

The physics3 package*

Mingyu Xia[†]

Released 2026-02-07 v0.2A

Abstract

This is the document for physics3 package, which defines commands for typesetting math formulae faster and more simply. physics3 is a modularized package, each module provides its own function¹.

Contents

1	Introduction	2
2	Modules of physics3	3
A	For the legacy physics users	12
B	Code Implementation	17
	Index	47

*<https://github.com/myhsia/physics3>, <https://ctan.org/pkg/physics3>

[†]xiamingyu@westlake.edu.cn

¹If you are a user of the legacy physics package, see Section A before you start.

1 Introduction

The physics3 package aims to provide a bundle of commands for typesetting math faster in different modules. The commands provided by physics3 and its different modules are designed to be short and easy to memorize.

1.1 Packages required

The physics3 package itself only requires the keyval package, which is part of the latex-graphics bundle. Almost every L^AT_EX distribution will include this bundle.

Different modules of physics3 might require different packages. It will be explained in the following sections that which module requires which package.

The physics3 package requires L^AT_EX 2_ε kernel released after 2020/10. Please make sure that your L^AT_EX distribution is not too old.

1.2 Loading physics3 and its modules

Just like loading any package, write

```
\usepackage{physics3}
```

in the preamble to load the physics3 package. In the current version, physics3 doesn't provide a package option.

physics3 itself doesn't provide many features. You need to load different modules of physics3 by using the \usephxmodu command only *after* loading this package in the preamble to have different features applied to your document.

```
\usephxmodu <options> {<module>}
```

The usage of \usephxmodu is similar to \usepackage, so you can load several modules in one line. For example,

```
\usephxmodu [tightbraces = true] {ab}  
\usephxmodu {ab.braket, doubleprod}
```


The options of a physics3 module can be a comma-separated key-value list. These two lines load the ab module with option `tightbraces = true` and load `ab.braket` and `doubleprod` modules.

The following section introduce all the user-level modules of physics3. The modules have similar names like `<module>.legacy` are designed to provide solutions to maintain documents written with the legacy physics package. It's not suggested to use them in a new document.

<pre> \$ \ab (\frac{1}{2}) \$ \quad \$ \ab [\frac{1}{2}] \$ \quad \$ \ab\{ \frac{1}{2} \} \$ </pre>	$\left(\frac{1}{2}\right) \quad \left[\frac{1}{2}\right] \quad \left\{\frac{1}{2}\right\}$
---	--

You can also write a command from `\big` to `\Biggg` between `\ab` and the first delimiter, which means to specify the size of delimiters manually. Also, you can write a star (*) between `\ab` and the first delimiter, to prevent `\ab` from setting the size of delimiters. For example,

<pre> \$ \ab <\frac{1}{2}> \$ \quad \$ \ab\biggg \frac{1}{2} \$ \quad \$ \ab* \ \frac{1}{2}\ \$ </pre>	$\left\langle\frac{1}{2}\right\rangle \quad \left \frac{1}{2}\right \quad \left\ \frac{1}{2}\right\ $
--	--

 Always remember, do not put an `\ab` separately at the end of math mode like `$$\ab$`, because `\ab` will try to absorb the following math shift character (\$) as its argument.

TeXhackers note: The `ab` module uses “document commands” module of $\text{\LaTeX}_{2\epsilon}$ kernel (source file: `lcmd.dtx`). This $\text{\LaTeX}_{2\epsilon}$ kernel module provides a document-level command parser. `\ab` is a complex encapsulation of some internal document-level commands. Take an example, if you define a document-level command like this:

```
\NewDocumentCommand \foo { r() } {:#1::}
```

You can write `\foo(bar)` legally, but `\foo()` will be regarded illegal when you write another document-level command or end the paragraph. Similarly, things like `\ab()` will also cause errors.

The `ab` module also provides `\Xab` commands, where `X` can be `p`, `b`, `B`, `a`, `v` and `V`. These commands take a normal argument but not an argument delimited with paired delimiters. For example,

<pre> \def\0{\frac{1}{2}} \$ \pab{\0} \$ \$ \bab{\0} \$ \$ \Bab{\0} \$ \$ \aab{\0} \$ \$ \vab{\0} \$ \$ \Vab{\0} \$ </pre>	$\left(\frac{1}{2}\right) \quad \left[\frac{1}{2}\right] \quad \left\{\frac{1}{2}\right\} \quad \left\langle\frac{1}{2}\right\rangle \quad \left \frac{1}{2}\right \quad \left\ \frac{1}{2}\right\ $
--	---

These `\Xab` commands can take an optional star and an optional [`biggg`] argument. Star stands for using the default sizes. For example,

<pre> \def\0{n+\frac{1}{2}} \$ \pab[Big]{\0} \$ \quad \$ \bab*{\0} \$ </pre>	$\left(n + \frac{1}{2}\right) \quad \left[n + \frac{1}{2}\right]$
--	---

`tightbraces = <true|false>`: Influences whether thin skips are reserved around the paired delimiters. It only works with the automatically sized delimiters.

2.3 The ab.braket and braket module — Dirac bra-ket notation

This two modules both contain four basic commands

`\bra`, `\ket`, `\braket`, `\ketbra`

These commands in the two modules share the same goal, but differ in syntax.

`ab.braket` A star (*) or a size command can follow these commands, which are similar to the syntaxes of `\ab` module. The size commands can take the control sequences:

`\big`, `\Big`, `\bigg`, `\Bigg`, `\biggg` or `\Biggg`.


`braket` A star (*) or a square-bracket-delimited size option, the size option can take the following values:

`big`, `Big`, `bigg`, `Bigg`, `biggg` or `Biggg`.

The star in the two modules stands for “do not size the bra-ket automatically”. Four basic commands’ syntaxes are described as follows respectively. **Please notice that the two module are conflict with each other, so don’t use them together.** One can choose one module according to personal habit.

<hr/>	<code>\bra</code>	<code>\bra</code>	<code>< * or size command ></code>	<code><{subformula} </code>	<code>--></code>	<code>ab.braket</code>	module
	<code>\ket</code>	<code>\ket</code>	<code>< * or size command ></code>	<code> {subformula}></code>	<code>--></code>	<code>ab.braket</code>	module
<hr/>		<code>\bra</code>	<code>< * or [(size option)] ></code>	<code>{subformula}</code>	<code>--></code>	<code>braket</code>	module
		<code>\ket</code>	<code>< * or [(size option)] ></code>	<code>{subformula}</code>	<code>--></code>	<code>braket</code>	module

- In `ab.braket` module, the argument of `\bra` should be delimited with `<` and `|`, and the argument of `\ket` should be delimited with `|` and `>`.

 If you want to write “`>`” and “`<`” for relations in the argument of `\bra` and `\ket`, you can write `\mathrel{>}` and `\mathrel{<}`.

- In `braket` module, `\bra` and `\ket` take one mandatory argument, which should be braced with `{` and `}`.

For example, in `ab.braket` module,

<pre> \def\0{\frac\phi2} \$ \bra<\0 \$ \quad \$ \bra*<\0 \$% \quad \$ \bra\Big<\phi \$\\[1ex] \$ \ket \0> \$ \quad \$ \ket* \0> \$% \quad \$ \ket\Big \psi> \$ </pre>	$\left\langle \frac{\phi}{2} \right \quad \left\langle \frac{\phi}{2} \right \quad \left\langle \phi \right $ $\left \frac{\phi}{2} \right\rangle \quad \left \frac{\phi}{2} \right\rangle \quad \left \psi \right\rangle$
--	---

and in braket module

<pre>\def\0{\frac\phi2} \$ \bra \0 \$\quad \$ \bra* \0 \$% \quad \$ \bra[Big] \psi \$\\[1ex] \$ \ket \0 \$\quad \$ \ket* \0 \$% \quad \$ \ket[Big] \psi \$</pre>	$\left\langle \frac{\phi}{2} \middle \quad \left\langle \frac{\phi}{2} \middle \quad \langle \psi \middle \right.$ $\left \frac{\phi}{2} \right\rangle \quad \left \frac{\phi}{2} \right\rangle \quad \psi\rangle$
--	--

	<code>\braket</code>	<code>\braket <* or size command> <subformula></code>	<code>--> ab.braket module</code>
	<code>\braket</code>	<code><* or [size option, i]> i*{subformula}</code>	<code>--> braket module</code>

- In `ab.braket` module, the argument of `\braket` should be delimited with `<` and `>`, every “|” will be regarded as an extensible vertical bar in the `<subformula>` argument.



If you want to write “>” and “<” for relations in the argument of `\braket` and `\ketbra` in this module, you can write `\>` and `\<`. It is quite different from `\mathrel{>}` or `\mathrel{<}` because in these commands’ argument, `>` and `<` will be redefined.

- In `braket` module, the `\braket` command, in default, can take two arguments. If you want `\braket` to take one or three arguments, you can specific the number of arguments by number `i` in the square bracket, separate from the size option with a comma.

For example, in `ab.braket` and `braket` module,

<pre>\def\0{\frac\phi2} \$ \braket < \hat A > \$\quad \$ \braket <\0 \hat A \psi> \$\quad \$ \braket\big <\0 \psi> \$\\[1ex] \$ \braket* <\0 \psi> \$\quad \$ \braket\Big <\0 \psi> \$\quad \$ \ab \{\braket<\psi \hat A \psi>\} \$</pre>	$\langle \hat{A} \rangle \quad \left\langle \frac{\phi}{2} \middle \hat{A} \middle \psi \right\rangle \quad \left\langle \frac{\phi}{2} \middle \psi \right\rangle$ $\left\langle \frac{\phi}{2} \middle \psi \right\rangle \quad \left\langle \frac{\phi}{2} \middle \psi \right\rangle \quad \left \langle \psi \hat{A} \psi \rangle \right $
---	---



Commands from `ab.braket` should not be placed barely in `\ab|subformula|` or errors will arise. So, adding braces like the last line in the example above.

<pre>\def\0{\frac\phi2} \$ \braket [1] {\hat A} \$\quad \$ \braket [3] {\0}{\hat A}{\psi} \$\quad \$ \braket[big] {\0}{\psi} \$\\[1ex] \$ \braket* {\0}{\psi} \$\quad \$ \braket[Big] {\0}{\psi} \$\quad \$ \ab \{\braket[3]\psi{\hat A}\psi\} \$</pre>	$\langle \hat{A} \rangle \quad \left\langle \frac{\phi}{2} \middle \hat{A} \middle \psi \right\rangle \quad \left\langle \frac{\phi}{2} \middle \psi \right\rangle$ $\left\langle \frac{\phi}{2} \middle \psi \right\rangle \quad \left\langle \frac{\phi}{2} \middle \psi \right\rangle \quad \left \langle \psi \hat{A} \psi \rangle \right $
---	---

<u>\ketbra</u>	<code>\ketbra <* or size command></code>	<code> <subformula₁></code>	
	<code><optional formula></code>	<code><subformula₂ </code>	--> ab.braket module
	<code>\ketbra <* or [size option]></code>	<code>{<subformula₁></code>	
	<code><optional formula></code>	<code>{<subformula₂></code>	--> braket module

- In ab.braket module, the argument of \ketbra should be delimited with | and |, > and < will be regarded as extensible) and (in the argument.
- In braket module, the \ketbra command takes two mandatory arguments by default, and one optional argument between the two mandatory arguments () and ().

For example, in ab.braket and braket module,

<pre>\def\0{\frac\phi2} \$ \ketbra \0 >< \psi \$\quad \$ \ketbra* \0 >< \psi \$\quad \$ \ketbra\Big \0 >_x^y< \psi \$</pre>	$\left \frac{\phi}{2}\right\rangle\langle\psi \quad \left \frac{\phi}{2}\right\rangle\langle\psi \quad \left \frac{\phi}{2}\right\rangle_x^y\langle\psi $
<pre>\def\0{\frac\phi2} \$ \ketbra {0} {\psi} \$\quad \$ \ketbra* {0} {\psi} \$\quad \$ \ketbra [Big] {0} [_x^y] {\psi} \$</pre>	$\left \frac{\phi}{2}\right\rangle\langle\psi \quad \left \frac{\phi}{2}\right\rangle\langle\psi \quad \left \frac{\phi}{2}\right\rangle_x^y\langle\psi $

For convince, the two modules also contain three advanced commands

`\bknorm`, `\kbproj`, `\expval`

These commands have the same syntax in the two modules.

<u>\bknorm</u>	<code>\bknorm <*> [<i>size</i>] {<i>basis</i>}</code>
<u>\kbproj</u>	<code>\kbproj <*> [<i>size</i>] {<i>basis</i>}</code>
<u>\expval</u>	<code>\expval <*> [<i>basis, size</i>] {<i>operator</i>}</code>

For example,

<pre>\def \0{\frac\alpha2} \edef\1{\uparrow,\downarrow} \$ \bknorm \0 \$\quad \$ \bknorm [big]\0 \$ \quad \$ \bknorm*\0 \$\[2ex] \$ \kbproj \0 \$\quad \$ \kbproj [big]\0 \$ \quad \$ \kbproj*\0 \$\[2ex] \$ \expval [Big, \1] {\hat A} \$\quad \$ \expval*[Big, \psi]\0 \$</pre>	$\left\langle\frac{\alpha}{2}\middle \frac{\alpha}{2}\right\rangle \quad \left\langle\frac{\alpha}{2}\middle \frac{\alpha}{2}\right\rangle \quad \left\langle\frac{\alpha}{2}\middle \frac{\alpha}{2}\right\rangle$ $\left \frac{\alpha}{2}\right\rangle\langle\frac{\alpha}{2} \quad \left \frac{\alpha}{2}\right\rangle\langle\frac{\alpha}{2} \quad \left \frac{\alpha}{2}\right\rangle\langle\frac{\alpha}{2} $ $\langle\uparrow,\downarrow \hat{A} \uparrow,\downarrow\rangle \quad \langle\psi \frac{\alpha}{2} \psi\rangle$
---	--

2.4 The doubleprod module – tensors’ double product operator

This module provides the `\doublecross` and `\doubledot` commands, which are regarded as binary operators by \TeX . Take an example of this module:

<code>\$ A \doublecross B \doubledot C \$</code>	$A \times B : C$
--	------------------

crossscale, **dotscale** = $\langle fp \ num \rangle$: Scales of “ \times ” and “ \cdot ” (Default: .8 and 1).

crossopenup, **dotopenup** = $\langle fp \ num \rangle$: Spaces between “ \times ”s and “ \cdot ”s = $\langle fp \ num \rangle * \text{font size}$ (Default: .02 and .2).

doubledot, **crosssymbol** = $\langle symbol \rangle$: Pieces of `\doublecross` and `\doubledot` (Default: `\times`, `\cdot`).

Their default values can be configured in module option. For example,

```
\usephxmodu[crossscale = .75, dotscale = 1.2,
crossopenup = .05, dotopenup = .25]{doubleprod}
```

2.5 The diagmat module – simple diagonal matrices

This module requires the `mathtools` package and provides six `*diagmat` commands:

<code>\diagmat</code>	$\langle delimiter \ type \rangle \text{diagmat} \langle * \rangle$ [$\langle options \rangle$] { $\langle diag \rangle$ }
<code>\pdiagmat</code>	where $\langle diag \rangle$ is the diagonal of the diagonal matrix. The entries should be separated
<code>\bdiagmat</code>	by commas. Prefixes like p, b, V have the same meanings as <code>pmatrix</code> , <code>bmatrix</code> and
<code>\Bdiagmat</code>	<code>Vmatrix</code> in the <code>mathtools</code> package, a star (*) follows the <code>*diagmat</code> commands corre-
<code>\vdiagmat</code>	spond to <code>*smallmatrix</code> in the <code>mathtools</code> package.
<code>\Vdiagmat</code>	

empty = $\langle token \ list \rangle$: Value of `\diagmat`’s empty entries (Default is blank).

align = $\langle l \ | \ c \ | \ r \rangle$: Align of `\diagmat`’s diagonal entries (Default: r).

Their default values can be configured in the module option like this:

```
\usephxmodu[empty = \cdot, align = r]{diagmat}
```

For example,

<pre>\$ \pdiagmat [empty = \mathbf{0}] { \diagmat*[align = r]{1, \sqrt[3]{4}}, \diagmat*{a, b} } \$, \$ \bdiagmat* {1, -1, -1, -1} \$</pre>	$\left(\begin{array}{cc} 1 & \mathbf{0} \\ \mathbf{0} & \sqrt[3]{4} \end{array} \right),$ $\left[\begin{array}{cccc} 1 & & & \\ & -1 & & \\ & & -1 & \\ & & & -1 \end{array} \right]$
---	---

2.6 The xmat module – matrices with formatted entries

This module requires the mathtools package and provides six `*xmat` commands:

<code>\xmat</code>	<code>\langle delimiter type \rangle xmat \langle * \rangle [\langle options \rangle] \langle entry \rangle \langle row label \rangle \langle col label \rangle</code>		
<code>\pxmat</code>	Prefixes like p, b, V have the same meanings as pmatrix, bmatrix and Vmatrix		
<code>\bxmat</code>	in the mathtools package, a star (*) follows the <code>*xmat</code> commands correspond to		
<code>\Bxmat</code>	<code>*smallmatrix</code> in the mathtools package. For example,		
<code>\vxmat</code>			
<code>\Vxmat</code>	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td><code>\$ \vxmat*{a}{2}{3} \$</code></td> <td>$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{vmatrix}$</td> </tr> </table>	<code>\$ \vxmat*{a}{2}{3} \$</code>	$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{vmatrix}$
<code>\$ \vxmat*{a}{2}{3} \$</code>	$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{vmatrix}$		

align = $\langle l | c | r \rangle$: Similarly to the diagmat module.

If $\langle rows/col label \rangle$ contains non-digit characters, dots will be added. For example,

<code>\$ \bxmat*[showtop = 1, showleft = 2, align = l]{X}{m}{n} \$</code>	$\begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{m1} & X_{m2} & \cdots & X_{mn} \end{bmatrix}$
---	---

showtop, **showleft** = $\langle int num \rangle$: Numbers of rows and columns to be shown at the top and the left side (Default: `MaxMatrixCols - 2`).

Their default values can be configured in the module option like this:

`\usephxmodu[showtop = 1, showleft = 2]{xmat}`

This will also influence “`\xmat`”s with digital $\langle row label \rangle$ and $\langle col label \rangle$ they are larger than the values corresponding to `showtop` and `showleft`. For example,

<code>\$ \pxmat*[showtop = 1, showleft = 2]{A}{8}{8} \$</code>	$\begin{pmatrix} A_{11} & A_{12} & \cdots & A_{18} \\ \vdots & \vdots & \ddots & \vdots \\ A_{81} & A_{82} & \cdots & A_{88} \end{pmatrix}$
--	---

The `format` option allows users to use a new entry format.

format = $\langle custom entry format in terms of \#1, \#2, and \#3 \rangle$:

\#1: The common entry, or the first mandatory $\langle entry \rangle$ argument of `\xmat`;

\#2: The row index;

\#3: The column index.

This option should be only used in the optional argument of `\xmat`. For example,

<code>\$ \Bxmat*[showtop = 2, showleft = 1, format = \texttt{\#1[\#2][\#3]}]xmn \$</code>	$\left\{ \begin{array}{l} x[1][1] \cdots x[1][n] \\ x[2][1] \cdots x[2][n] \\ \vdots \\ x[m][1] \cdots x[m][n] \end{array} \right\}$
---	--

2.7 The operator module – log-like and nabla-related operators

This module provides a series of commands for log-like operators and some commands for nabla-related operators (∇)

```

\asin  \acos  \atan  \acsc  \asec  \acot  \erf
\erfc  \Tr    \tr    \rank  \Res   \res   \Re
\Im    \PV    \pv    \upe   \iu    \grad  \curl
\div   \laplacian  \identity  \Identity

```

- The first fifteen commands yield what they look like in math mode
- `\PV` yields “ \mathcal{P} ” as an ordinary symbol and `\pv` yields “p.v.”.
- `\Re` and `\Im` are redefined as “Re” and “Im”. \Re and \Im are redefined as `\Resymbol` and `\Imsymbol`, in default.
- The “ \div ” symbol was redefined as `\divsymbol`.
- This module requires the `fixdif` package with file date 2023/01/31 at minimum.

ReIm = $\langle true | false \rangle$: Determines whether to redefine `\Re` and `\Im`. If you want to reserve the definition of `\Re` and `\Im`, you can write like this:

```
\usephxmodu[ReIm = false]{operator}
```

This module also provides the `\identity` and `\Identity` commands for inputting the double-stroke “i” (I) and “I” (I) respectively.

```
\identity \identity [font family] \Identity [font family]
```

`\Identity` The optional argument can config the font for the double-stroke glyphs, with the default value `bbold`. One can also config it with the module option.

identity = $\langle font family \rangle$: Default font family for the double-stroke `\identity` and `\Identity`.

For example,

<pre> % \usephxmodu[identity = bbold]{operator} % \usephxmodu{braket} \$ \asin x \$, \$ \PV f(z) = \pv f(z) \$\$\[1ex] \$ \erfc(x) = 1 - \erf(x) \$\$\[1ex] \$ \identity = \Identity[bbold] = \int d^2 \alpha \ \text{proj} \ \alpha / \pi \$\$\[1ex] \$ \Re z = \Im z = \exp(\iu \pi) = -1 \$\$\[1ex] \$ \grad V\$, \$ \div (x,y,z)\$, \$ \curl(x,y,z) \$ </pre>	$\operatorname{asin} x, \mathcal{P}f(z) = \text{p.v. } f(z)$ $\operatorname{erfc}(x) = 1 - \operatorname{erf}(x)$ $1 = I = \int d^2 \alpha \alpha\rangle \langle \alpha / \pi$ $\operatorname{Re} z = \operatorname{Im} z = \exp(i\pi) = -1$ $\nabla V, \nabla \cdot (x, y, z), \nabla \times (x, y, z)$
--	--

2.8 The `ab.legacy` module

`\abs` $\langle cmd \rangle * \langle biggg \rangle \{ \langle subformula \rangle \}$.

`\norm` This module provides the left commands. They share the same syntax as `Star` and
`\eval` $\langle biggg \rangle$ are optional. `Star` stands for “use the default size”. For example,
`\peval`

<code>\beval</code> <code>\order</code>	<pre> \def\0{1+\frac12} \$ \abs\0 \$, \$ \norm[Big]\0 \$, \$ \order*\0 \$ \def\0{1+\frac{x}{2}} \$ \eval {\0}_a^b \$, \$ \peval* {\0}_a^b \$, \$ \beval[big]{\0}_a^b \$ </pre>	$\left 1 + \frac{1}{2} \right , \left\ 1 + \frac{1}{2} \right\ , \mathcal{O}\left(1 + \frac{1}{2}\right)$ $1 + \frac{x}{2} \Big _a^b, \left(1 + \frac{x}{2}\right)_a^b, \left[1 + \frac{x}{2}\right]_a^b$
--	--	---

You can set the “order” symbol in this module through the `order` option like this:

```
\usephxmodu[order = 0]{ab.legacy}
```

Then $\$ \backslash order\{N\} \$$ yields $O(N)$.

2.9 The `qtext.legacy` module

The `qtext.legacy` module provides the $\backslash q\langle foo \rangle$ commands for quad-wrapped texts. These commands have the same syntax as `physics`. For example,

<pre> \$ A \qq {foo bar} B \$\\ \$ A \qq*{foo bar} B \$\\ \$ C \qcc D \qcc* E \$\\ \$ F \qif G \qthen H \$ </pre>	<pre> A foo bar B Afoo bar B C c.c Dc.c E F if G then H </pre>
---	--

All the commands described in §2.4 of [physics documentation](#) are supported when using `qtext.legacy` module, but I don’t recommend using this module unless you are maintaining a document written with `physics`’s $\backslash q\langle foo \rangle$ commands.

A For the legacy physics users

This section describes physics3 package for those who are used to the physics package, which is only a simple reference manual for:

- Frequent users of the legacy physics package;
- Those who have to maintain a document written with physics;
- Users who failed to use unicode-math with physics.

In this section, the modules of physics3 will be introduced in the same order as the physics documentation.

A.1 Legacy problems with physics package

The physics package provides `\qty` command for automatic-sizing braces. The `\qty` command would cause conflict with the `siunitx` package, which provides a unified method to typeset numbers and units correctly.

Besides, after you loaded physics, when you type `\homework` you will get Maxwell equations and Schrödinger equation. The `\homework` command is “declared” in `physics.sty` but it was not described in the documentation. That is, if you have defined `\homework` before loading physics package, physics would overwrite the definition “silently”.

The vector-notation part of physics uses `amsmath`’s (more exactly, `amsbsy.sty`’s) `\boldsymbol` command to generate bold vectors. Commands for cross/dot product are defined with `\boldsymbol`. `\boldsymbol` uses `\mathversion`, a $\TeX 2_{\epsilon}$ kernel command that works well with traditional TFM-based fonts but fails when using unicode-math.

In the definition of `\imat`, `\xmat`, `\dmat` and `\admat` commands from physics, there is a `\newtoks` command which allocates a token list register and two `\newcount` commands allocating two count registers. Every time you write a command like `\imat` in your document, then one token list register and two count registers will be wasted. What’s even worse is that, if you wrote really too many matrix commands from physics (for example, 32767 `\imats` in Lua \TeX), there’d be no room for a new `\count`.

physics integrated all the functions in one file (`physics.sty`), that is, you cannot load one of the total seven parts of functions; you have to load the seven parts altogether, even included the extra `\homework` command we mentioned in the first paragraph.

Moreover, the code of `physics.sty` “abuses” the g-type arguments of `xparse` package. Therefore the syntax of physics package looks kind of weird. See [here](#) for more.

A.2 Loading physics3

The physics3 package includes different modules, among which every module focuses on one single function.

Write the following line in the preamble to load physics3:

```
\usepackage{physics3}
```

But this is not enough. physics3 contains different modules. If you want to load any module of physics3, write this line after loading physics3 package:

```
\usephxmodu{<module list>}
```

For example, “\usephxmodu{ab, doubleprod}” loads the ab module and the doubleprod module.

You can also load a module with options:

```
\usephxmodu[<option list>]{<module>}
```

For example, “\usephxmodu[legacy]{ab}” loads ab with the option “legacy”.



Attention, if you used any font package in your document, remember that physics3 requires to be loaded *after* font packages.

A.3 List of commands

A.3.1 Automatic bracing

As mentioned in §A.1, the \qty command from physics will conflict with siunitx. The command for automatic braces in physics3 is \ab, short for **automatic braces**.

physics also provides the following commands:

```
\abs \norm \eval \order \comm \acomm \pb
```



These commands are not originally supported by physics3, but the first four commands can be used through the ab.legacy module of physics3:

```
\usephxmodu{ab.legacy}
```

Users of the legacy physics package should notice that the syntax of \eval has been changed. The ab.legacy module abandoned the \eval(foo)-like syntax. The new \eval’s syntax is just like other commands in this module. There are also two variants of \eval – \peval and \beval.

The `\comm`, `\acomm` and `\pb` (Poisson bracket) are not supported. But you can write like `\ab{foo,baz}` or `\bab{foo,baz}` instead.

By the way, you can set the “order” symbol in `ab.legacy` through the `order` option like this:

```
\usephxmodu[order = 0]{ab.legacy}
```

Then `\order(N)` yields $O(N)$.

A.3.2 Vector notation

Unfortunately, there is not a plan for `physics3` to support this part of physics completely, but the rest of this section will show some methods to maintain the document written with `physics`.

The `\vb(*)`, `\va(*)` and `\vu(*)` are not supported in any module of `physics3`. But these commands can be defined by copying the following lines below and pasting them in the preamble:

```
\makeatletter
\newcommand\vb{\@ifstar\boldsymbol\mathbf}
\newcommand\va[1]{\@ifstar{\vec{#1}}{\vec{\mathrm{#1}}}}
\newcommand\vu[1]{%
\@ifstar{\hat{\boldsymbol{#1}}}{\hat{\mathbf{#1}}}}
\makeatother
```

The `\boldsymbol` command requires the `amsmath` or `bm` package. If you prefer to use `bm`, you can also use the `\bm` command. What’s more, if you tried the commands above, you might find that, the result of `\va` above is different from that of `physics`. This is because, if you choose to present a vector in bold, there’s almost no need to put a `\vec` (↵) sign above it.

However, the method above may not work well with `unicode-math` because there are so many OpenType math fonts without a bold version. When using `unicode-math`, it’s recommended to use `\symbf` and `\symbfit` for a separate vector².

The `\vdot` and `\cross` commands are not supported in any module of `physics3`. Actually, there is no need to use a bold “ \cdot ” or “ \times ” for the products of two vectors. Using `\cdot` and `\times` is enough.

The commands related to “ ∇ ” are supported through `operator` module. These commands are `\grad`, `\div` and `\curl`. These commands should not be put in the end

²See [Issue #1991](#) in the `LaTeX 2ε` repository on [GitHub](#), this bug will be resolved someday.

of a math formula either (just like `\ab`). Notice that the former `\div` command for a “÷” (if there exists one) is redefined as `\divsymbol`.

Actually, the nabla-related commands end with `\ab`. Thus, the subformula after these commands can be delimited with `()`, `[]` and `\{\}`.

The operator requires the `fixdif` package at least version 2.0 (file date on or after 2023/01/31).

A.3.3 Operators

There’s no plan for `physics3` to support this part of `physics` completely. The syntax in this part of `physics` (such as `\tan[2](x)`) abuses `xparse`.

It’s suggested to write like this if you used the `ab` module:

```
$ \sin^2 \ab( \frac{\alpha}{2} ) $
```

Rather than take the superscript as an optional argument of the command of log-like functions.

The `physics` package provides a bundle of commands for log-like functions that have not been defined in the $\text{\LaTeX}_{2\epsilon}$ kernel. They can be used with the `operator` module; this module does not support the syntax of `physics` either.

The `\Re` and `\Im` commands are redefined as operators “Re” and “Im”, while \Re and \Im are reserved as `\Resymbol` and `\Imsymbol`. \Re and \Im are ordinary symbols but `Re` and `Im` are operators.

A.3.4 Quick quad text

All the commands described in §2.4 of [physics documentation](#) are supported when using `qtext.legacy` module. Click here to see the `qtext.legacy` module.

A.3.5 Derivatives

There is no plan for `physics3` to support this part of `physics`. If you want to typeset the differential operators on a better sense, you can try the `fixdif` package; if you want an easy way to type derivatives, you can try the `derivative` package. These two packages can be used together. For example,

<pre>% \usepackage{fixdif, derivative} \$ \pdv{f}{x,y,z} \d x \$\[1ex] Math ($\d x$) v.s. \ Text ($\d x$)</pre>	$\frac{\partial^3 f}{\partial x \partial y \partial z} dx$ <p>Math (dx) v.s. Text ($\d x$)</p>
---	--

Here are the documentations of [fixdif](#) and [derivative](#). `fixdif`’s commands behave better in superscripts and subscripts.

A.3.6 Dirac bra-ket notation

There are two solutions to Dirac bra-ket in physics3 – `ab.braket` and `braket`. These two modules are *not* compatible and neither of them supports physics’s syntax completely. Click [here](#) to see the `ab.braket` module and [here](#) to see the `braket` module.

A.3.7 Matrix macros

Unfortunately, physics3 do not support the `\mqty` command from physics. If you are used to this command, you can write like this:

```
\newcommand\mqty[1]{\begin{matrix}#1\end{matrix}}
\newcommand\pmqty[1]{\begin{pmatrix}#1\end{pmatrix}}
 $\ab(\mqty{foo})$  or  $\pmqty{foo}$ 
```

These are equal to physics’s `\mqty(foo)` (require `amsmath`).

physics3’s `diagmat` module provides `\diagmat` command for diagonal matrices. `\pdiagmat`, `\bdiagmat`, `\Bdiagmat`, `\vdiagmat` and `\Vdiagmat` are also available.

physics3’s `xmat` module provides `\xmat` command for matrices with formatted entries. `\pxmat`, `\bxmat`, `\Bxmat`, `\vxmat` and `\Vxmat` are also available.

B Code Implementation

B.1 The physics3 package

```
1 <*package>
```

We use `phx` as the namespace for physics3 modules.

```
2 <@@=phx>
3 \NeedsTeXFormat{LaTeX2e}[2020/10/01]
4 \def \phx@date {2026-02-07}
5 \def \phx@version {v0.2A}
6 \ProvidesExplPackage {physics3} {\phx@date} {\phx@version}
7 {Tools for typesetting math for physics.}
```

B.1.1 Common variables

```
\l__phx_tmpa_box \phx@temp<register type><a or b>
```

Some \LaTeX $z\epsilon$ variables starting with “`\phx@temp`”. These variables can be shared by any module of physics3.

```
8 \box_new:N \l__phx_tmpa_box
```

(End of definition for `\l__phx_tmpa_box`.)

B.1.2 Package requirements and module-loading methods

physics3 requires `keyval` (part of the graphics bundle) to process options of modules.

```
9 \RequirePackage{keyval}
10 \def\phx@true{true}
11 \def\phx@false{false}
```

```
\__phx_define_key:nnnn \phx@define@key {<module>} {<key>} [<default value>] {<code>}
\__phx_setkeys:nn \phx@setkeys {<module>} {<key-val list>}
\__phx_processkeyopt:n \phx@processkeyopt {<module>}
```

The position of `\phx@processkeyopt` in a physics3 module is just the same as the position of `\ProcessOptions` in a regular \LaTeX package.

```
12 \cs_new_nopar:Npn \__phx_define_key:nnnn #1#2#3#4
13 { \define@key {phx-#1} {#2} [#3] {#4} }
14 \cs_new_nopar:Npn \__phx_setkeys:nn #1#2
15 { \setkeys {phx-#1} {#2} }
16 \cs_new:Npn \__phx_processkeyopt:n #1
17 {
18 \let\reserved@a\@empty
19 \edef\reserved@a{\@optionlist{\@currname.\@currentt}}%
```

```

20 \edef\reserved@a{\noexpand\phx@setkeys{#1}{\reserved@a}}%
21 \reserved@a% the next line thanks to 'geometry'
22 \AtEndOfPackage{\let\@unprocessedoptions\relax}
23 }
24 \cs_set_eq:NN \phx@define@key \__phx_define_key:nnnn
25 \cs_set_eq:NN \phx@setkeys \__phx_setkeys:nn
26 \cs_set_eq:NN \phx@processkeyopt \__phx_processkeyopt:n

```

(End of definition for `__phx_define_key:nnnn`, `__phx_setkeys:nn`, and `__phx_processkeyopt:n`.)

We use almost the same way to load physics3 modules as $\LaTeX 2_{\epsilon}$ kernel does. We use a lot of kernel commands in $\LaTeX 2_{\epsilon}$.

```

\usephxmodu \usephxmodu [<key-val options>]{<module>} [<key-val options>]
\RequirePhxmodu \RequirePhxmodu [<key-val options>]{<module>} [<key-val options>]

```

`\usephxmodu` is a user command, and `\RequirePhxmodu` is a developer command.

```

27 \def\usephxmodu{\phx@FWoptions\@pkgextension}
28 \let\RequirePhxmodu\usephxmodu
29 \@onlypreamble\usephxmodu
30 \def\phx@FWoptions#1{\@ifnextchar [%]
31   {\phx@FWoptions#1}{\phx@FWoptions#1 []}}
32 \@onlypreamble\phx@FWoptions
33 \def\phx@FWoptions#1[#2]#3{\@ifnextchar [%]
34   {\phx@FWoptions#1[#2]#3}{\phx@FWoptions#1[#2]#3 []}}
35 \@onlypreamble\phx@FWoptions
36 \def\phx@FWoptions#1[#2]#3[#4]{%
37   \def\reserved@b##1,{%
38     \ifx\@nnil##1\relax\else
39       \ifx\@nnil##1\@nnil\else
40         \noexpand\@onefilewithoptions{phx-##1}{\unexpanded{#2}} [{#4}]%
41         \noexpand\@pkgextension
42       \fi
43       \expandafter\reserved@b
44     \fi}%
45   \edef\reserved@a{\zap@space#3~\@empty}%
46   \edef\reserved@a{\expandafter\reserved@b\reserved@a,\@nnil,}%
47   \reserved@a}
48 \@onlypreamble\phx@FWoptions

```

(End of definition for `\usephxmodu` and `\RequirePhxmodu`. These functions are documented on page 2.)

B.1.3 The (used to be) common module

The code below used to be the automatically-loaded common module, but now we load it together with physics3's code. This change may bring better performance in Windows system.

Check if unicode-math loaded and (re)define the vert symbols. The `\relax`'s at the ends of `\vert` and `\Vert`'s definitions must not be removed. They are for `\ifx` to compare. `unicode-math` sets these symbols `\fam1`, `\symoperators` is equal to 1 in $\text{\LaTeX 2}_{\epsilon}$ kernel. Moreover, we make `\mid` as a delimiter but it may not work.

```
49 \AtBeginDocument{\ifcsname symrm\endcsname
50   \protected\def|\{\Udelimiter 0 \symoperators "2016 }%
51   \protected\def\vert{\Udelimiter 0 \symoperators "007C\relax}%
52   \protected\def\Vert{\Udelimiter 0 \symoperators "2016\relax}%
53   \protected\def\mid{\Udelimiter 3 \symoperators "007C }%
54 \fi}
55 \protected\def\Vert{\delimiter"026B30D\relax}
56 \protected\def\mid{\delimiter"326A30C }
```

```
\delopen   \delopen <left delimiter>
\delclose  \delclose <right delimiter>
```

Actually in \TeX , `\left` and `\right` will enclose the subformula as “inner”, but `\delopen` and `\delclose` will make the subformula an empty open node and a non-empty close node.

```
57 \DeclareRobustCommand\delopen{\mathopen{}\mathclose\bgroup\left}
58 \DeclareRobustCommand\delclose{\aftergroup\egroup\right}
59 % Extension to 2e kernel's or amsmath's biggggg commands.
```

(End of definition for `\delopen` and `\delclose`. These functions are documented on page 3.)

`\bBigg@` is a command from `amsmath`. The code below should update with `amsmath` together.

```
60 \ifdefined\bBigg@
61   \DeclareRobustCommand\biggg{\bBigg@{3}}
62   \DeclareRobustCommand\Biggg{\bBigg@{3.5}}
63 \else
64   \DeclareRobustCommand\biggg[1]{\leavevmode@ifvmode
65     {\hbox{\$ \left#1 \vbox to 20.5\p@{\} \right. \n@space$}}}
66   \DeclareRobustCommand\Biggg[1]{\leavevmode@ifvmode
67     {\hbox{\$ \left#1 \vbox to 23.5\p@{\} \right. \n@space$}}}
68   \AtBeginDocument{\ifdefined\bBigg@
69     \DeclareRobustCommand\biggg{\bBigg@{3}}%
70     \DeclareRobustCommand\Biggg{\bBigg@{3.5}}%
```

```

71 \fi}
72 \fi
73 \DeclareRobustCommand\bigggl{\mathopen\biggg}
74 \DeclareRobustCommand\bigggm{\mathrel\biggg}
75 \DeclareRobustCommand\bigggr{\mathclose\biggg}
76 \DeclareRobustCommand\Bigggl{\mathopen\Biggg}
77 \DeclareRobustCommand\Bigggm{\mathrel\Biggg}
78 \DeclareRobustCommand\Bigggr{\mathclose\Biggg}

```

`_phx_mathvphantom:n`

`\phx@mathvphantom {⟨math mode material⟩}`

This command is just like `\vphantom` in $\text{\LaTeX} 2_{\epsilon}$ kernel but only works in math mode.

```

79 \cs_new:Npn \_phx_mathvphantom:n #1
80 {
81   \hbox_set:Nn \l__phx_tmpa_box {}
82   \tex_mathchoice:D
83   {
84     \hbox_set:Nn \l_tmpa_box { $\displaystyle#1$ }
85     \box_set_ht:Nn \l__phx_tmpa_box { \box_ht:N \l_tmpa_box }
86     \box_set_dp:Nn \l__phx_tmpa_box { \box_dp:N \l_tmpa_box }
87     \box_use_drop:N \l__phx_tmpa_box
88   }
89   {
90     \hbox_set:Nn \l_tmpa_box { $\textstyle#1$ }
91     \box_set_ht:Nn \l__phx_tmpa_box { \box_ht:N \l_tmpa_box }
92     \box_set_dp:Nn \l__phx_tmpa_box { \box_dp:N \l_tmpa_box }
93     \box_use_drop:N \l__phx_tmpa_box
94   }
95   {
96     \hbox_set:Nn \l_tmpa_box { $\scriptstyle#1$ }
97     \box_set_ht:Nn \l__phx_tmpa_box { \box_ht:N \l_tmpa_box }
98     \box_set_dp:Nn \l__phx_tmpa_box { \box_dp:N \l_tmpa_box }
99     \box_use_drop:N \l__phx_tmpa_box
100  }
101  {
102    \hbox_set:Nn \l_tmpa_box { $\scriptscriptstyle#1$ }
103    \box_set_ht:Nn \l__phx_tmpa_box { \box_ht:N \l_tmpa_box }
104    \box_set_dp:Nn \l__phx_tmpa_box { \box_dp:N \l_tmpa_box }
105    \box_use_drop:N \l__phx_tmpa_box
106  }
107 }
108 \cs_set_eq:NN \phx@mathvphantom \_phx_mathvphantom:n

```

(End of definition for `_phx_mathvphantom:n`.)

B.1.4 The (used to be) `explsetup` module

`\l__phx_tmpa_tl` Some common variables and functions for experimental L^AT_EX₃ syntax.

```
\l__phx_tmpb_tl 109 \tl_new:N \l__phx_tmpa_tl
                  110 \tl_new:N \l__phx_tmpb_tl
```

(End of definition for `\l__phx_tmpa_tl` and `\l__phx_tmpb_tl`.)

`\phx@del*` The syntax seems not important. These following lines seems only for `\ifcsname` to judge if the commands are defined.

```
\phx@del\big 111 \def\phx@del#1#2#3{\phx@abopen#1#3\phx@abclose#2}
\phx@del\Big 112 \expandafter\def\csname phx@del\string*\endcsname
\phx@del\bigg 113 #1#2#3{\mathopen#1#3\mathclose#2}
\phx@del\Bigg 114 \expandafter\def\csname phx@del\string\big\endcsname
\phx@del\biggg 115 #1#2#3{\bigl#1#3\bigr#2}
\phx@del\Biggg 116 \expandafter\def\csname phx@del\string\Big\endcsname
117 #1#2#3{\Bigl#1#3\Bigr#2}
118 \expandafter\def\csname phx@del\string\bigg\endcsname
119 #1#2#3{\biggl#1#3\biggr#2}
120 \expandafter\def\csname phx@del\string\Bigg\endcsname
121 #1#2#3{\Biggl#1#3\Biggr#2}
122 \expandafter\def\csname phx@del\string\biggg\endcsname
123 #1#2#3{\bigggl#1#3\bigggr#2}
124 \expandafter\def\csname phx@del\string\Biggg\endcsname
125 #1#2#3{\Biggggl#1#3\Biggggr#2}
```

(End of definition for `\phx@del*` and others.)

Parsing the clist in optional argument.

```
126 \def\@phx@ev@do@opt#1,{\ifx#1\relax\@empty\else
127 \edef\reserved@a{\zap@space#1-\@empty}%
128 \ifx\reserved@a\@empty\else
129 \ifcsname phx@del\expandafter \string \expandafter
130 \csname \expandafter \detokenize{\reserved@a}\endcsname \endcsname
131 \xdef\@phx@ev{\reserved@a}%
132 \else
133 \xdef\@phx@ev@basis{\reserved@a}%
134 \fi
135 \fi
136 \expandafter\@phx@ev@do@opt\fi}
137 \def\phx@ev@doopt#1{%
138 \edef\@phx@ev{\@empty}\edef\@phx@ev@basis{\@empty}\@phx@ev@do@opt#1,\relax,}
```

```
139 \file_input_stop:
```

```
140 </package>
```

Temporarily disable the namespace.

```
141 <@=@>
```

B.2 The ab module

```
<*gibberish>
```

This module is important but the code is hard to read. One of the motivations I manage physics3 with DocStrip is that, when I tried to write a new module based on ab after 5 months when I maintained physics3 the last time, I found that I could not understand the code I wrote at all! Therefore, it's significant to comment out the alien code in ab.

```
</gibberish>
```

```
142 <*ab>
```

```
143 \ProvidesFile {phx-ab.sty} [\phx@date\ 'ab' (autobraces) module of physics3]
```

If you don't know when to use `\phx@define@key`, `\phx@setkeys` and `\phx@processkeyopt` in a module, see ahead. In ab, the `|tightbraces|` option can control if the automatically-sized braces are tight or not. Do you remember `\delopen` and `\delclose`?

```
144 \phx@define@key{ab}{tightbraces}{true}{\def\@phx@abtight{#1}}
```

```
145 \phx@setkeys{ab}{tightbraces=true}
```

```
146 \phx@processkeyopt{ab}
```

```
\phx@abopen      \phx@abopen <left delimiter>
\phx@abclose     \phx@abclose <right delimiter>
```

They are defined either `{\delopen, \delclose}` or `{\left, \right}`. If a module requires ab, these two commands are likely to be used.

```
147 \ifx\@phx@abtight\phx@true
```

```
148 \let\phx@abopen\delopen
```

```
149 \let\phx@abclose\delclose
```

```
150 \else
```

```
151 \let\phx@abopen\left
```

```
152 \let\phx@abclose\right
```

```
153 \fi
```

(End of definition for `\phx@abopen` and `\phx@abclose`.)

B.2.1 The implementation of `\ab`

This is the alienest part of `ab`. It's better to draw something rather than write boring comments. First let's take a look at `\ab`'s syntax. After `\ab` should be a pair of delimiters; take `()` as an example. Between `\ab` and “`(`” can be a `biggg` command or `star`, or even nothing. `\ab` is defined as follows:

```
\ab ← begindef
      \phx@d@lx {mb} {ab}
enddef
```

where `ab` is the branch name of `\ab()`, and `mb` is the branch name of `\ab\big()` and `\ab*()`. Then let's see the syntax of `\phx@d@lx`.

```
\phx@d@lx {biggg or star branch name}
          {automatic branch name} {#3}
```

Here exists an `#3`. `#3` is one token immediately following `\ab`, which can be {a `biggg` command or a `star`} or a “`(`”, under our assumption.

`\phx@d@lx` is defined as follows:

```
\phx@d@lx ← begindef( #1: biggg or star branch name, <mb>; #2:
              automatic branch name, <ab>; #3, the token after
              \ab )
if#3 == biggg or #3 == star (⇔ csname {phx@del\string#3}
                             is defined)then
    let <next cs> = csname {phx@d@lx<mb>}
else
    let <next cs> = csname {phx@d@lx<ab>}
endif
    <next cs> #3
enddef
```

The condition should be true when `#3` is `\big` or `*`, and it should be false when `#3` is “`(`”. Accordingly, in math mode,

```
\ab \big ( → \phx@d@lxmb \big (
\ab      ( → \phx@d@lxab      (
```

Then we meet two new commands — `\phx@d@lxmb` and `\phx@d@lxab`. Syntax is as follows.

```

\phx@d@l@xmb <biggg or *> <left delimiter>
  <subformula> <right delimiter>
\phx@d@l@xab          <left delimiter>
  <subformula> <right delimiter>

```

Notice that ab and mb in the above commands are names of \ab's two branches — they are like namespaces. \phx@d@l@xmb and \phx@d@l@xab are defined by the following two lines:

```

\phx@d@l@genxm{mb} \phx@d@l@genxa{ab}

```

\phx@d@l@genxm and \phx@d@l@genxa are defined as follows:

```

\phx@d@l@genxm ← begindef(#1: biggg or star branch name, <mb>)
  \phx@d@l@x<mb> ← begindef(##1: biggg or star;
    ##2: left delimiter)
    \begingroup
    if##1 == starthen
      <temp> ← \relax
    else
      <temp> ← ##1
    endif
    csname {phx@<mb>@\string##2}
      <temp> ##2
    % requires an \endgroup af-
      ter the right delimiter
    enddef
  enddef
\phx@d@l@genxa ← begindef(#1: automatic branch name, <ab>)
  \phx@d@l@x<ab> ← begindef(##1: left delimiter)
    csname {phx@<ab>@\string##1}
      ##1
    enddef
  enddef

```


So we can get

```

\ab \big ( → \begingroup csname {phx@mb@} \big (
\ab * ( → \begingroup csname {phx@mb@} \relax (
\ab ( → csname {phx@ab@} (

```

The csnames above (`\phx@mb@` and `\phx@ab@`) are generated with `\phx@AB@gen`.

```

\phx@AB@gen {<branch name>} <left delimiter>
  {<arg spec>} {<definition>}

```

If `<branch name>` is `mb`, `{<arg spec>}` should be `mr()`, where `m` is for `bigg` or `star`; If `<branch name>` is `ab`, `{<arg spec>}` should be `r()`.

T_EXhackers note: The “(” in the example above must not be replaced by a subformula braced by a pair of `{}`.

```

\phx@AB@gen      \phx@AB@gen {<branch name>} <left delimiter>
                  {<arg spec>} {<definition>}

154 \def\phx@AB@gen#1#2{\expandafter
155   \DeclareDocumentCommand\csname phx@#1@\string#2\endcsname}
156   \phx@AB@gen{ab}({r()})%
157     {\phx@abopen(#1\phx@abclose)}
158   \phx@AB@gen{ab}({r[]})%
159     {\phx@abopen[#1\phx@abclose]}
160   \phx@AB@gen{ab}({r\{}})%
161     {\phx@abopen\{#1\phx@abclose\}}
162   \phx@AB@gen{ab}({r|})%
163     {\phx@abopen|#1\phx@abclose|}
164   \phx@AB@gen{ab}({r\|})%
165     {\phx@abopen\|#1\phx@abclose\|}
166   \phx@AB@gen{ab}(<r>)%
167     {\phx@abopen<#1\phx@abclose>}
168   \phx@AB@gen{ab}(\lbrace{r\lbrace\rbrace})%
169     {\phx@abopen\lbrace#1\phx@abclose\rbrace}
170   \phx@AB@gen{ab}(\vert{r\vert\vert})%
171     {\phx@abopen\vert#1\phx@abclose\vert}
172   \phx@AB@gen{ab}(\Vert{r\Vert\Vert})%
173     {\phx@abopen\Vert#1\phx@abclose\Vert}
174   \phx@AB@gen{ab}(\langle{r\langle\rangle})%
175     {\phx@abopen\langle#1\phx@abclose\rangle}

```

`\endgroup`'s in the end of the following definitions are corresponding to `\begingroup`'s in the definition of `\phx@d@l@genxm`.

```

176 \phx@AB@gen{mb}{\mr{}}%
177   {\mathopen#1(#2\mathclose#1)\endgroup}
178 \phx@AB@gen{mb}[\mr{}]%
179   {\mathopen#1[#2\mathclose#1]\endgroup}
180 \phx@AB@gen{mb}\{\mr{\}\}%
181   {\mathopen#1\lbrace#2\mathclose#1\rbrace\endgroup}
182 \phx@AB@gen{mb}|{\mr||}%
183   {\mathopen#1\vert#2\mathclose#1\vert\endgroup}
184 \phx@AB@gen{mb}\|{\mr\|\|\}%
185   {\mathopen#1\Vert#2\mathclose#1\Vert\endgroup}
186 \phx@AB@gen{mb}<{\mr<>}%
187   {\mathopen#1\langle#2\mathclose#1\rangle\endgroup}
188 \phx@AB@gen{mb}\lbrace{\mr\lbrace\rbrace}%
189   {\mathopen#1\lbrace#2\mathclose#1\rbrace\endgroup}
190 \phx@AB@gen{mb}\vert{\mr\vert\vert}%
191   {\mathopen#1\vert#2\mathclose#1\vert\endgroup}
192 \phx@AB@gen{mb}\Vert{\mr\Vert\Vert}%
193   {\mathopen#1\Vert#2\mathclose#1\Vert\endgroup}
194 \phx@AB@gen{mb}\langle{\mr\langle\rangle}%
195   {\mathopen#1\langle#2\mathclose#1\rangle\endgroup}

```

(End of definition for `\phx@AB@gen`.)

```

\phx@d@lx      \phx@d@lx {<biggg or star branch name>} {<automatic branch name>}
               {#3}

```

```

196 \def\phx@d@lx#1#2#3{%
197   \ifcsname phx@del\string#3\endcsname
198     \def\reserved@a{#1}% #3 is star or \biggg
199   \else
200     \def\reserved@a{#2}% #3 is delimiter
201   \fi
202   \csname phx@d@lx\reserved@a\endcsname#3}

```

(End of definition for `\phx@d@lx`.)

```

\phx@d@l@genxm \phx@d@l@genxm {<biggg or star branch name>}
\phx@d@l@genxa \phx@d@l@genxa {<automatic branch name>}

```

```

203 \def\phx@d@l@genxm#1{%
204   \expandafter\def\csname phx@d@lx#1\endcsname##1##2{%
205     \begingroup % \endgroup is at the end of #4 of \phx@AB@gen

```

```

206     \ifx##1*\let\phx@tempa=\relax\else\let\phx@tempa=##1\fi
207     \csname phx@#1@\string##2\endcsname\phx@tempa##2}}
208 \def\phx@d@l@genxa#1{%
209     \expandafter\def\csname phx@d@lx#1\endcsname##1{%
210     \csname phx@#1@\string##1\endcsname##1}}

```

(End of definition for `\phx@d@l@genxm` and `\phx@d@l@genxa`.)

```

\phx@d@l@xmb     \phx@d@l@xmb <bigg or *> <left delimiter>
\phx@d@l@xab     <subformula> <right delimiter>
                 \phx@d@l@xab     <left delimiter>
                 <subformula> <right delimiter>

```

```

211 \phx@d@l@genxm{mb}
212 \phx@d@l@genxa{ab}

```

(End of definition for `\phx@d@l@xmb` and `\phx@d@l@xab`.)

\ab The users' command `\ab`.

```

213 \DeclareRobustCommand\ab{\phx@d@lx{mb}{ab}}

```

(End of definition for `\ab`. This function is documented on page 3.)

B.2.2 \pab-like commands

This is so simple. No need to comment a lot.

```

\phx@d@l@geny     \phx@d@l@geny <command> <left delimiter> <right delimiter>
                 This command used to define commands like \pab.
214 \def\phx@d@l@geny#1#2#3{%

```

#1 : star; **#2** : bigg (csname); **#3** : subformula.

```

215     \DeclareDocumentCommand#1{som}{
216         \IfBooleanTF{##1}%
217         {#2##3#3}%
218         {\IfValueTF{##2}%
219             {\csname##21\endcsname#2##3\csname##2r\endcsname#3}%
220             {\phx@abopen#2##3\phx@abclose#3}%
221         }%
222     }%
223 }
224 \phx@d@l@geny\pab()

```

```

225 \phx@d@l@geny\bab[]
226 \phx@d@l@geny\Bab\lbrace\rbrace
227 \phx@d@l@geny\vab\vert\vert
228 \phx@d@l@geny\aab\langle\rangle
229 \phx@d@l@geny\Vab\Vert\Vert
230 \protect \endinput
231 </ab>

```

(End of definition for \phx@d@l@geny.)

B.3 The ab.braket module

```

232 <*ab.braket>
233 \ProvidesFile {phx-ab.braket.sty} [\phx@date\ 'ab.braket' module of physics3]

```

This module requires \phx@abopen and \phx@abclose from ab. This module may have conflict with braket.

```

234 \RequirePhxmodu{ab}
235 \ifdefined\phx@bra@@
236   \PackageWarning{physics3}{You cannot load 'ab.braket' and 'braket'
237     modules together.\MessageBreak Only 'ab.braket' module works now.}
238 \fi

```

\bra \bra < *subformula* |

(End of definition for \bra. This function is documented on page 5.)

```

239 \phx@AB@gen{br.m}<{mr<|}{\mathopen#1\langle#2\mathclose#1\vert\endgroup}
240 \phx@AB@gen{br.a}<{r<|}{\phx@abopen\langle#1\phx@abclose\vert}
241 \phx@d@l@genxm{br.m}
242 \phx@d@l@genxa{br.a}
243 \DeclareRobustCommand\bra{\phx@d@lx{br.m}{br.a}}

```

\ket \ket | *subformula* >

(End of definition for \ket. This function is documented on page 5.)

```

244 \phx@AB@gen{kt.m}|{mr|>}{\mathopen#1\vert#2\mathclose#1\rangle\endgroup}
245 \phx@AB@gen{kt.a}|{r|>}{\phx@abopen\vert#1\phx@abclose\rangle}
246 \phx@d@l@genxm{kt.m}
247 \phx@d@l@genxa{kt.a}
248 \DeclareRobustCommand\ket{\phx@d@lx{kt.m}{kt.a}}

```

```

\braket      \braket <⟨subformula1⟩ | ⟨subformula2⟩ [| ⟨subformula3⟩ ...] >
249 \begingroup
250 \catcode'\|=\active
251 \gdef\phx@mb@bk#1#2{\begingroup
252   \mathcode'\|="8000\def|{\egroup#1\vert\bgroup}%
253   \def\<{\mathrel{<}}\def\>{\mathrel{>}}%
254   \mathopen#1\langle\bgroup#2\egroup\mathclose#1\rangle\endgroup}
255 \gdef\phx@ab@bk#1{\begingroup
256   \mathcode'\|="8000\def|{\egroup\phx@ab@bkv\bgroup}%
257   \def\<{\mathrel{<}}\def\>{\mathrel{>}}%
258   \phx@abopen\langle\bgroup#1\egroup\phx@abclose\rangle\endgroup}
259 \endgroup
260 \def\phx@ab@bkv{\middle\vert}
261 \phx@AB@gen{bk.m}<{mr<>}{\phx@mb@bk#1#2}\endgroup}
262 \phx@AB@gen{bk.a}<{r<>}{\phx@ab@bk{#1}}
263 \phx@d@l@genxm{bk.m}
264 \phx@d@l@genxa{bk.a}
265 \DeclareRobustCommand \braket {\phx@d@l{x}{bk.m}{bk.a}}

```

(End of definition for \braket. This function is documented on page 6.)

```

\ketbra      \braket |⟨subformula1⟩> ⟨subformula2⟩ <⟨subformula3⟩|
266 \begingroup
267 \catcode'\<=\active
268 \catcode'\>=\active
269 \gdef\phx@mb@kb#1#2{\begingroup
270   \mathcode'\<="8000 \mathcode'\>="8000%
271   \def<{#1\langle}\def>{#1\rangle}%
272   \def\<{\phx@ab@l}\def\>{\phx@ab@r}%
273   \mathopen#1\vert#2\mathclose#1\vert\endgroup}
274 \endgroup
275 \gdef\phx@ab@kb#1#2<#3\phx@end{\begingroup
276   \def\<{\phx@ab@l}\def\>{\phx@ab@r}%
277   \phx@abopen\vert\mathopen{\phx@mathvphantom{#3}}#1\phx@abclose\rangle#2%
278   \phx@abopen\langle#3\mathclose{\phx@mathvphantom{#1}}\phx@abclose\vert
279 \endgroup}
280 \AtBeginDocument{\ifcsname symbf\endcsname
281   \def\phx@ab@l{\Umathchar 3 \symoperators "003C }%
282   \def\phx@ab@r{\Umathchar 3 \symoperators "003E }%
283   \fi}
284 \def\phx@ab@l{\mathchar"313C }
285 \def\phx@ab@r{\mathchar"313E }

```

```

286 \phx@AB@gen{kb.m}|{mr||}{\phx@omb@kb#1{#2}\endgroup}
287 \phx@AB@gen{kb.a}|{r||}{\phx@ab@kb#1\phx@end}
288 \phx@d@l@genxm{kb.m}
289 \phx@d@l@genxa{kb.a}
290 \DeclareRobustCommand \ketbra {\phx@d@lx{kb.m}{kb.a}}

```

(End of definition for `\ketbra`. This function is documented on page 7.)

\bknorm Define some high-level commands for inner product, outer product of basis and expectation value.

\kbproj

\expval

```

291 \DeclareDocumentCommand \bknorm { s o m }
292   {%
293     \IfBooleanTF{#1} {\mathopen\langle#3\vert #3\mathclose\rangle}
294     {%
295       \IfValueTF{#2}
296         {\expanded{\noexpand\phx@d@lx{bk.m}{bk.a}}%
297          \expandafter\noexpand\csname #2\endcsname}<#3|#3>}
298         {\phx@d@lx{bk.m}{bk.a}<#3|#3>}%
299     }%
300   }
301 \DeclareDocumentCommand \kbproj { s o m }
302   {%
303     \IfBooleanTF{#1}
304     {%
305       \mathopen\vert#3\mathclose\rangle
306       \mathopen\langle#3\mathclose\vert
307     }
308     {%
309       \IfValueTF{#2}
310         {\expanded{\noexpand\phx@d@lx{kb.m}{kb.a}}%
311          \expandafter\noexpand\csname #2\endcsname|#3><#3|}
312         {\phx@d@lx{kb.m}{kb.a}|#3><#3|}%
313     }%
314   }
315 \DeclareDocumentCommand \expval { s o m }
316   {%
317     \phx@ev@doopt{#2}%
318     \IfValueTF {#2}
319     {%
320       \ifx\@phx@ev\@empty
321         \ifx \@phx@ev@basis\@empty \else
322         \IfBooleanTF{#1}

```

```

323         {\phx@d@lx{bk.m}{bk.a}*<\@phx@ev@basis|#3|\@phx@ev@basis>}
324         {\phx@d@lx{bk.m}{bk.a}<\@phx@ev@basis|#3|\@phx@ev@basis>}%
325     \fi
326 \else
327     \ifx\@phx@ev@basis\@empty
328     \IfBooleanTF{#1} {\phx@d@lx{bk.m}{bk.a}*<#3>}
329     {%
330         \expanded{%
331             \noexpand \phx@d@lx{bk.m}{bk.a}\expandafter \noexpand
332             \csname \@phx@ev\endcsname}<#3>%
333         }%
334     \else
335     \IfBooleanTF{#1}
336     {%
337         \phx@d@lx{bk.m}{bk.a}*<\@phx@ev@basis|#3|\@phx@ev@basis>%
338     }
339     {%
340         \expanded{%
341             \noexpand \phx@d@lx{bk.m}{bk.a}\expandafter
342             \noexpand \csname \@phx@ev\endcsname}%
343         <\@phx@ev@basis|#3|\@phx@ev@basis>%
344     }%
345     \fi
346 \fi
347 }
348 {%
349     \IfBooleanTF{#1}
350     {\phx@d@lx{bk.m}{bk.a}*<#3>} {\phx@d@lx{bk.m}{bk.a}<#3>}%
351 }%
352 }

```

(End of definition for `\bknorm`, `\kbproj`, and `\expval`. These functions are documented on page 7.)

```

353 \protect \endinput

```

```

354 </ab.braket>

```

B.4 The braket module

```

355 <*braket>
356 \ProvidesFile {phx-braket.sty} [\phx@date\ 'braket' module of physics3]

```

This module requires `\phx@abopen` and `\phx@abclose` from `ab`. This module may have conflict with `ab.braket`.

```

357 \RequirePhxmodu{ab}
358 \ifdefined\phx@abb@bkv
359   \PackageWarning{physics3}{You cannot load ‘ab.braket’ and ‘braket’
360     modules together.\MessageBreak Only ‘braket’ module works now.}
361 \fi

```

\bra $\bra * [⟨biggg⟩] \{⟨subformula⟩\}$

```

362 \DeclareDocumentCommand\bra{ s o m }{%
363   \IfBooleanTF{#1}
364     {\mathopen\langle#3\mathclose\vert}
365     {\IfValueTF{#2}
366       {\csname#2l\endcsname\langle#3\csname#2r\endcsname\vert}
367       {\phx@abopen\langle#3\phx@abclose\vert}}%
368   }%
369 }

```

(End of definition for \bra. This function is documented on page 5.)

\ket $\ket * [⟨biggg⟩] \{⟨subformula⟩\}$

```

370 \DeclareDocumentCommand\ket{ s o m }{%
371   \IfBooleanTF{#1}
372     {\mathopen\vert#3\mathclose\rangle}
373     {\IfValueTF{#2}
374       {\csname#2l\endcsname\vert#3\csname#2r\endcsname\rangle}
375       {\phx@abopen\vert#3\phx@abclose\rangle}}%
376   }%
377 }

```

(End of definition for \ket. This function is documented on page 5.)

\braket $\ket * [⟨biggg⟩, n \in \{1, 2, 3\}] \{⟨subformula 1⟩\} \dots \{⟨subformula n⟩\}$

```

378 \DeclareDocumentCommand \braket { s O{} }
379   {%
380     \IfBooleanTF{#1}
381       {%
382         \gdef\@phx@bk@argnum{ii}%
383         \phx@bk@doopt{#2}%
384         \gdef\@phx@bk@l{\mathopen}%
385         \gdef\@phx@bk@m{\mathord}%
386         \gdef\@phx@bk@r{\mathclose}%
387       }
388     {%

```



```

389     \gdef\@phx@bk@argnum{ii}%
390     \gdef\@phx@bk@l{phx@abopen}%
391     \gdef\@phx@bk@m{middle}%
392     \gdef\@phx@bk@r{phx@abclose}%
393     \phx@bk@doopt{#2}%
394 }%
395 \csname phx@bk@in@\@phx@bk@argnum\endcsname
396 }

```

(End of definition for `\braket`. This function is documented on page 6.)

```

\phx@bk@in@i     \phx@bk@in@<n.roman> {\langle subformula 1\rangle} ... {\langle subformula n\rangle}
\phx@bk@in@ii    <n.roman> is n in roman lowercase, where n ∈ {1, 2, 3}.
\phx@bk@in@iii   \def\phx@bk@in@i#1{%
397   \csname\@phx@bk@l\endcsname\langle#1\rangle%
398   \csname\@phx@bk@r\endcsname\rangle}
\phx@bk@in@iii   \def\phx@bk@in@ii#1#2{%
400   \csname\@phx@bk@l\endcsname\langle#1\rangle%
401   \csname\@phx@bk@m\endcsname\vert{#2}%
402   \csname\@phx@bk@r\endcsname\rangle}
\phx@bk@in@iii   \def\phx@bk@in@iii#1#2#3{%
404   \csname\@phx@bk@l\endcsname\langle#1\rangle%
405   \csname\@phx@bk@m\endcsname\vert{#2}%
406   \csname\@phx@bk@m\endcsname\vert{#3}%
407   \csname\@phx@bk@r\endcsname\rangle}
408 }

```

(End of definition for `\phx@bk@in@i`, `\phx@bk@in@ii`, and `\phx@bk@in@iii`.)

```

\phx@bk@doopt    \phx@bk@doopt {\langle clist\rangle}
\phx@bk@do@pt    Parse the optional argument of \braket. This will add 3 entries to hash.
409 \def\@phx@bk@do@pt#1,{\ifx#1\relax\@empty\else
410 \edef\reserved@a{\zap@space#1 \@empty}%
411 \ifx\reserved@a\@empty\else
412   \ifcsname phx@del\expandafter\string\csname\reserved@a\endcsname\endcsname
413     \xdef\@phx@bk@l{\reserved@a l}%
414     \xdef\@phx@bk@m{\reserved@a}% but not m (m stands for \mathrel)
415     \xdef\@phx@bk@r{\reserved@a r}%
416   \else
417     \ifnum\reserved@a>3%
418     \PackageError{physics3}{\string\braket\space can only take 3
419     mandatory arguments at most}{Check if you had written a number
420     more than 3 in the [optional] argument.}%

```

```

421     \fi
422     \xdef\@phx@bk@argnum{\romannumeral\reserved@a}%
423     \fi
424     \fi
425     \expandafter\@phx@bk@do@pt\fi}
426 \def\phx@bk@doopt#1{\@phx@bk@do@pt#1,\relax,}

```

(End of definition for \phx@bk@doopt and \phx@bk@do@pt.)

```

\ketbra      \ketbra * [(biggg)] {\subformula 1}
              [(\between 1 and 2)] {\subformula 2}

427 \DeclareDocumentCommand \ketbra { s o m 0{} m }
428   {%
429   \IfBooleanTF{#1}
430     {%
431       \mathopen\vert#3\mathclose\rangle#4%
432       \mathopen\langle#5\mathclose\vert
433     }
434     {%
435       \IfValueTF{#2}
436         {\csname#2l\endcsname\vert#3\csname#2r\endcsname\rangle#4%
437          \csname#2l\endcsname\langle#5\csname#2r\endcsname\vert}
438         {\begingroup
439          \phx@abopen\vert
440          \mathopen{\phx@mathvphantom{#5}}#3\phx@abclose\rangle#4%
441          \phx@abopen\langle#5
442          \mathclose{\phx@mathvphantom{#3}}\phx@abclose\vert
443          \endgroup}%
444         }%
445   }

```

(End of definition for \ketbra. This function is documented on page 7.)

\bknorm Define some high-level commands for inner product, outer product of basis and expectation value.

```

\kbproj
\expval
446 \DeclareDocumentCommand \bknorm { s o m }
447   {%
448   \IfBooleanTF{#1}
449     {%
450       \gdef\@phx@bk@l{\mathopen}%
451       \gdef\@phx@bk@m{\mathord}%
452       \gdef\@phx@bk@r{\mathclose}%

```

```

453 }
454 {%
455   \gdef\@phx@bk@l{phx@abopen}%
456   \gdef\@phx@bk@m{middle}%
457   \gdef\@phx@bk@r{phx@abclose}%
458   \IfValueT{#2}
459     {%
460       \ifcsname phx@del\expandafter\string\csname#2\endcsname\endcsname
461         \xdef\@phx@bk@l{#2l}%
462         \xdef\@phx@bk@m{#2}%
463         \xdef\@phx@bk@r{#2r}%
464       \fi
465     }%
466   }%
467   \phx@bk@in@ii {#3} {#3}%
468 }
469 \DeclareDocumentCommand \kbproj { s o m }
470 {%
471   \IfBooleanTF{#1}
472     {%
473       \mathopen\vert#3\mathclose\rangle
474       \mathopen\langle#3\mathclose\vert
475     }
476     {%
477       \IfValueTF{#2}
478         {\csname#2l\endcsname\vert#3\csname#2r\endcsname\rangle
479          \csname#2l\endcsname\langle#3\csname#2r\endcsname\vert}
480         {\begingroup
481           \phx@abopen\vert
482           \mathopen{\phx@mathvphantom{#3}}#3\phx@abclose\rangle
483           \phx@abopen\langle#3
484           \mathclose{\phx@mathvphantom{#3}}\phx@abclose\vert
485           \endgroup}%
486         }%
487     }
488 \DeclareDocumentCommand \expval { s O{ } m }
489 {
490   \phx@ev@doopt{#2}%
491   \IfBooleanTF{#1}
492     {%
493       \gdef\@phx@bk@l{mathopen}%
494       \gdef\@phx@bk@m{mathord}%

```

```

495     \gdef\@phx@bk@r{mathclose}%
496   }
497   {%
498     \gdef\@phx@bk@l{phx@abopen}%
499     \gdef\@phx@bk@m{middle}%
500     \gdef\@phx@bk@r{phx@abclose}%
501     \ifx \@phx@ev@empty \else
502       \xdef\@phx@bk@l{\@phx@ev l}%
503       \xdef\@phx@bk@m{\@phx@ev}%
504       \xdef\@phx@bk@r{\@phx@ev r}%
505     \fi
506   }%
507   \ifx \@phx@ev@basis@empty
508     \phx@bk@in@i {#3}%
509   \else
510     \phx@bk@in@iii {\@phx@ev@basis} {#3} {\@phx@ev@basis}%
511   \fi
512 }

```

(End of definition for `\bknorm`, `\kbproj`, and `\expval`. These functions are documented on page 7.)

```
513 \protect \endinput
```

```
514 </braket>
```

Restore the namespace.

```
515 <@=@phx>
```

B.5 The doubleprod module

```
516 <*doubleprod>
```

```
517 \ProvidesExplFile {phx-doubleprod.sty} {\phx@date} {\phx@version}
```

```
518 {'doubleprod' (vertically stacked binary operators) module of physics3}
```

Boolean options.

```
519 \__phx_define_key:nnnn { doubleprod } { crosssymbol } { } { \def\@phx@dbl@c {#1} }
```

```
520 \__phx_define_key:nnnn { doubleprod } { dotsymbol } { } { \def\@phx@dbl@d {#1} }
```

```
521 \__phx_define_key:nnnn { doubleprod } { crossscale } { } { \def\@phx@dbl@sc{#1} }
```

```
522 \__phx_define_key:nnnn { doubleprod } { dotscale } { } { \def\@phx@dbl@sd{#1} }
```

```
523 \__phx_define_key:nnnn { doubleprod } { crossopenup } { } { \def\@phx@dbl@oc{#1} }
```

```
524 \__phx_define_key:nnnn { doubleprod } { dotopenup } { } { \def\@phx@dbl@od{#1} }
```

```
525 \__phx_setkeys:nn { doubleprod }
```

```
526 {
```

```
527   crosssymbol = \times, crossscale = 0.8, crossopenup = .02,
```

```

528     dotsymbol = \ldotp, dotscale = 1, dotopenup = .2
529   }
530 \__phx_processkeyopt:n { doubleprod }
531 \def\phx@dbl@gen#1#2#3#4{%
532   \DeclareRobustCommand#1{\mathbin{\vcenter{\baselineskip\z@skip%
533     \lineskip#4\phx@dblcurrf@size%
534     \hbox_set:Nn \l_tmpa_box {\fontsize{#2\phx@dblcurrf@size}\z@#3$}
535     \box_use:N \l_tmpa_box \box_use_drop:N \l_tmpa_box}}}}
536 \def\phx@dblcurrf@size{\dimexpr\fontdimen\relax}
537 \phx@dbl@gen\doublecross\@phx@dbl@sc\@phx@dbl@c\@phx@dbl@oc
538 \phx@dbl@gen\doubledot\@phx@dbl@sd\@phx@dbl@d\@phx@dbl@od
539 \file_input_stop:
540 \</doubleprod>

```

B.6 The diagmat module

```

541 \<*diagmat>
542 \ProvidesExplFile {phx-diagmat.sty} {\phx@date} {\phx@version}
543   {'diagmat' module of physics3}

```

This module requires mathtools and xpatch.

```

544 \RequirePackage { mathtools, etoolbox }

```

Do expansion of column specification argument (thanks @egreg on [T_EX StackExchange](#)).

```

545 \patchcmd \MT_matrix_begin:N { \exp_args:Ne \array } { \exp_args:Ne \array } {}
546   { \patchcmd \MT_matrix_begin:N { \array } { \exp_args:Ne \array } {} {} }
547 \__phx_define_key:nnnn { diagmat } { empty } {   }
548   { \tl_gset:Nn \l__phx_mat_empty_tl { #1 } }
549 \__phx_define_key:nnnn { diagmat } { align } { r }
550   { \tl_gset:Nn \l__phx_mat_align_tl { #1 } }

```

This module requires some new variables.

```

\l__phx_mat_diag_clist
\l__phx_mat_dim_int
\l__phx_mat_line_tl
\l__phx_diagmat_tl
\l__phx_mat_empty_tl
\l__phx_mat_align_tl
\l__phx_xmat_align_c_tl
551 \clist_new:N \l__phx_mat_diag_clist
552 \int_new:N \l__phx_mat_dim_int
553 \tl_new:N \l__phx_mat_line_tl
554 \tl_new:N \l__phx_diagmat_tl
555 \tl_new:N \l__phx_mat_empty_tl
556 \tl_new:N \l__phx_mat_align_tl
557 \tl_const:Nn \l__phx_mat_align_c_tl { c }
558 \__phx_processkeyopt:n { diagmat }
559 \keys_define:nn { phy / diagmat }
560   {
561     empty .tl_set:N = \l__phx_mat_empty_tl,
562     align .tl_set:N = \l__phx_mat_align_tl,

```

563 }

(End of definition for `\l__phx_mat_diag_clist` and others.)

```
\diagmat      \langle delimiter type \rangle diagmat [⟨key-val list⟩] {⟨diagonal⟩}
\pdiagmat
\bdiagmat     564 \clist_map_inline:nn { {}, p, b, B, v, V }
\Bdiagmat     565 {
\vdigmat      566   \exp_args:Nc \DeclareDocumentCommand { #1diagmat } { s O{} m }
\Vdiigmat     567   {
                    568     \IfBooleanTF { ##1 }
                    569       { \__phx_diagmat_type:nnn { #1small } { ##2 } { ##3 } }
                    570       { \__phx_diagmat_type:nnn { #1      } { ##2 } { ##3 } }
                    571     }
572 }
```

(End of definition for `\diagmat` and others. These functions are documented on page 8.)

```
\__phx_diagmat_type:nnn  \__phx_diagmat_type:nnn {⟨delimiter type⟩}
                        {⟨key-val list⟩} {⟨diagonal⟩}
573 \cs_new:Npn \__phx_diagmat_type:nnn #1#2#3
574   {
575     \group_begin:
576     \clist_set:Nn \l__phx_mat_diag_clist {#3}
577     \int_set:Nn \l__phx_mat_dim_int { \clist_count:N \l__phx_mat_diag_clist }
578     \int_compare:nNnT { \l__phx_mat_dim_int } > { \value { MaxMatrixCols } }
579       { \setcounter { MaxMatrixCols } { \l__phx_mat_dim_int } }
580     \keys_set:nn { phy / diagmat } {#2}
581     \tl_gclear:N \l__phx_diagmat_tl
582     \int_step_inline:nnn { 1 } { \l__phx_mat_dim_int }
583       {
584         \int_step_inline:nnn { 1 } { \l__phx_mat_dim_int }
585           {
586             \int_compare:nNnTF { ##1 } = { ####1 }
587               {
588                 \clist_gpop:NN \l__phx_mat_diag_clist \l__phx_tmpa_tl
589                 \tl_if_empty:NTF \l__phx_tmpa_tl
590                   { \tl_gput_right:Nn \l__phx_mat_line_tl { \l__phx_mat_empty_tl } }
591
592                 { \tl_gput_right:No \l__phx_mat_line_tl { \l__phx_tmpa_tl } }

```

Maybe it's better to use `\expandafter\scantokens\expandafter{\l__phx_tmpa_tl}` in the next line.

```

593         { \tl_gput_right:Nn \l__phx_mat_line_tl { \l__phx_mat_empty_tl } }
594     \int_compare:nNnTF { ###1 } = { \l__phx_mat_dim_int }
595     {
596         \tl_gput_right:Nn \l__phx_mat_line_tl { \ }
597     }
598     {
599         \tl_gput_right:Nn \l__phx_mat_line_tl { & }
600     }
601 }
602 \tl_gput_right:No \l__phx_diagmat_tl { \l__phx_mat_line_tl }
603 \tl_gclear:N \l__phx_mat_line_tl
604 }
605 \bool_lazy_or:nnTF
606 { \tl_if_empty_p:N \l__phx_mat_align_tl }
607 { \tl_if_eq_p:NN \l__phx_mat_align_tl \l__phx_mat_align_c_tl }
608 {
609     \begin { #1 matrix }
610         \tl_use:N \l__phx_diagmat_tl
611     \end { #1 matrix }
612 }
613 {
614     \begin { #1 matrix* } [ \l__phx_mat_align_tl ]
615         \tl_use:N \l__phx_diagmat_tl
616     \end { #1 matrix* }
617 }
618 \group_end:
619 }

```

(End of definition for `__phx_diagmat_type:nnn`.)

```
620 \file_input_stop:
```

```
621 </diagmat>
```

B.7 The `xmat` module

```

622 <*xmat>
623 \ProvidesExplFile {phx-xmat.sty} {\phx@date} {\phx@version}
624   {'xmat' module of physics3}

```

This module requires `mathtools` and `xpatch`.

```
625 \RequirePackage { mathtools, etoolbox }
```

Do expansion of column specification argument (thanks @egreg on [T_EX StackExchange](#)).

```
626 \patchcmd \MT_matrix_begin:N { \exp_args:Ne \array } { \exp_args:Ne \array } {}
```

```

627 { \patchcmd \MT_matrix_begin:N { \array } { \exp_args:Ne \array } {} {} }
628 \__phx_define_key:nnnn { xmat } { showtop } { }
629 { \int_gset:Nn \l__phx_xmat_showtop_int { #1 } }
630 \__phx_define_key:nnnn { xmat } { showleft } { }
631 { \int_gset:Nn \l__phx_xmat_showleft_int { #1 } }
632 \__phx_define_key:nnnn { diagmat } { align } { r }
633 { \tl_gset:Nn \l__phx_xmat_align_tl { #1 } }

```

`\l__phx_xmat_extra_vdots_bool` This module requires some new variables.

```

\l__phx_xmat_extra_cdots_bool 634 \bool_new:N \l__phx_xmat_extra_vdots_bool

```

```

\l__phx_xmat_showtop_int 635 \bool_new:N \l__phx_xmat_extra_cdots_bool

```

```

\l__phx_xmat_showleft_int 636 \int_new:N \l__phx_xmat_showtop_int

```

```

\l__phx_xmat_tl 637 \int_new:N \l__phx_xmat_showleft_int

```

```

\l__phx_xmat_align_tl 638 \tl_new:N \l__phx_xmat_tl

```

```

\l__phx_xmat_align_c_tl 639 \tl_new:N \l__phx_xmat_align_tl

```

```

640 \tl_const:Nn \l__phx_xmat_align_c_tl { c }

```

```

641 \int_gset:Nn \l__phx_xmat_showtop_int { \value { MaxMatrixCols } - 2 }

```

```

642 \int_gset:Nn \l__phx_xmat_showleft_int { \value { MaxMatrixCols } - 2 }

```

(End of definition for `\l__phx_xmat_extra_vdots_bool` and others.)

```

643 \cs_new:Npn \__phx_xmat_entry_format:nnn #1#2#3

```

```

644 {
645   #1 \c_math_subscript_token { #2 #3 }
646 }

```

```

647 \__phx_processkeyopt:n { xmat }

```

```

648 \clist_map_inline:nn { {}, p, b, B, v, V }

```

```

649 {
650   \exp_args:Nc \DeclareDocumentCommand { #1xmat } { s O{} m m m }

```

```

651   {
652     \IfBooleanTF { ##1 }
653     { \__phx_xmat_type:nnnnn { #1small } { ##2 } { ##3 } { ##4 } { ##5 } }
654     { \__phx_xmat_type:nnnnn { #1      } { ##2 } { ##3 } { ##4 } { ##5 } }
655   }

```

```

656 }

```

```

657 \keys_define:nn { phx / xmat }

```

```

658 {
659   format .cs_set:Np = \__phx_xmat_entry_format:nnn #1#2#3 ,
660   showtop .int_set:N = \l__phx_xmat_showtop_int ,
661   showleft.int_set:N = \l__phx_xmat_showleft_int ,
662   align .tl_set:N = \l__phx_xmat_align_tl ,
663 }

```



```

\__phx_if_digits_only_p:n      \__phx_if_digits_only:nTF {<token list>} {<true code>} {<false code>}
\__phx_if_digits_only:nTF     Use LATEX3 regular expression to tell if <token list> (the numbers of rows or
                               columns) contain digits only.

```

(End of definition for __phx_if_digits_only:nTF.)

```

664 \prg_new_conditional:Npnn \__phx_if_digits_only:n #1 { TF }
665   {
666     \regex_match:nnTF { \A [[:digit:]]* \Z } { #1 }
667     { \prg_return_true: } { \prg_return_false: }
668   }

```

```

\__phx_xmat_type:nnnnn      \__phx_xmat_type:nnnnn {<delimiter type>} {<key-val list>}
                               {<common entry>} {<row number>} {<column number>}

```

(End of definition for __phx_xmat_type:nnnnn.)

```

669 \cs_new:Npn \__phx_xmat_type:nnnnn #1#2#3#4#5
670   {
671     \group_begin:
672     \tl_gclear:N \l__phx_xmat_tl
673     \keys_set:nn { phx / xmat } { #2 } %
674     \__phx_if_digits_only:nTF { #4 }
675     {
676       \int_compare:nNnTF { #4 } < { \l__phx_xmat_showtop_int + 1 }
677       {
678         \int_set:Nn \l__phx_xmat_showtop_int { #4 }
679         \bool_set_false:N \l__phx_xmat_extra_vdots_bool
680       }
681       {
682         \bool_set_true:N \l__phx_xmat_extra_vdots_bool
683       }
684     }
685     {
686       \bool_set_true:N \l__phx_xmat_extra_vdots_bool
687     }
688     \__phx_if_digits_only:nTF { #5 }
689     {
690       \int_compare:nNnTF { #5 } < { \l__phx_xmat_showleft_int + 1 }
691       {
692         \int_set:Nn \l__phx_xmat_showleft_int { #5 }
693         \bool_set_false:N \l__phx_xmat_extra_cdots_bool
694       }
695     }

```

```

696         \bool_set_true:N \l__phx_xmat_extra_cdots_bool
697     }
698 }
699 {
700     \bool_set_true:N \l__phx_xmat_extra_cdots_bool
701 }
702 \int_step_inline:nn { \l__phx_xmat_showtop_int }
703 {
704     \tl_put_right:Nn \l__phx_xmat_tl
705     { \__phx_xmat_entry_format:nnn { #3 } { ##1 } { 1 } }
706     \int_step_inline:nnn { 2 } { \l__phx_xmat_showleft_int }
707     {
708         \tl_put_right:Nn \l__phx_xmat_tl
709         { & \__phx_xmat_entry_format:nnn { #3 } { ##1 } { ####1 } }
710     }
711     \bool_if:NT \l__phx_xmat_extra_cdots_bool
712     {
713         \tl_put_right:Nn \l__phx_xmat_tl
714         { & \cdots & \__phx_xmat_entry_format:nnn { #3 } { ##1 } { #5 } }
715     }
716     \tl_put_right:Nn \l__phx_xmat_tl { \ }
717 }
718 \bool_if:NT \l__phx_xmat_extra_vdots_bool
719 {
720     \tl_put_right:Nn \l__phx_xmat_tl { \vdots }
721     \prg_replicate:nn { \l__phx_xmat_showleft_int - 1 }
722     {
723         \tl_put_right:Nn \l__phx_xmat_tl { & \vdots }
724     }
725     % Add \ddots if vdots_bool and cdots_bool be true simultaneously.
726     \bool_if:NT \l__phx_xmat_extra_cdots_bool
727     {
728         \tl_put_right:Nn \l__phx_xmat_tl { & \ddots & \vdots }
729     } % else relax
730     \tl_put_right:Nn \l__phx_xmat_tl { \ }
731     % The last row.
732     \tl_put_right:Nn \l__phx_xmat_tl
733     { \__phx_xmat_entry_format:nnn { #3 } { #4 } { 1 } }
734     \int_step_inline:nnn { 2 } { \l__phx_xmat_showleft_int }
735     {
736         \tl_put_right:Nn \l__phx_xmat_tl
737         { & \__phx_xmat_entry_format:nnn { #3 } { #4 } { ##1 } }

```

```

738     }
739     \bool_if:NT \l__phx_xmat_extra_cdots_bool
740     {
741         \tl_put_right:Nn \l__phx_xmat_tl
742         { & \cdots & \__phx_xmat_entry_format:nnn { #3 } { #4 } { #5 } }
743     }
744     } % else relax
745     \bool_lazy_or:nnTF
746     { \tl_if_empty_p:N \l__phx_xmat_align_tl }
747     { \tl_if_eq_p:NN \l__phx_xmat_align_tl \l__phx_xmat_align_c_tl }
748     {
749         \begin { #1 matrix }
750         \tl_use:N \l__phx_xmat_tl
751         \end { #1 matrix }
752     }
753     {
754         \begin { #1 matrix* } [ \l__phx_xmat_align_tl ]
755         \tl_use:N \l__phx_xmat_tl
756         \end { #1 matrix* }
757     }
758     \group_end:
759 }
760 \file_input_stop:
761 </xmat>

```

B.8 The operator module

```

762 <*operator>
763 \ProvidesExplFile {phx-operator.sty} {\phx@date} {\phx@version}
764   {'operator' module of physics3}
765   \__phx_define_key:nnnn { operator } { ReIm } { true }
766   { \def\phx@reserveda {#1} }
767   \__phx_define_key:nnnn { operator } { identity } { } { \def\phx@oplega@id {#1} }
768   \__phx_define_key:nnnn { operator } { PV } { } { \def\@phx@oplega@PV{#1} }
769   \__phx_define_key:nnnn { operator } { pv } { } { \def\@phx@oplega@pv{#1} }
770   \__phx_setkeys:nn { operator }
771   { PV = \mathcal{P}, pv = {p.v.}, ReIm = true, identity = bold }
772   \__phx_processkeyopt:n { operator }

\asin
\acos 773 \clist_map_inline:nn
\atan 774 { asin, acos, atan, acsc, asec, acot, erf, erfc, Tr, tr, rank, Res, res }
\acsc
\asec
\acot
\erf
\erfc
\Tr
\tr
\rank

```

```

775 {
776   \exp_args:Nc \DeclareRobustCommand {#1}
777     { \mathop { \operator@font #1 } \nolimits }
778 }
779 \DeclareRobustCommand\PV  {\mathord{\@phx@oplega@PV}}
780 \DeclareRobustCommand\pv  {\mathop{\operator@font\@phx@oplega@pv{}}\nolimits}
781 \DeclareRobustCommand\upe {\mathrm e}
782 \DeclareRobustCommand\iu  {\mathrm i\mkern1mu}
783 \clist_map_inline:nn { i, I }
784 {
785   \exp_args:Nc \DeclareDocumentCommand {#1identity} { o }
786   {
787     \group_begin:
788     \mode_if_math:TF
789     {
790       \mbox
791       {
792         \fontencoding{U} \fontseries{m} \fontshape{n}
793         \IfValueTF { ##1 }
794           { \fontfamily { ##1 } } { \fontfamily { \phx@oplega@id } }
795         \selectfont
796         \tl_if_eq:nnT {#1} { i } { 1 }
797         \tl_if_eq:nnT {#1} { I } { I }
798       }
799     }
800     {
801       \exp_args:Nnx \msg_error:nn { physics3 }
802       {
803         The ~ \expandafter\string\csname #1identity\endcsname\space
804         command ~ requires ~ math ~ mode.
805       }
806     }
807   \group_end:
808 }
809 }

```

(End of definition for \asin and others.)

\Re Restore \Re and \Im in \Resymbol and \Imsymbol. The \AtBeginDocument hook is used for the compatibility of unicode-math.

```

810 \ifx \phx@reserveda \phx@true
811   \AtBeginDocument

```

```

812 {
813   \let\Resymbol\Re
814   \let\Imsymbol\Im
815   \DeclareRobustCommand\Re{\mathop{\operator@font Re}\nolimits}%
816   \DeclareRobustCommand\Im{\mathop{\operator@font Im}\nolimits}%
817 }
818 \fi

```

(End of definition for `\Re` and `\Im`.)

Requires fixdif version 2.x.

```

819 \RequirePackage{fixdif}[2023/01/31]
820 \letdif\phx@nl@nabla{nabla}

```

```

\grad   ∇-related operators.
\div    821 \RequirePhxmodu{ab}
\curl   822 \AtBeginDocument
\laplacian 823 {
824   \ifcsname div\endcsname\let\divsymbol\div\fi
825   \DeclareRobustCommand \grad    {\phx@nl@nabla\ab}
826   \DeclareRobustCommand \div     {\phx@nl@nabla\cdot\ab}
827   \DeclareRobustCommand \curl    {\phx@nl@nabla\times\ab}
828   \DeclareRobustCommand \laplacian{\phx@nl@nabla^2\ab}
829 }

```

(End of definition for `\grad` and others.)

```

830 \file_input_stop:
831 \</operator>

```

B.9 The `ab.legacy` module

```

832 \<*ab.legacy>
833 \ProvidesFile {phx-ab.legacy.sty} [\phx@date\ 'ab.legacy' module of physics3]

```

Requires `ab`'s `tight` option.

```

834 \RequirePhxmodu{ab}
835 \phx@define@key{ab.legacy}{order}{\mathcal 0}{\def\phx@ab@ordersym{#1}}
836 \phx@setkeys{ab.legacy}{order}
837 \phx@processkeyopt{ab.legacy}
838 \phx@d@l@geny\abs\vert\vert
839 \phx@d@l@geny\norm\Vert\Vert
840 \DeclareDocumentCommand\order{som}{%
841   \phx@ab@ordersym

```

```

842 \IfBooleanTF{#1} {(#3)}
843   {\IfValueTF{#2}
844     {\csname#2l\endcsname(#3\csname#2r\endcsname)}
845     {\phx@abopen(#3\phx@abclose)}}%
846   }%
847 }
848 \phx@d@l@geny\eval.\vert
849 \phx@d@l@geny\peval(\vert
850 \phx@d@l@geny\beval[\vert
851 \protect \endinput
852 </ab.legacy>

```

B.10 The `qtext.legacy` module

This module is written for the compatibility with the bad commands provided by physics only. The commands in this module should NEVER be used!

```

853 <*qtext.legacy>
854 \ProvidesExplFile {phx-qtext.legacy.sty} {\phx@date} {\phx@version}
855 {'qtext.legacy' module of physics3.sty}
856 \RequirePackage{amstext}
857 \def\phx@qtext@#1#2{#1\text{#2}\quad}
858 \DeclareRobustCommand\qqtext{\@ifstar{\phx@qtext@{}}{\phx@qtext@\quad}}
859 \DeclareRobustCommand\qq    {\qqtext}
860 \DeclareRobustCommand\qcomma{,\quad}
861 \DeclareRobustCommand\qc    {\qcomma}
862 \DeclareRobustCommand\qcc
863   {\@ifstar{\phx@qtext@{c.c}}{\phx@qtext@\quad{c.c}}}
864 \clist_map_inline:nn
865   {
866     if,      then,   else,   otherwise,  unless,
867     give,    using,  unless, assume,  since,
868     let,     for,    all,    even,      odd,
869     integer, and,   or,     as,       in
870   }
871   {
872     \exp_args:Nc \DeclareRobustCommand { q#1 }
873     { \@ifstar{ \phx@qtext@ { } {#1} } { \phx@qtext@ \quad {#1} } }
874   }
875 \file_input_stop:
876 </qtext.legacy>

```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Symbols	
\langle	253, 257, 267, 270, 272, 276
\langle	6
\rangle	253, 257, 268, 270, 272, 276
\rangle	6
\backslash	596, 716, 730
$\{$	160, 161, 180
$\}$	160, 161, 180
$\langle cmd \rangle$	11
$\langle delimiter type \rangle xmat$	9
$\langle delimiter type \rangle diagmat$	8, 38
\sqcup	143, 233, 356, 833
\lvert	50, 164, 165, 184, 250, 252, 256
A	
$\backslash A$	666
$\backslash aab$	228
$\backslash ab$	3-6, 13, 15, 23-25, 27, <u>213</u> , 825, 826, 827, 828
ab.braket (option)	5
$\backslash abs$	11, 838
$\backslash acomm$	14
$\backslash acos$	<u>773</u>
$\backslash acot$	<u>773</u>
$\backslash acsc$	<u>773</u>
$\backslash active$	250, 267, 268
$\backslash aftergroup$	58
$\backslash array$	545, 546, 626, 627
$\backslash asec$	<u>773</u>
$\backslash asin$	<u>773</u>
$\backslash atan$	<u>773</u>
$\backslash AtBeginDocument$	44, 49, 68, 280, 811, 822
$\backslash AtEndOfPackage$	22
B	
$\backslash Bab$	226
$\backslash bab$	225
$\backslash baselineskip$	532
$\backslash Bdiagmat$	8, 16, <u>564</u>
$\backslash bdiagmat$	8, 16, <u>564</u>
$\backslash begin$	609, 614, 749, 754
$\backslash begingroup$	24-26, 205, 249, 251, 255, 266, 269, 275, 438, 480
$\backslash beval$	11, 13, 850
$\backslash bgroup$	57, 252, 254, 256, 258
$\backslash Big$	5, 116
$\backslash big$	4, 5, 23, 25, 114
$\backslash Bigg$	3, 5, 120
$\backslash bigg$	5, 118
$\backslash Biggg$	3-5, 62, 66, 70, 76, 77, 78, 124
$\backslash biggg$..	3, 5, 61, 64, 69, 73, 74, 75, 122, 198
$\backslash Bigggl$	3, 76, 125
$\backslash bigggl$	3, 73, 123
$\backslash Bigggm$	3, 77
$\backslash bigggm$	3, 74
$\backslash Bigggr$	3, 78, 125
$\backslash bigggr$	3, 75, 123
$\backslash Biggl$	121
$\backslash biggl$	119
$\backslash Biggr$	121
$\backslash biggr$	119
$\backslash Bigl$	117
$\backslash bigl$	115
$\backslash Bigr$	117
$\backslash bigr$	115
$\backslash bknorm$	7, <u>291</u> , <u>446</u>
$\backslash bm$	14
$\backslash boldsymbol$	14
bool commands:	
$\backslash bool_if:N\TF$	711, 718, 726, 739
$\backslash bool_lazy_or:nn\TF$	605, 745
$\backslash bool_new:N$	634, 635

<code>\bool_set_false:N</code>	679, 693		
<code>\bool_set_true:N</code>	682, 686, 696, 700		
box commands:			
<code>\box_dp:N</code>	86, 92, 98, 104		
<code>\box_ht:N</code>	85, 91, 97, 103		
<code>\box_new:N</code>	8		
<code>\box_set_dp:Nn</code>	86, 92, 98, 104		
<code>\box_set_ht:Nn</code>	85, 91, 97, 103		
<code>\box_use:N</code>	535		
<code>\box_use_drop:N</code>	87, 93, 99, 105, 535		
<code>\l_tmpa_box</code>	84, 85, 86, 90,		
	91, 92, 96, 97, 98, 102, 103, 104, 534, 535		
<code>\bra</code>	5, 28, 32, <u>239</u> , 243, <u>362</u>		
<code>\braket</code> (option)	5		
<code>\braket</code>	5, 6, 29, 33, <u>249</u> , <u>378</u> , 418		
<code>\Bxmat</code>	9, 16		
<code>\bxmat</code>	9, 16		
		C	
<code>\catcode</code>	250, 267, 268		
<code>\cdot</code>	14, 826		
<code>\cdots</code>	714, 742		
clist commands:			
<code>\clist_count:N</code>	577		
<code>\clist_gpop:NN</code>	588		
<code>\clist_map_inline:n</code>			
	564, 648, 773, 783, 864		
<code>\clist_new:N</code>	551		
<code>\clist_set:Nn</code>	576		
<code>\comm</code>	14		
<code>\count</code>	12		
<code>\cross</code>	14		
cs commands:			
<code>\cs_new:Npn</code>	16, 79, 573, 643, 669		
<code>\cs_new_nopar:Npn</code>	12, 14		
<code>\cs_set_eq:NN</code>	24, 25, 26, 108		
<code>\csname</code> 112, 114, 116, 118, 120, 122, 124,			
	130, 155, 202, 204, 207, 209, 210, 219,		
	297, 311, 332, 342, 366, 374, 395, 398,		
	399, 401, 402, 403, 405, 406, 407, 408,		
	412, 436, 437, 460, 478, 479, 803, 844		
<code>\curl</code>	14, <u>773</u> , <u>821</u>		
		D	
<code>\ddots</code>	725, 728		
<code>\DeclareDocumentCommand</code>	155,		
	215, 291, 301, 315, 362, 370, 378,		
	427, 446, 469, 488, 566, 650, 785, 840		
<code>\DeclareRobustCommand</code>	57, 58, 61,		
	62, 64, 66, 69, 70, 73, 74, 75, 76, 77,		
	78, 213, 243, 248, 265, 290, 532, 776,		
	779, 780, 781, 782, 815, 816, 825, 826,		
	827, 828, 858, 859, 860, 861, 862, 872		
<code>\def</code>	4, 5, 10, 11, 27, 30, 33, 36, 37,		
	50, 51, 52, 53, 55, 56, 111, 112, 114,		
	116, 118, 120, 122, 124, 126, 137, 144,		
	154, 196, 198, 200, 203, 204, 208, 209,		
	214, 252, 253, 256, 257, 260, 271, 272,		
	276, 281, 282, 284, 285, 397, 400, 404,		
	409, 426, 519, 520, 521, 522, 523, 524,		
	531, 536, 766, 767, 768, 769, 835, 857		
<code>\delclose</code>	3, 19, 22, <u>57</u> , 149		
<code>\delimiter</code>	55, 56		
<code>\delopen</code>	3, 19, 22, <u>57</u> , 148		
<code>\detokenize</code>	130		
<code>\diagmat</code>	8, 16, <u>564</u>		
<code>\dimexpr</code>	536		
<code>\displaystyle</code>	84		
<code>\div</code>	14, 15, <u>773</u> , <u>821</u>		
<code>\divsymbol</code>	10, 15, 824		
<code>\doublecross</code>	8, 537		
<code>\doubledot</code>	8, 538		
		E	
<code>\edef</code>	19, 20, 45, 46, 127, 138, 410		
<code>\egroup</code>	58, 252, 254, 256, 258		
<code>\else</code>	38,		
	39, 63, 126, 128, 132, 150, 199, 206,		
	321, 326, 334, 409, 411, 416, 501, 509		
<code>\end</code>	611, 616, 751, 756		
<code>\endcsname</code>	49, 112, 114,		
	116, 118, 120, 122, 124, 130, 155, 197,		
	202, 204, 207, 209, 210, 219, 280, 297,		
	311, 332, 342, 366, 374, 395, 398, 399,		

401, 402, 403, 405, 406, 407, 408, 412,
 436, 437, 460, 478, 479, 803, 824, 844
 \endgroup 24,
 26, 177, 179, 181, 183, 185, 187, 189,
 191, 193, 195, 205, 239, 244, 254, 258,
 259, 261, 273, 274, 279, 286, 443, 485
 \endinput 230, 353, 513, 851
 \erf 773
 \erfc 773
 \eval 11, 13, 848
 exp commands:
 \exp_args:Nc .. 566, 650, 776, 785, 872
 \exp_args:Ne 545, 546, 626, 627
 \exp_args:Nnx 801
 \expandafter
 ... 43, 46, 112, 114, 116, 118, 120,
 122, 124, 129, 130, 136, 154, 204, 209,
 297, 311, 331, 341, 412, 425, 460, 803
 \expanded 296, 310, 330, 340
 \expval 7, 291, 446

F

\fam 19
 \fi 42,
 44, 54, 71, 72, 134, 135, 136, 153, 201,
 206, 238, 283, 325, 345, 346, 361, 421,
 423, 424, 425, 464, 505, 511, 818, 824
 file commands:
 \file_input_stop:
 139, 539, 620, 760, 830, 875
 \fontencoding 792
 \fontfamily 794
 \fontseries 792
 \fontshape 792
 \fontsize 534

G

\gdef 251, 255, 269, 275, 382, 384, 385, 386,
 389, 390, 391, 392, 450, 451, 452, 455,
 456, 457, 493, 494, 495, 498, 499, 500
 \grad 14, 773, 821
 group commands:
 \group_begin: 575, 671, 787

\group_end: 618, 758, 807

H

\hbox 65, 67
 hbox commands:
 \hbox_set:Nn .. 81, 84, 90, 96, 102, 534

I

\Identity 10, 773
 \identity 10, 773
 \IfBooleanTF 216,
 293, 303, 322, 328, 335, 349, 363, 371,
 380, 429, 448, 471, 491, 568, 652, 842
 \ifcsname 21, 49, 129, 197, 280, 412, 460, 824
 \ifdefined 60, 68, 235, 358
 \ifnum 417
 \IfValueT 458
 \IfValueTF 218, 295,
 309, 318, 365, 373, 435, 477, 793, 843
 \ifx 19, 38, 39, 126, 128, 147, 206,
 320, 321, 327, 409, 411, 501, 507, 810
 \Im 10, 15, 44, 810
 \imat 12
 \ImSymbol 10, 15, 44, 814
 int commands:
 \int_compare:nNnTF
 578, 586, 594, 676, 690
 \int_gset:Nn 629, 631, 641, 642
 \int_new:N 552, 636, 637
 \int_set:Nn 577, 678, 692
 \int_step_inline:nn 702
 \int_step_inline:nnn
 582, 584, 706, 734
 \iu 773

K

\kbproj 7, 291, 446
 \ket 5, 28, 32, 244, 248, 370
 \ketbra 5-7, 34, 266, 427
 keys commands:
 \keys_define:nn 559, 657
 \keys_set:nn 580, 673

L		N	
<code>\langle</code>	3, 174, 175, 187, 194, 195, 228, 239, 240, 254, 258, 271, 278, 293, 306, 364, 366, 367, 398, 401, 405, 432, 437, 441, 474, 479, 483	<code>\NeedsTeXFormat</code>	3
<code>\laplacian</code>	773, 821	<code>\newcount</code>	12
<code>\lbrace</code>	3, 168, 169, 181, 188, 189, 226	<code>\newtoks</code>	12
<code>\ldotp</code>	8, 528	<code>\noexpand</code>	20, 40, 41, 296, 297, 310, 311, 331, 341, 342
<code>\left</code>	19, 22, 57, 65, 67, 151	<code>\nolimits</code>	777, 780, 815, 816
<code>\let</code>	18, 22, 28, 148, 149, 151, 152, 206, 813, 814, 824	<code>\norm</code>	11, 839
<code>\letdif</code>	820	O	
<code>\lineskip</code>	533	options:	
M		<code>ab.braket</code>	5
<code>\mathbin</code>	532	<code>braket</code>	5
<code>\mathcal</code>	771, 835	<code>\order</code>	11, 840
<code>\mathchar</code>	284, 285	P	
<code>\mathclose</code> ..	57, 75, 78, 113, 177, 179, 181, 183, 185, 187, 189, 191, 193, 195, 239, 244, 254, 273, 278, 293, 305, 306, 364, 372, 431, 432, 442, 473, 474, 484	<code>\pab</code>	27, 224
<code>\mathcode</code>	252, 256, 270	<code>\PackageError</code>	418
<code>\mathop</code>	777, 780, 815, 816	<code>\PackageWarning</code>	236, 359
<code>\mathopen</code> ...	57, 73, 76, 113, 177, 179, 181, 183, 185, 187, 189, 191, 193, 195, 239, 244, 254, 273, 277, 293, 305, 306, 364, 372, 431, 432, 440, 473, 474, 482	<code>\patchcmd</code>	545, 546, 626, 627
<code>\mathord</code>	779	<code>\pb</code>	14
<code>\mathrel</code>	74, 77, 253, 257, 414	<code>\pdiagmat</code>	8, 16, 564
<code>\mathrm</code>	781, 782	<code>\peval</code>	11, 13, 849
<code>\mbox</code>	790	phx internal commands:	
<code>\MessageBreak</code>	237, 360	<code>_phx_define_key:nnnn</code> ..	12, 12, 24, 519, 520, 521, 522, 523, 524, 547, 549, 628, 630, 632, 765, 767, 768, 769
<code>\mid</code>	19, 53, 56	<code>\l_phx_diagmat_tl</code>	551, 581, 602, 610, 615
<code>\middle</code>	260	<code>_phx_diagmat_type:nnn</code>	38, 569, 570, 573, 573
<code>\mkern</code>	782	<code>_phx_if_digits_only:n</code>	664
mode commands:		<code>_phx_if_digits_only:nTF</code> ...	41, 664, 674, 688
<code>\mode_if_math:TF</code>	788	<code>_phx_if_digits_only_p:n</code> ...	664
<code>\mqty</code>	16	<code>\l_phx_mat_align_c_tl</code> ..	557, 607
msg commands:		<code>\l_phx_mat_align_tl</code>	550, 551, 606, 607, 614
<code>\msg_error:nn</code>	801	<code>\l_phx_mat_diag_clist</code>	551, 576, 577, 588
MT commands:		<code>\l_phx_mat_dim_int</code>	551, 577, 578, 579, 582, 584, 594
<code>\MT_matrix_begin:N</code>	545, 546, 626, 627		

<code>\l__phx_mat_empty_tl</code>	<code>\PV</code>	10 , 773
..... 548 , 551 , 590 , 593	<code>\pv</code>	10 , 773
<code>\l__phx_mat_line_tl</code>	<code>\pxmat</code>	9 , 16
. 551 , 590 , 591 , 593 , 596 , 599 , 602 , 603		
<code>__phx_mathvphantom:n</code> .		Q
79 , 79 , 108	<code>\q</code>	11
<code>__phx_processkeyopt:n</code>	<code>\qc</code>	861
..... 12 , 16 , 26 , 530 , 558 , 647 , 772	<code>\qcc</code>	862
<code>__phx_setkeys:nn</code> 12 , 14 , 25 , 525 , 770	<code>\qcomma</code>	860 , 861
<code>\l__phx_tmpa_box</code> ...	<code>\qq</code>	859
8 , 81 , 85 , 86 ,	<code>\qqtext</code>	858 , 859
87 , 91 , 92 , 93 , 97 , 98 , 99 , 103 , 104 , 105	<code>\qty</code>	13
<code>\l__phx_tmpa_tl</code> 38 , 109 , 588 , 589 , 591	<code>\quad</code>	11 , 857 , 858 , 860 , 863 , 873
<code>\l__phx_tmpb_tl</code>		
109		R
<code>\l__phx_xmat_align_c_tl</code>	<code>\rangle</code>	3 , 174 , 175 ,
..... 551 , 634 , 747	187 , 194 , 195 , 228 , 244 , 245 , 254 , 258 ,
<code>\l__phx_xmat_align_tl</code>	271 , 277 , 293 , 305 , 372 , 374 , 375 , 399 ,
..... 633 , 634 , 662 , 746 , 747 , 754	<code>\rank</code>	403 , 408 , 431 , 436 , 440 , 473 , 478 , 482
<code>__phx_xmat_entry_format:nnn</code>	<code>\rbrace</code>	3 , 168 , 169 , 181 , 188 , 189 , 226
. 643 , 659 , 705 , 709 , 714 , 733 , 737 , 742	<code>\Re</code>	10 , 15 , 44 , 810
<code>\l__phx_xmat_extra_cdots_bool</code>	regex commands:	
.....	<code>\regex_match:nnTF</code>	666
634 , 693 , 696 , 700 , 711 , 726 , 739	<code>\relax</code>	19 , 24 , 25 , 22 , 38 ,
<code>\l__phx_xmat_extra_vdots_bool</code>	51 , 52 , 55 , 126 , 138 , 206 , 409 , 426 , 536
.....	<code>\RequirePackage</code> ...	9 , 544 , 625 , 819 , 856
634 , 679 , 682 , 686 , 718	<code>\RequirePhxmodu</code>	18 , 27 , 234 , 357 , 821 , 834
<code>\l__phx_xmat_showleft_int</code> ...	<code>\Res</code>	773
. 631 , 634 , 661 , 690 , 692 , 706 , 721 , 734	<code>\res</code>	773
<code>\l__phx_xmat_showtop_int</code> ...	<code>\Resymbol</code>	10 , 15 , 44 , 813
.....	<code>\right</code>	19 , 22 , 58 , 65 , 67 , 152
629 , 634 , 660 , 676 , 678 , 702	<code>\romannumeral</code>	422
<code>\l__phx_xmat_tl</code>		
. 634 , 672 , 704 , 708 , 713 , 716 , 720 ,		
723 , 728 , 730 , 732 , 736 , 741 , 750 , 755		
<code>__phx_xmat_type:nnnnn</code>		
..... 41 , 653 , 654 , 669 , 669		
prg commands:		
<code>\prg_new_conditional:Npnn</code> ...		664
<code>\prg_replicate:nn</code>		721
<code>\prg_return_false:</code>		667
<code>\prg_return_true:</code>		667
<code>\ProcessOptions</code>		17
<code>\protect</code>		230 , 353 , 513 , 851
<code>\protected</code>		50 , 51 , 52 , 53 , 55 , 56
<code>\ProvidesExplFile</code>		517 , 542 , 623 , 763 , 854
<code>\ProvidesExplPackage</code>		6
<code>\ProvidesFile</code>		143 , 233 , 356 , 833
		S
	<code>\scriptscriptstyle</code>	102
	<code>\scriptstyle</code>	96
	<code>\selectfont</code>	795
	<code>\setcounter</code>	579
	<code>\setkeys</code>	15
	<code>\space</code>	418 , 803

`\string` 23, 24,
 112, 114, 116, 118, 120, 122, 124, 129,
 155, 197, 207, 210, 412, 418, 460, 803
`\symbf` 14
`\symbfit` 14
`\symoperators` . 19, 50, 51, 52, 53, 281, 282

T

TeX and L^AT_EX 2_ε commands:

`\@currentt` 19
`\@currname` 19
`\@empty` . 18, 45, 126, 127, 128, 138,
 320, 321, 327, 409, 410, 411, 501, 507
`\@ifnextchar` 30, 33
`\@ifstar` 858, 863, 873
`\@nnil` 38, 39, 46
`\@onefilewithoptions` 40
`\@onlypreamble` 29, 32, 35, 48
`\@phx@abtight` 144, 147
`\@phx@bk@cargnum` . . 382, 389, 395, 422
`\@phx@bk@do@pt` 409, 425, 426
`\@phx@bk@l` 384, 390, 398, 401,
 405, 413, 450, 455, 461, 493, 498, 502
`\@phx@bk@m` 385, 391, 402, 406,
 407, 414, 451, 456, 462, 494, 499, 503
`\@phx@bk@r` 386, 392, 399, 403,
 408, 415, 452, 457, 463, 495, 500, 504
`\@phx@dbl@c` 519, 537
`\@phx@dbl@d` 520, 538
`\@phx@dbl@oc` 523, 537
`\@phx@dbl@od` 524, 538
`\@phx@dbl@sc` 521, 537
`\@phx@dbl@sd` 522, 538
`\@phx@ev` 131,
 138, 320, 332, 342, 501, 502, 503, 504
`\@phx@ev@basis` 133, 138,
 321, 323, 324, 327, 337, 343, 507, 510
`\@phx@ev@do@pt` 126, 136, 138
`\@phx@oplega@PV` 768, 779
`\@phx@oplega@pv` 769, 780
`\@pkgextension` 27, 41
`\@optionlist` 19

`\@unprocessedoptions` 22
`\bBigg@` 19, 60, 61, 62, 68, 69, 70
`\define@key` 13
`\f@size` 536
`\leavevmode@ifvmode` 64, 66
`\n@space` 65, 67
`\operator@font` . . . 777, 780, 815, 816
`\p@` 65, 67
`\phx@@ab@bk` 255, 262
`\phx@@ab@kb` 275, 287
`\phx@@end` 275, 287
`\phx@@mb@bk` 251, 261
`\phx@@mb@kb` 269, 286
`\phx@ab@C` 25
`\phx@AB@gen` 25, 154, 205,
 239, 240, 244, 245, 261, 262, 286, 287
`\phx@ab@ordersym` 835, 841
`\phx@abb@bkv` 256, 260, 358
`\phx@abb@l` 272, 276, 281, 284
`\phx@abb@r` 272, 276, 282, 285
`\phx@abc@close` 22, 28, 31, 111,
 147, 157, 159, 161, 163, 165, 167, 169,
 171, 173, 175, 220, 240, 245, 258, 277,
 278, 367, 375, 440, 442, 482, 484, 845
`\phx@ab@open` 22, 28, 31, 111,
 147, 157, 159, 161, 163, 165, 167, 169,
 171, 173, 175, 220, 240, 245, 258, 277,
 278, 367, 375, 439, 441, 481, 483, 845
`\phx@bk@do@pt` 409
`\phx@bk@doopt` 33, 383, 393, 409
`\phx@bk@in@⟨n.roman⟩` 33
`\phx@bk@in@i` 397, 508
`\phx@bk@in@ii` 397, 467
`\phx@bk@in@iii` 397, 510
`\phx@bra@@` 235
`\phx@d@l@genxa`
 . . . 24, 26, 203, 212, 242, 247, 264, 289
`\phx@d@l@genxm`
 . . . 24, 26, 203, 211, 241, 246, 263, 288
`\phx@d@l@geny`
 27, 214, 838, 839, 848, 849, 850

<code>\phx@d@lx</code>	23, 26, 196, 213, 243, 248, 265, 290, 296, 298, 310, 312, 323, 324, 328, 331, 337, 341, 350	<code>\z@</code>	534
<code>\phx@d@lx(ab)</code>	24	<code>\z@skip</code>	532
<code>\phx@d@lx(mb)</code>	24	<code>\zap@space</code>	45, 127, 410
<code>\phx@d@lxab</code>	23, 24, 27, 211	tex commands:	
<code>\phx@d@lxmb</code>	23, 24, 27, 211	<code>\tex_mathchoice:D</code>	82
<code>\phx@date</code>	4, 6, 143, 233, 356, 517, 542, 623, 763, 833, 854	<code>\text</code>	857
<code>\phx@dbl@gen</code>	531, 537, 538	<code>\textstyle</code>	90
<code>\phx@dbl@curr@size</code>	533, 534, 536	<code>\times</code>	8, 14, 527, 827
<code>\phx@define@key</code>	17, 22, 24, 144, 835	tl commands:	
<code>\phx@del</code>	111	<code>\tl_const:Nn</code>	557, 640
<code>\phx@del*</code>	111	<code>\tl_gclear:N</code>	581, 603, 672
<code>\phx@del\Big</code>	111	<code>\tl_gput_right:Nn</code>	590, 591, 593, 596, 599, 602
<code>\phx@del\big</code>	111	<code>\tl_gset:Nn</code>	548, 550, 633
<code>\phx@del\Bigg</code>	111	<code>\tl_if_empty:NTF</code>	589
<code>\phx@del\bigg</code>	111	<code>\tl_if_empty_p:N</code>	606, 746
<code>\phx@del\Biggg</code>	111	<code>\tl_if_eq:nnTF</code>	796, 797
<code>\phx@del\biggg</code>	111	<code>\tl_if_eq_p:NN</code>	607, 747
<code>\phx@ev@doopt</code>	137, 317, 490	<code>\tl_new:N</code>	109, 110, 553, 554, 555, 556, 638, 639
<code>\phx@false</code>	11	<code>\tl_put_right:Nn</code>	704, 708, 713, 716, 720, 723, 728, 730, 732, 736, 741
<code>\phx@FW@opti@ns</code>	34, 36, 48	<code>\tl_use:N</code>	610, 615, 750, 755
<code>\phx@FW@options</code>	31, 33, 35	token commands:	
<code>\phx@FW@options</code>	27, 30, 32	<code>\c_math_subscript_token</code>	645
<code>\phx@mathvphantom</code>	20, 108, 277, 278, 440, 442, 482, 484	<code>\Tr</code>	773
<code>\phx@mb@()</code>	25	<code>\tr</code>	773
<code>\phx@nl@nabla</code>	820, 825, 826, 827, 828	U	
<code>\phx@oplega@id</code>	767, 794	<code>\Udelimiter</code>	50, 51, 52, 53
<code>\phx@processkeyopt</code>	17, 22, 26, 146, 837	<code>\Umathchar</code>	281, 282
<code>\phx@qtext@</code>	857, 858, 863, 873	<code>\unexpanded</code>	40
<code>\phx@reserveda</code>	766, 810	<code>\upe</code>	773
<code>\phx@setkeys</code>	17, 22, 20, 25, 145, 836	<code>\usepackage</code>	2
<code>\phx@temp</code>	17	<code>\usephxmodu</code>	2, 13, 18, 27
<code>\phx@tempa</code>	206, 207	V	
<code>\phx@true</code>	10, 147, 810	<code>\va</code>	14
<code>\phx@version</code>	5, 6, 517, 542, 623, 763, 854	<code>\Vab</code>	229
<code>\reserved@a</code>	18, 19, 20, 21, 45, 46, 47, 127, 128, 130, 131, 133, 198, 200, 202, 410, 411, 412, 413, 414, 415, 417, 422	<code>\vab</code>	227
<code>\reserved@b</code>	37, 43, 46	<code>\value</code>	578, 641, 642
		<code>\vb</code>	14
		<code>\vbox</code>	65, 67

<code>\vcenter</code>	532	478, 479, 481, 484, 838, 848, 849, 850
<code>\Vdiagmat</code>	8, 16, <u>564</u>	<code>\vphantom</code>
<code>\vdiagmat</code>	8, 16, <u>564</u>	20
<code>\vdot</code>	14	<code>\vu</code>
<code>\vdots</code>	720, 723, 728	14
<code>\vec</code>	14	<code>\Vxmat</code>
<code>\Vert</code>	3, 19,	9, 16
	52, 55, 172, 173, 185, 192, 193, 229, 839	<code>\vxmat</code>
<code>\vert</code>	3, 19, 51, 170, 171, 183,	
	190, 191, 227, 239, 240, 244, 245, 252,	
	260, 273, 277, 278, 293, 305, 306, 364,	
	366, 367, 372, 374, 375, 402, 406, 407,	
	431, 432, 436, 437, 439, 442, 473, 474,	
		X
		<code>\xdef</code>
		131, 133, 413, 414,
		415, 422, 461, 462, 463, 502, 503, 504
		<code>\xmat</code>
		9, 16
		Z
		<code>\Z</code>
		666