164
<code>\Ustopmath</code>	168

V

<code>\vcenter</code>	132, 150
-----------------------	----------

W

<code>wrong-meaning (message)</code>	26
--------------------------------------	----

Z

<code>\z@</code>	221, 225, 226, 245, 246, 247
------------------	------------------------------