The HEP-MATH package*

Math macros for operators, derivatives, delimiters, bra-ket notation and expectation values

Jan Hajer†

2026/11/13

Abstract

The HEP-MATH package extends MATHTOOLS and AMSMATH with HEP-oriented math notation. It provides math-only command definitions, improved overlines and arrow accents, operator names, fractions and unit helpers, differentials and derivative builders, paired delimiters, probabilities, brackets, quantum-mechanical bra-ket notation, matrix elements, and expectation values.

To use the package place `\usepackage{hep-math}` in the preamble.

The MATHTOOLS [1] package is loaded, which in turn loads the $\mathcal{AMS}$-LaTeX AMSMATH [2] package. Horizontal spacing in inline equations and page breaks in block equations are marginally adjusted.

`\left` `\right` Spacing around `\left` and `\right` is fixed with the MLEFTRIGHT package [3].

1 Macros

`\mathdef` The `\mathdef{`⟨*name*⟩`}[`⟨*arguments*⟩`]{`⟨*code*⟩`}` macro (re-)defines macros only within math mode without changing the text mode definition.

`\i` `\d` The imaginary unit `\i` and the differential `\d` are defined using this functionality. Their math definitions are set at the beginning of the document, unless they have been defined with `\mathdef` in the preamble, e.g. via `\mathdef{\d}{\mathrm d}`. Repeated uses of `\mathdef` for the same macro replace its math definition and keep its text definition.

`\overline` The `\overline` macro is adjusted to work also outside of math mode using the SOUL [4] package. Text containing macros unknown to SOUL, such as `\ref{`⟨*key*⟩`}`, is overlined as a whole without line breaks.

`\oset` `\overleft` `\overright` `\overleftright` A better looking over left right arrow is defined i.e. $\overleftrightarrow{\partial}$ using a new `\oset{`⟨*over*⟩`}{`⟨*math*⟩`}` functionality.

Diagonal matrix `\diag`, signum `\sgn`, trace `\tr`, `\Tr`, and `\rank` operators are defined.

`\diag` The real and imaginary projectors are redefined to look like ordinary operators.

`\sgn` `\Re` `\Im` `\arccsc` `\sin` `\cos` `\tan` `\arccsc` and other inverse trigonometric functions are defined, including the short forms `\asin`, `\acos`, and so on.

`\cos`, `\tan`, and all other trigonometric functions and their inverses are adjusted to have the same height as `\sin`.

*This document corresponds to HEP-MATH v1.6.

†jan.hajer@tecnico.ulisboa.pt

1.1 Fractions and units

\unit \inv

The correct spacing for units is provided by the macro `\unit[⟨value⟩]{⟨unit⟩}` from the SIUNITX package [5] which can also be used in text mode. The macro `\inv[⟨power⟩]{⟨text⟩}` allows to avoid math mode also for inverse units such as $5\,\mathrm{fb}^{-1}$ typeset via `\unit[5]{\inv{fb}}`.

\nicefrac \flatfrac \textfrac

The `\frac{⟨number⟩}{⟨number⟩}` macro is accompanied by `\nicefrac{⟨number⟩}{⟨number⟩}`, `\textfrac{⟨number⟩}{⟨number⟩}`, and `\flatfrac{⟨number⟩}{⟨number⟩}` leading to $\frac{1}{2}$, $1/2$, $1/2$, and 1/2. The `\textfrac` macro is mostly intended if a font with oldstyle numerals is used.

Some macros of the PHYSICS package [6] are reimplemented with a more conventional typesetting in mind. Finer details about mathematical typesetting can be found in [7].

1.2 Differentials and derivatives

\differential \newderivative \newpartialderivative

The three macros `\differential{⟨symbol⟩}`, `\newderivative {\name}{⟨symbol⟩}`, and `\newpartialderivative {\name}{⟨symbol⟩}` allow to define a differential with correct spacing, a derivative using this differential, and if necessary a partial derivative that can handle three dimensional derivatives.

\d \dv

These macros are used for the usual differential and derivative, producing dx via `\d x` and

`\dv[f]x`	`\dv*[f]x^n`	`\dv[f]x*^n`	`\dv*[f]x*^n`
$\frac{df}{dx}$	d^nf/dx^n	$\frac{d^nf}{dx^n}$	d^nf/dx^n
`\dv xf`	`\dv*xf`	`\dv x*f`	`\dv*x*f`
$\frac{d}{dx}f$	$d/dx\,f$	$\frac{d}{dx}f$	$d/dx\,f$

via `\dv*[⟨f⟩]{⟨x⟩}*^{⟨n⟩}`. The fraction follows the math style, like `\frac`. The first star selects the flat fraction `\flatfrac` and the second star the small fractions `\tfrac` and `\nicefrac`, respectively. Upright differential can be produced via `\renewcommand{\diffsymbol}{\mathrm d}`. The differential takes care of the correct spacing as long as it is placed at the end of the integral $\int f(x)\,dx$. In order to archive correct spacing when it is placed at the beginning of the integral it is advisable to place the whole expression in a `\mathop{\int\d x} f(x)` such that $\int dx\,f(x)$.

\pd \pdv

Similarly a partial differential and derivative are defined that can be used according to `\pdv*[⟨f⟩]{⟨x⟩}*^{⟨a⟩}[⟨y⟩]^{⟨b⟩}[⟨z⟩]^{⟨c⟩}`.

`\pdv[f]x`	`\pdv[f]x[y]`	`\pdv[f]x^3`	`\pdv[f]x^2[y]`
$\frac{\partial f}{\partial x}$	$\frac{\partial^2 f}{\partial x\partial y}$	$\frac{\partial^3 f}{\partial x^3}$	$\frac{\partial^3 f}{\partial x^2\partial y}$
`\pdv[f]x^2[y]^3`	`\pdv[f]x[y]^3`	`\pdv x[y]f`	
$\frac{\partial^5 f}{\partial x^2\partial y^3}$	$\frac{\partial^4 f}{\partial x\partial y^3}$	$\frac{\partial^2}{\partial x\partial y}f$	

Integer orders are summed in the numerator, while other orders are added, as in $\frac{\partial^{n+1} f}{\partial x^n\partial y}$ typeset via `\pdv[f]x^n[y]`.

\var

Similarly a functional variation and functional derivative are defined.

\fdv \cancel \slashed

The `\cancel{⟨characters⟩}` macro from the CANCEL package [8] and the `\slashed {⟨character⟩}` macro from the SLASHED package [9] allow to ~~cancel~~ math and use the Dirac slash notation i.e. $\not{\partial}$, respectively.

1.3 Paired delimiters

\abs \norm

`\abs x`	`\norm x`	`\norm[2]x`	`\norm*[2]x`
$\lvert x\rvert$	$\lVert x\rVert$	$\lVert x\rVert_2$	$\lVert x\rVert_2$

\eval
\order

```
\order x   \eval x_0^\infty   \eval* x_0^\infty
```
$$\mathcal{O}(x) \qquad x\big|_0^\infty \qquad x\Big|_0^\infty$$

\newpair
\comm
\acomm

The `\newpair{`⟨*name*⟩`}{`⟨*left delim*⟩`} {`⟨*right delim*⟩`}_{`⟨*subscript*⟩`}^{`⟨*superscript*⟩`}` macro is defined and used for the definition of (anti-)commutators and Poisson brackets.

```
\pb xy   \comm xy   \acomm xy
```
$$\{x,y\} \qquad [x,y] \qquad \{x,y\}$$

They can easily be redefined using e.g. `\newpair\comm\lbrack\rbrack_-`.

\bra
\ket
\braket
\ketbra
\mel
\ev
\vev

Macros for the bra-ket notation are introduced.

```
\bra x    \ket x     \braket xy      \ketbra xy
\mel xyz  \ev x    \ev[\Omega] x     \vev x
```
$$\langle x| \qquad |x\rangle \qquad \langle x|y\rangle \qquad |x\rangle\langle y|$$
$$\langle x|y|z\rangle \qquad \langle x\rangle \qquad \langle \Omega|x|\Omega\rangle \qquad \langle 0|x|0\rangle$$

\column
\row

Macros for row and column vectors are introduced together with a symbol for transpose vectors.

```
\column{x,y,z}   \row{x,y,z}^\trans
```
$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \qquad (x,\, y,\, z)^{\mathsf{T}}$$

1.4 PDF strings

If the HYPERREF package [10] is loaded, the macros also work in moving arguments that end up in PDF strings such as bookmarks. There `\i` and `\d` become i and d, overlines and slashes keep their argument, operators such as `\cos` keep their name, `\unit[5]{GeV}` becomes 5 GeV, `\inv{fb}` becomes fb^-1, `^\trans` becomes ^T, and fractions become flat as in 1/2. Other macros such as derivatives and paired delimiters keep only their arguments.

2 Environments

eqnarray

The `eqnarray` environment is depreciated, the `split`, `multline`, `align`, `multlined`, `aligned`, `alignedat`, and `cases` environments of the AMSMATH and MATHTOOLS packages should be used instead.

equation

Use the `equation` environment for short equations.

```
\begin{equation}
  left = right \ .
\end{equation}
```

$$\boxed{\text{left}} = \boxed{\text{right}}\,. \tag{2.1}$$

multline

Use the `multline` environment for longer equations.

```
\begin{multline}
  left = right 1 \\
  + right 2 \ .
\end{multline}
```

$$\boxed{\text{left}} = \boxed{\text{right 1}} \\ \boxed{+\ \text{right 2}}\,. \tag{2.2}$$

split

Use the `split` sub environment for equations in which multiple equal signs should be aligned.

```
\begin{equation} \begin{split}
  left &= right 1 \\
  &= right 2 \ .
\end{split} \end{equation}
```

$$\boxed{\text{left}} = \boxed{\text{right 1}} \\ = \boxed{\text{right 2}}\ . \tag{2.3}$$

`align` Use the `align` environment for the vertical alignment and horizontal distribution of multiple equations.

```
\begin{subequations} \begin{align}
  left &= right \ , &
  left &= right \ , \\
  left &= right \ , &
  left &= right \ .
\end{align} \end{subequations}
```

$$\boxed{\text{left}} = \boxed{\text{right}}\ , \quad \boxed{\text{left}} = \boxed{\text{right}}\ , \tag{2.4a}$$
$$\boxed{\text{left}} = \boxed{\text{right}}\ , \quad \boxed{\text{left}} = \boxed{\text{right}}\ . \tag{2.4b}$$

`aligned` Use the `aligned` environment within a `equation` environment if the aligned equations should be labeled with a single equation number.

`multlined` Use the `multlined` environment if either `split` or `align` contain very long lines.

```
\begin{equation} \begin{split}
  left &= right 1 \\ &=
  \begin{multlined}[t]
     right 2 \\ + right 3 \ .
  \end{multlined}
\end{split} \end{equation}
```

$$\boxed{\text{left}} = \boxed{\text{right 1}} \\ = \boxed{\text{right 2}} \\ \boxed{\text{+ right 3}}\ . \tag{2.5}$$

`alignat` Use the `alignat` environment together with the `\mathllap` macro for the alignment of multiple equations with vastly different lengths.

```
\begin{subequations}
\begin{alignat}{2}
  left &= long right && \ , \\
  le. 2 &= ri. 2 \ , &
  \mathllap{le. 3 = ri. 3} & \ .
\end{alignat}
\end{subequations}
```

$$\boxed{\text{left}} = \boxed{\text{long right}}\ , \tag{2.6a}$$
$$\boxed{\text{le. 2}} = \boxed{\text{ri. 2}}\ , \quad \boxed{\text{le. 3}} = \boxed{\text{ri. 3}}\ . \tag{2.6b}$$

As a rule of thumb if you have to use `\notag`, `\nonumber`, or perform manual spacing via `\quad` you are probably using the wrong environment.

A Implementation

<*package>

Load the MATHTOOLS package [1] which loads the AMSMATH package [2]. Allow page breaks within equations if necessary. Adjust the thick and med mu skips slightly.

```
\RequirePackage{mathtools}
\mathtoolsset{centercolon}
\numberwithin{equation}{section}
\allowdisplaybreaks[1]
\thickmuskip=5mu plus 3mu minus 1mu
\medmuskip=4mu plus 2mu minus 3mu
```

`\mathdef` Define the `\mathdef{`⟨*name*⟩`}[`⟨*arguments*⟩`]{`⟨*macro*⟩`}` macro which (re-)defines macros in math mode only. This macro is implemented using the XPARSE package [11]. The dispatch between the

two definitions is expandable, so that the macro also works in PDF strings, where a `\let` would be dropped. The text definition is copied with `\DeclareCommandCopy`, which also copies robust commands correctly. If the macro is still the dispatcher of an earlier `\mathdef`, only the math definition is replaced, so that the original text definition is kept and the dispatcher does not call itself. The dispatcher is recorded in `\hep@mathdef`⟨*name*⟩ for this test.

```
\RequirePackage{xparse}
\DeclareDocumentCommand{\mathdef}{mO{0}om}{%
  \hep@if@mathdef#1{}{%
    \ifdefined#1%
      \expandafter\DeclareCommandCopy\csname hep@text\string#1\endcsname#1%
    \else
      \expandafter\let\csname hep@text\string#1\endcsname=#1%
    \fi
  }%
  \expandafter\let\csname hep@math\string#1\endcsname\relax
  \expandafter\newcommand\expandafter*\csname hep@math\string#1%
  \IfNoValueTF{#3}{\endcsname[#2]}{\endcsname[#2][#3]}{#4}%
  \DeclareRobustCommand#1{%
    \csname hep@\ifmmode math\else text\fi\string#1\endcsname
  }%
  \expandafter\let\csname hep@mathdef\string#1\endcsname#1%
}
```

`\hep@if@mathdef` Test whether the macro is the dispatcher of an earlier `\mathdef`.

```
\newcommand*{\hep@if@mathdef}[1]{%
  \ifcsname hep@mathdef\string#1\endcsname
    \expandafter\ifx\csname hep@mathdef\string#1\endcsname#1%
      \expandafter\expandafter\expandafter\@firstoftwo
    \else
      \expandafter\expandafter\expandafter\@secondoftwo
    \fi
  \else
    \expandafter\@secondoftwo
  \fi
}
```

`\hep@mathdef@default` Define the default math meaning of a macro at the beginning of the document, unless the preamble already defined one with `\mathdef`.

```
\newcommand*{\hep@mathdef@default}[2]{%
  \AtBeginDocument{\hep@if@mathdef#1{}{\mathdef{#1}{#2}}}%
}
```

`\i` Provide an upright imaginary unit in math mode.

```
\newcommand*{\imaginaryunit}{i}
\hep@mathdef@default{\i}{\imaginaryunit}
```

`\hep@soul@if@safe` Test whether the SOUL package [4] can process a text. This is the case if every macro in the text, outside of inline math, is known to SOUL: registered with `\soulregister`, such as font commands

and accents, or handled by SOUL itself, such as `\\` and `\mbox`. Other macros may take arguments, which SOUL would separate from the macro, or look ahead, as `\xspace` does. The same test is defined by the HEP-TEXT package for `\underline`, so it is only defined if it does not exist yet. `\hep@soul@if@safe` is the LaTeX 2_ε name of the conditional `\hep_soul_if_safe:nTF`.

```
\ExplSyntaxOn
\cs_if_exist:NF \hep_soul_if_safe:nTF {
  \seq_new:N \l__hep_soul_seq
  \tl_new:N \l__hep_soul_tl
  \tl_new:N \l__hep_soul_known_tl
  \bool_new:N \l__hep_soul_bool
  \regex_const:Nn \c__hep_soul_math_regex { \cM\$ .*? \cM\$ }
  \regex_const:Nn \c__hep_soul_cs_regex { \c{.*} }
  \tl_const:Nn \c__hep_soul_known_tl {
    \\0\\ \\0\slash \\0\mbox \\0\hbox \\0\soulomit
    \\0\break \\0\linebreak \\0\- \\0\par
  }
  \prg_new_protected_conditional:Npnn \hep_soul_if_safe:n #1 { TF } {
    \group_begin:
    \cs_set:Npn \\ ##1 ##2 { \token_to_str:N ##2 , }
    \tl_set:No \l__hep_soul_known_tl { \the \SOUL@cmds }
    \tl_set:Ne \l__hep_soul_known_tl
      { , \l__hep_soul_known_tl \c__hep_soul_known_tl }
    \tl_set:Nn \l__hep_soul_tl {#1}
    \regex_replace_all:NnN \c__hep_soul_math_regex { } \l__hep_soul_tl
    \regex_extract_all:NVN \c__hep_soul_cs_regex
      \l__hep_soul_tl \l__hep_soul_seq
    \bool_set_true:N \l__hep_soul_bool
    \seq_map_inline:Nn \l__hep_soul_seq {
      \token_if_macro:NT ##1 {
        \exp_args:NNe \tl_if_in:NnF \l__hep_soul_known_tl
          { , \token_to_str:N ##1 , }
          { \bool_set_false:N \l__hep_soul_bool \seq_map_break: }
      }
    }
    \bool_if:NTF \l__hep_soul_bool
      { \group_end: \prg_return_true: }
      { \group_end: \prg_return_false: }
  }
  \cs_new_eq:NN \hep@soul@if@safe \hep_soul_if_safe:nTF
}
\ExplSyntaxOff
```

`\overline` Redefine `\overline` to be a text macro using the SOUL package [4]. Extend it as a math macro with the original definition from the AMSMATH package [2]. The SOUL package can only process text whose macros it knows, so text containing other macros, such as `\ref{`⟨*key*⟩`}` or `\enquote{`⟨*text*⟩`}`, is overlined as a whole with the TeX primitive instead.

```
\RequirePackage{soul}
% \def\overline#1{{\renewcommand*{\ULdepth}{-1.9ex}{}\uline{#1}}}
\newcommand*\textoverline[1]{%
```

```
  \hep@soul@if@safe{#1}{%
    {\setul{-1.9ex}{}\ul{#1}}%
  }{%
    \ensuremath{\m@th\@@overline{\hbox{#1}}}%
  }%
}
\let\overline\textoverline
\DeclareRobustCommand{\over@line}[1]{\@@overline{#1}}
\mathdef{\overline}{\over@line}
\newcommand*\hep@widebar[1]{%
  \mkern2.5mu\overline{\mkern-2.5mu#1\mkern-.5mu}\mkern.5mu%
}
\newcommand*\widebar[1]{%
  \settowidth{\dimen0}{\ensuremath{#1}}%
  \ifdim\dimen0>.475em\hep@widebar{#1}\else\bar{#1}\fi%
}
```

`\oset` Define a new overset macro `\oset[`⟨*offset*⟩`]{`⟨*over*⟩`}{`⟨*base*⟩`}`

```
\newcommand*{\oset}[3][-1pt]{%
  \text{\raisebox{.2ex}{$\mathop{#3}\limits^{%
    \vbox to#1{\kern-2\ex@\hbox{$\scriptscriptstyle#2$}\vss}%
  }$}}%
}
```

`\overleftright` Define a over left right arrow `\overleftright{`⟨*base*⟩`}`.

```
\newcommand*{\overleft}[1]{\oset{\leftarrow}{#1}}
\newcommand*{\overright}[1]{\oset{\rightarrow}{#1}}
\newcommand*{\overleftright}[1]{\oset{\leftrightarrow}{#1}}
```

`eqnarray` Undefine the `eqnarray` environment if not prevented by package option.

```
% \newif\ifhep@eqnarray\hep@eqnarraytrue
% \ifhep@eqnarray\else
%   \let\eqnarray\@undefined
%   \let\endeqnarray\@undefined
% \fi
```

A.1 Operators

`\tr` `\Tr` `\rank` `\erf` `\Res` `\sgn` `\sgn` `\diag` Provide the `\diag`, `\sgn`, and some other operators.

```
\DeclareMathOperator{\tr}{tr}
\DeclareMathOperator{\Tr}{Tr}
\DeclareMathOperator{\rank}{rank}
\DeclareMathOperator{\erf}{erf}
\DeclareMathOperator{\Res}{Res}
\DeclareMathOperator{\sgn}{sgn}
\DeclareMathOperator{\diag}{diag}
\let\det\relax\DeclareMathOperator{\det}{det}
```

Redefine the real and imaginary projectors.

```
\let\hep@Re\Re\let\Re\relax\DeclareMathOperator{\Re}{\hep@Re}
\let\hep@Im\Im\let\Im\relax\DeclareMathOperator{\Im}{\hep@Im}
```

Define a robust transpose symbol. UNICODE-MATH [12] already provides `\intercal` and clashes with AMSSYMB [13] if that is loaded after it.

```
\@ifpackageloaded{unicode-math}{}{\RequirePackage{amssymb}}
\newcommand*{\hep@transpose}[2]{\raisebox{\depth}{$\m@th#1\intercal$}}
\DeclareRobustCommand*{\transpose}{{\mathpalette\hep@transpose{}}}
\let\trans\transpose
```

A.1.1 Trigonometric functions

Adjust the height of the trigonometric functions to be equal to sin. The phantom i is typeset in the operator font, since a plain `\vphantom{i}` measures the math italic i.

```
\newcommand*{\hep@trig@strut}{\vphantom{\operator@font i}}
\let\cos\undefined\DeclareMathOperator{\cos}{cos\hep@trig@strut}
\let\tan\undefined\DeclareMathOperator{\tan}{tan\hep@trig@strut}
\let\cot\undefined\DeclareMathOperator{\cot}{cot\hep@trig@strut}
\let\sec\undefined\DeclareMathOperator{\sec}{sec\hep@trig@strut}
\let\csc\undefined\DeclareMathOperator{\csc}{csc\hep@trig@strut}
\let\arccos\undefined
\DeclareMathOperator{\arccos}{arccos\hep@trig@strut}
\let\arctan\undefined
\DeclareMathOperator{\arctan}{arctan\hep@trig@strut}
```

Define arc operators.

```
\DeclareMathOperator{\arccsc}{arccsc\hep@trig@strut}
\DeclareMathOperator{\arcsec}{arcsec\hep@trig@strut}
\DeclareMathOperator{\arccot}{arccot\hep@trig@strut}
```

Define shorthand for arc operators.

```
\DeclareMathOperator{\asin}{asin}
\DeclareMathOperator{\acos}{acos\hep@trig@strut}
\DeclareMathOperator{\atan}{atan\hep@trig@strut}
\DeclareMathOperator{\acsc}{acsc\hep@trig@strut}
\DeclareMathOperator{\asec}{asec\hep@trig@strut}
\DeclareMathOperator{\acot}{acot\hep@trig@strut}
```

Define csch and sech operators.

```
\DeclareMathOperator{\csch}{csch}
\DeclareMathOperator{\sech}{sech}
```

A.2 Units and fractions

\nicefrac Define the \nicefrac macro using the XFRAC package [14].

```
\RequirePackage{xfrac}
\let\nicefrac\sfrac
```

Load the SIUNITX package [5].

```
\RequirePackage{siunitx}
\providecommand*{\hep@lining@style}{\ifdefined\lstyle\lstyle\fi}
\sisetup{
  reset-math-version=false,
  reset-text-family=false,
  reset-text-series=false,
  uncertainty-mode=separate,
  exponent-product=\cdot,
  unit-mode=text,
  text-font-command=\hep@lining@style,
}
```

Provide a compatibility macro for older installations.

```
\ExplSyntaxOn
\cs_if_exist:NF \siunitx_declare_prefix:Nne
{
  \cs_generate_variant:Nn
  \siunitx_declare_prefix:Nnn
  { Nne }
}
\ExplSyntaxOff
```

\unit \unitfrac Ensure that its macros are compatible with the definitions used in the depreciated UNIT package.

```
\NewCommandCopy{\hep@unit}{\unit}
\ExplSyntaxOn
  \siunitx_declare_prefix:Nne\mu{-6}{\__siunitx_unit_non_latin:n{"03BC}}
  \prg_new_conditional:Npnn\hep_if_keyval:n#1{TF}{
    \tl_if_in:nnTF{#1}{=}{\prg_return_true:}{\prg_return_false:}
  }
  \cs_new_protected:Npn\hep_unit:nn#1#2{
    \hep_if_keyval:nTF{#1}{\hep@unit[#1]{#2}}
      {\qty[parse-numbers=false]{#1}{#2}}
  }
  \RenewDocumentCommand{\unit}{om}{
    \IfNoValueTF{#1}{\hep@unit{#2}}{\hep_unit:nn{#1}{#2}}
  }
\ExplSyntaxOff
\NewDocumentCommand{\unitfrac}{omm}{%
  \IfNoValueTF{#1}{\hep@unit{#2\per#3}}{\qty{#1}{#2\per#3}}%
}
```

`\inv` Provide a macro for the inverse, useful in combination with the unit macro in text mode.

```
\newcommand*{\inv}[2][1]{#2\ensuremath{^{-#1}}}
```

`\textfrac` Provide the `\textfrac` macro useful in combination with a font using lining numerals.

```
\newcommand*{\textfrac}[2]{\ensuremath{\nicefrac{\text{#1}}{\text{#2}}}}
```

`\flatfrac` Provide a flat fraction.

```
\DeclarePairedDelimiterX{\hep@flatfrac}[2]{.}{.}{%
  \kern-\nulldelimiterspace#1\delimsize/%
  \hep@left@delim#2\kern-\nulldelimiterspace%
}
\NewDocumentCommand{\flatfrac}{somm}{%
  \mathinner{%
    \IfBooleanTF{#1}{%
      \hep@flatfrac*{#3}{#4}%
    }{%
      \IfNoValueTF{#2}{\hep@left@delim#3/\hep@left@delim#4%
      }{%
       \hep@flatfrac[#2]{#3}{#4}%
      }%
    }%
  }%
}
```

A.2.1 Differentials and derivatives

`\int` Redefine `\int`.

```
% \let\hep@int\int
% \RenewDocumentCommand{\int}{oe{_^}}{%
%   \def\temp{\hep@int_{\IfValueT{#2}{#2}}^{\IfValueT{#3}{#3}}}%
%   \IfValueTF{#1}{\mathop{\temp#1}}{\temp}%
% }
```

`\differential` Define a generic differential `\differential`.

```
\newcommand*{\differential}[1]{\mathop{}\!#1}
```

`\newderivative` Define a generic derivative.

```
\newcommand*\newderivative[2]{
  \NewDocumentCommand{#1}{somse{^}}{%
    \IfBooleanTF{##4}{%
      \IfBooleanTF{##1}{\nicefrac}{\tfrac}%
    }{%
      \IfBooleanTF{##1}{\flatfrac}{\frac}%
    }{%
      \differential#2\IfValueT{##5}{^{##5\!}}\IfValueT{##2}{##2}%
    }{%
```

```
      \differential#2{{}##3}\IfValueT{##5}{^{##5}}%
    }%
  }
}
```

\hep@order@clear
\hep@order@add
\hep@order@power
\hep@power

Collect the order of a partial derivative. Integer exponents are summed, other exponents are added symbolically in front of the sum of the integers, so that e.g. the exponents n, 1, and 2 give $n + 3$. `\hep@order@power` prints the total order as superscript and `\hep@power{`⟨*exponent*⟩`}` a single exponent, both only if they differ from one.

```
\ExplSyntaxOn
\tl_new:N \l__hep_order_tl
\int_new:N \l__hep_order_int
\regex_const:Nn \c__hep_integer_regex { \A \s* [\+\-]? \s* \d+ \s* \Z }
\cs_new_protected:Npn \hep@order@clear {
  \tl_clear:N \l__hep_order_tl
  \int_zero:N \l__hep_order_int
}
\cs_new_protected:Npn \hep@order@add #1 {
  \regex_match:NnTF \c__hep_integer_regex {#1} {
    \int_add:Nn \l__hep_order_int {#1}
  } {
    \tl_if_empty:NF \l__hep_order_tl
      { \tl_put_right:Nn \l__hep_order_tl { + } }
    \tl_put_right:Nn \l__hep_order_tl {#1}
  }
}
\cs_new_protected:Npn \hep@order@power {
  \tl_if_empty:NTF \l__hep_order_tl {
    \int_compare:nNnF \l__hep_order_int = 1
      { { ^ { \int_use:N \l__hep_order_int \! } } }
  } {
    { ^ {
      \l__hep_order_tl
      \int_compare:nNnT \l__hep_order_int > 0 { + }
      \int_compare:nNnF \l__hep_order_int = 0
        { \int_use:N \l__hep_order_int }
      \!
    } }
  }
}
\cs_new_protected:Npn \hep@power #1 {
  \regex_match:NnTF \c__hep_integer_regex {#1} {
    \int_compare:nNnF {#1} = 1 { { ^ {#1} } }
  } {
    { ^ {#1} }
  }
}
\ExplSyntaxOff
```

newpartialderivative Define a generic partial derivative

```
\newcommand*\newpartialderivative[2]{%
  \NewDocumentCommand{#1}{somsE{^}{1}oE{^}{1}oE{^}{1}}{%
    \hep@order@clear
    \hep@order@add{##5}%
    \IfValueT{##6}{\hep@order@add{##7}}%
    \IfValueT{##8}{\hep@order@add{##9}}%
    \IfBooleanTF{##4}{%
      \IfBooleanTF{##1}{\nicefrac}{\tfrac}%
    }{%
      \IfBooleanTF{##1}{\flatfrac}{\frac}%
    }{%
      \differential#2\hep@order@power
      \IfValueT{##2}{##2}%
    }{%
      \differential#2{{}##3}\hep@power{##5}%
      \IfValueT{##6}{#2##6\hep@power{##7}}%
      \IfValueT{##8}{#2##8\hep@power{##9}}%
    }%
  }%
}
```

\diffsymbol \diff \d \derivative \dv Define the differential \d and the usual derivative.

```
\providecommand*{\diffsymbol}{d}
\newcommand*{\diff}{\differential\diffsymbol}
\hep@mathdef@default{\d}{\diff}
\newderivative{\derivative}{\diffsymbol}
\newcommand*\dv{\derivative}
```

\partialdifferential \pd \partialderivative \pdv Define the partial differential and derivative.

```
\newcommand*\partialdifferential{\differential\partial}
\newcommand*\slashedpartialdifferential{\differential{\slashed\partial}}
\newcommand*\spd{\slashedpartialdifferential}
\newcommand*\pd{\partialdifferential}
\newpartialderivative{\partialderivative}{\partial}
\newcommand*\pdv{\partialderivative}
```

\gaugediffsymbol \gaugediff \D Define the gauge covariant differential \D.

```
\providecommand*{\gaugediffsymbol}{D}
\newcommand*{\gaugediff}{\differential\gaugediffsymbol}
\newcommand*{\D}{\gaugediff}
```

\covariantdiff \cd Define the covariant differential \cd.

```
\newcommand*{\covariantdiff}{\differential\nabla}
\newcommand*{\cd}{\covariantdiff}
```

\variation Define the functional variation and derivative.
\var
\functionalderivative
\fdv

```
\newcommand*\variation{\differential\delta}
\newcommand*\var{\variation}
\newpartialderivative{\functionalderivative}{\delta}
\newcommand*\fdv{\functionalderivative}
```

\cancel Load the CANCEL [8] and SLASHED [9] packages which provide the \cancel and \slashed macros.
\slashed Make \slashed robust, so that it can be used in moving arguments.

```
\RequirePackage{cancel}
\RequirePackage{slashed}
\MakeRobust{\slashed}
\declareslashed{}{/}{.14}{0}{L}
\declareslashed{}{/}{.055}{0}{\partial}
\declareslashed{}{/}{.06}{0}{\gaugediff}
\declareslashed{}{/}{.055}{0}{\partialdifferential}
```

A.3 Paired delimiters

\left Load the MLEFTRIGHT package [3] and adjust the spacing around \left and \right.
\right

```
\RequirePackage{mleftright}
\mleftright
```

\noargumentsymbol Allow for macros to have an empty argument using the ETOOLBOX package [15].
\optionalargument

```
\RequirePackage{etoolbox}
\newcommand*{\noargumentsymbol}{\:\cdot\:}
\newcommand*{\optionalargument}[1]{\ifblank{#1}{\noargumentsymbol}{#1}}
```

\abs Absolute value and norm.
\norm

```
\DeclarePairedDelimiterX\abs[1]\lvert\rvert{\optionalargument{#1}}
\DeclarePairedDelimiterX\hep@norm[1]\lVert\rVert{\optionalargument{#1}}
\DeclarePairedDelimiterXPP\hep@pnorm[2]{}\lVert\rVert{_{#1}}{#2}
\NewDocumentCommand{\norm}{som}{%
  \IfValueTF{#2}{%
    \IfBooleanTF{#1}{\hep@pnorm*}{\hep@pnorm}{#2}%
  }{%
    \IfBooleanTF{#1}{\hep@norm*}{\hep@norm}%
  }{\optionalargument{#3}}%
}
```

\floor Floor and ceiling paired delimiters.
\ceil

```
\DeclarePairedDelimiter\ceil{\lceil}{\rceil}
\DeclarePairedDelimiter\floor{\lfloor}{\rfloor}
```

\ordersymbol Order symbol and macro.
\order

```
\providecommand*{\ordersymbol}{\mathcal{O}}
\DeclarePairedDelimiterXPP\order[1]{\ordersymbol}(){}{#1}
```

\evaluated Vertical evaluation bar
\eval

```
\DeclarePairedDelimiter{\hep@evaluated}{.}{\rvert}
\NewDocumentCommand{\evaluated}{som}{%
  \IfBooleanTF{#1}{%
    \hep@evaluated*{#3}%
  }{%
    \IfNoValueTF{#2}{#3\rvert}{\hep@evaluated[#2]{#3}}%
  }%
}
\newcommand*\eval{\evaluated}
```

\row Shortcuts for rows and columns
\column

```
\newcommand*{\rowseperator}{,\,}
\ExplSyntaxOn
\seq_new:N \hep@seq
\newcommand*{\hep@row}[1]{
 \seq_set_split:Nnn\hep@seq{,}{#1}
 \begin{matrix}\seq_use:Nn\hep@seq{\rowseperator}\end{matrix}
}
\newcommand*{\hep@column}[1]{%
 \seq_set_split:Nnn\hep@seq{,}{#1}%
 \begin{matrix}\seq_use:Nn\hep@seq{\\}\end{matrix}%
}
\ExplSyntaxOff
\DeclarePairedDelimiterX{\hep@row@}[1]{(}{)}{\hep@row{#1}}
\NewDocumentCommand{\hep@column@}{me{^}e{_}}{%
  \left(\hep@column{#1}\right)%
  \IfValueT{#2}{^{\!\!\!#2}}\IfValueT{#3}{_{\!\!\!#3}}%
}
\providecommand*{\column}{}
\mathdef{\column}{\hep@column@}
\providecommand*{\row}{}
\mathdef{\row}{\hep@row@}
```

A.3.1 Set and Probability

\midbar Define a generic midbar.

```
\newcommand*\hep@left@delim{\mathopen{}}
\providecommand*{\midbar}[1][]{%
  \nonscript\:#1\vert\allowbreak\nonscript\:\hep@left@delim%
}
```

\suchthat Define a \set macro that allows a midbar via \suchthat.
\set

```
\providecommand*\suchthat{\midbar}
\DeclarePairedDelimiterX\set[1]\{\}{%
  \renewcommand*\suchthat{\midbar[\delimsize]}#1%
}
```

\probabilitysymbol \given \Pr Redefine the \Pr macro to a macro that takes a \given macro and generates a midbar.

```
\providecommand*{\probabilitysymbol}{\operatorname{Pr}}
\providecommand*\given{\midbar}
\DeclarePairedDelimiterXPP\hep@Pr[1]{%
  \probabilitysymbol}(){}{%
  \renewcommand*\given{\midbar[\delimsize]}#1%
}
\let\Pr\relax
\NewDocumentCommand{\Pr}{so}{%
  \IfValueTF{#2}{%
    \IfBooleanTF{#1}{\hep@Pr*}{\hep@Pr}{#2}%
  }{%
    \probabilitysymbol%
  }%
}
```

A.3.2 Commutators

\newpair Define the \newpair macro that generates pairs surrounded by brackets.

```
\NewDocumentCommand{\newpair}{mmme{_}e{^}}{%
  \IfNoValueTF{#4}{%
    \IfNoValueTF{#5}{%
      \DeclarePairedDelimiterX{#1}[2]{#2}{#3}%
    }{%
      \DeclarePairedDelimiterXPP{#1}[2]{}{#2}{#3}{^{#5}}%
    }%
  }{%
    \DeclarePairedDelimiterXPP{#1}[2]{}{#2}{#3}{_{#4}}%
  }{%
    \optionalargument{##1},\optionalargument{##2}%
  }%
}
```

\innerproduct \poissonbracket \pb \commutator \comm \acomm Poissonbracket, commutator and anti-commutator.

```
\newpair\innerproduct\langle\rangle
\newpair\poissonbracket\lbrace\rbrace
\newpair\commutator\lbrack\rbrack
\newcommand*\pb{\poissonbracket}
\newcommand*\comm{\commutator}
\newcommand*\acomm{\poissonbracket}
```

A.3.3 Bra-ket notation

\braketspace Define the space within braket notation.

```
% \providecommand*\braketspace{\mskip1mu}
\providecommand*\braketouterspace{\mskip1mu}
\providecommand*\braketinnerspace{\mskip2mu}
\newcommand*\hep@midvert{%
```

```
  \braketinnerspace\delimsize\vert\braketinnerspace\hep@left@delim%
}
```

`\braket` Define the braket macro.

```
\DeclarePairedDelimiterX\braket[2]{\langle}{\rangle}{%
  \braketouterspace#1\hep@midvert#2\braketouterspace%
}
```

`\bra` Define the bra macro.

```
\DeclarePairedDelimiterXPP\hep@bra[1]{%
  }{\langle}{\rvert}{\braketinnerspace%
  }{\braketouterspace#1\braketinnerspace%
}
\NewDocumentCommand{\bra}{smt\ket sgt\ketbra sgg}{%
  \IfBooleanTF{#6}{%
    \IfBooleanTF{#1}{\braket*{#2}{#8}}{\braket{#2}{#8}}%
    \IfBooleanTF{#7}{\bra*{#9}}{\bra{#9}}%
  }{
    \IfBooleanTF{#3}{%
      \IfBooleanTF{#1}{\braket*}{%
        \IfBooleanTF{#4}{\braket*}{\braket}}{#2}{#5%
      }%
    }{%
      \IfBooleanTF{#1}{\hep@bra*}{\hep@bra}{#2}%
    }%
  }%
}
```

`\ket` Define the ket macro.

```
\DeclarePairedDelimiterXPP\ket[1]{%
  \braketinnerspace}{\lvert}{\rangle}{%
}{%
  \braketinnerspace\hep@left@delim#1\braketouterspace%
}
```

`\ketbra` Define the ketbra macro.

```
\NewDocumentCommand{\ketbra}{smm}{%
  \IfBooleanTF{#1}{%
    \ket*{#2}\bra*{#3}%
  }{%
    \ket{#2}\bra{#3}%
  }%
}
```

`\matrixelement` `\mel` Define the matrixelement macro.

```
\DeclarePairedDelimiterX\matrixelement[3]{%
  \langle}{\rangle
```

```
}{%
  \braketouterspace#1\hep@midvert#2\hep@midvert#3\braketouterspace%
}
\newcommand*\matrixel{\matrixelement}
\newcommand*\mel{\matrixelement}
```

\expectationvalue \ev \vev Define the expectationvalue and vev macros.

```
\DeclarePairedDelimiterX\hep@expvalue[1]{\langle}{\rangle}{%
  \braketouterspace#1\braketouterspace%
}
\NewDocumentCommand{\expectationvalue}{som}{%
  \IfNoValueTF{#2}{%
    \IfBooleanTF{#1}{\hep@expvalue*}{\hep@expvalue}{#3}%
  }{%
    \IfBooleanTF{#1}{\matrixelement*}{\matrixelement}{#2}{#3}{#2}%
  }%
}
\newcommand*\ev{\expectationvalue}
\newcommand*\vev[1]{\expectationvalue[0]{#1}}
```

A.4 PDF strings

\hep@mathpdf@ifkeyval Expandable test whether the optional argument of \unit is a key–value list.

```
\newcommand*{\hep@mathpdf@ifkeyval}[1]{%
  \hep@mathpdf@@ifkeyval#1=\hep@mathpdf@stop
}
\def\hep@mathpdf@@ifkeyval#1=#2\hep@mathpdf@stop{%
  \if\relax\detokenize{#2}\relax
    \expandafter\@secondoftwo
  \else
    \expandafter\@firstoftwo
  \fi
}
```

\hep@mathpdf@unprotect The operators of AMSMATH are protected, so that they would vanish from PDF strings. Replace them by unprotected copies, if they still are operators.

```
\newcommand*{\hep@mathpdf@unprotect}[1]{%
  \expandafter\hep@mathpdf@@unprotect#1\hep@mathpdf@stop#1%
}
\def\hep@mathpdf@@unprotect#1#2\hep@mathpdf@stop#3{%
  \ifx\qopname#1\def#3{#1#2}\fi
}
```

\hep@mathpdf@definitions Define the PDF-string versions of the macros, e.g. for bookmarks. The imaginary unit and the differential become i and d, overlines, slashes, and arrow accents keep only their base, and vectors are written as parentheses. Operators keep their name followed by a space, the probability \Pr keeps its brackets, transposes and inverses become a circumflex, fractions become flat, and

`\unit[`⟨*value*⟩`]{`⟨*unit*⟩`}` keeps its value. The PDF-string version of `\unit` from SIUNITX is copied at each use, since SIUNITX registers it at the beginning of the document.

```
\newcommand*{\hep@mathpdf@definitions}{%
  \@tfor\hep@mathpdf@op:=\arcsin\arg\cosh\coth\deg\dim\exp\gcd\hom\inf
    \ker\lg\lim\ln\log\max\min\sin\sinh\sup\tanh
    \do{\expandafter\hep@mathpdf@unprotect\hep@mathpdf@op}%
  \def\liminf{lim inf }%
  \def\limsup{lim sup }%
  \def\i{i}%
  \def\d{d}%
  \let\overline\@firstofone
  \let\widebar\@firstofone
  \let\slashed\@firstofone
  \RenewExpandableDocumentCommand{\oset}{omm}{##3}%
  \def\column##1{(##1)}%
  \def\row##1{(##1)}%
  \def\transpose{\textasciicircum T}%
  \let\trans\transpose
  \def\qopname##1##2##3{##3 }%
  \RenewExpandableDocumentCommand{\operatorname}{sm}{##2}%
  \let\hep@trig@strut\@empty
  \def\Pr{Pr}%
  \def\given{|}%
  \def\hep@Re{Re}%
  \def\hep@Im{Im}%
  \RenewExpandableDocumentCommand{\inv}{O{1}m}{%
    ##2\textasciicircum-##1%
  }%
  \RenewExpandableDocumentCommand{\nicefrac}{omom}{##2/##4}%
  \def\textfrac##1##2{##1/##2}%
  \RenewExpandableDocumentCommand{\flatfrac}{somm}{##3/##4}%
  \DeclareCommandCopy{\hep@mathpdf@unit}{\unit}%
  \DeclareCommandCopy{\hep@mathpdf@qty}{\qty}%
  \RenewExpandableDocumentCommand{\unit}{om}{%
    \IfNoValueTF{##1}{\hep@mathpdf@unit{##2}}{%
      \hep@mathpdf@ifkeyval{##1}%
        {\hep@mathpdf@unit[##1]{##2}}{\hep@mathpdf@qty{##1}{##2}}%
    }%
  }%
  \RenewExpandableDocumentCommand{\unitfrac}{omm}{%
    \IfNoValueTF{##1}%
      {\hep@mathpdf@unit{##2\per##3}}{\hep@mathpdf@qty{##1}{##2\per##3}}%
  }%
}
```

Register the PDF-string definitions if HYPERREF is loaded, before or after this package. This happens at the end of `\begin{document}`, after SIUNITX has registered its own definitions.

```
\AddToHook{begindocument/end}{%
  \@ifpackageloaded{hyperref}{%
```

```
    \pdfstringdefDisableCommands{\hep@mathpdf@definitions}%
  }{}%
}
```

</package>

B Test

<*test>

```
\documentclass{article}

\usepackage{hep-math}

\begin{document}

\begin{gather}
  \bra{x}\ket{y}
  \braket*{x}{y}\\
  \dv[f]{x}^3
  \pdv[f]{x}[y]^2[z]^3
  \fdv[f]{x}^3[y]\\
  \set{x \suchthat x \in X}
\end{gather}

\end{document}

```

</test>

C Readme

<*readme>

```
# The ‘hep-math‘ package

Math macros for operators, derivatives, delimiters, bra-ket notation
and expectation values

## Introduction

The ‘hep-math‘ package extends ‘mathtools‘ and ‘amsmath‘ with
HEP-oriented math notation. It provides math-only command definitions,
improved overlines and arrow accents, operator names, fractions and unit
helpers, differentials and derivative builders, paired delimiters,
probabilities, brackets, quantum-mechanical bra-ket notation, matrix
elements, and expectation values.

To use the package place ‘\usepackage{hep-math}‘ in the preamble.

```

```
## Author

Jan Hajer

## License

This file may be distributed and/or modified under the conditions of the
‘LaTeX‘ Project Public License, either version 1.3c of this license or
(at your option) any later version. The latest version of this license is
in ‘http://www.latex-project.org/lppl.txt‘ and version 1.3c or later is
part of all distributions of LaTeX version 2005/12/01 or later.
```

</readme>

References

[1] L. Madsen, M. Høgholm, W. Robertson, and J. Wright. ‘The mathtools package: Mathematical tools to use with amsmath’ (2004). CTAN: mathtools.

[2] *LaTeX Team*. ‘The amsmath package: AMS mathematical facilities for LaTeX’ (1994). CTAN: amsmath. URL: ams.org/tex/amslatex.

[3] H. Oberdiek. ‘The mleftright package: Variants of delimiters that act as maths open/close’ (2010). CTAN: mleftright.

[4] M. Franz. ‘The soul package: Hyphenation for letterspacing, underlining, and more’ (1998). CTAN: soul.

[5] J. Wright. ‘The siunitx package: A comprehensive (SI) units package’ (2008). CTAN: siunitx.

[6] S. C. de la Barrera. ‘The physics package: Macros supporting the Mathematics of Physics’ (2012). CTAN: physics.

[7] E. Gregorio. ‘TeX, LaTeX and math’ (2020). URL: latex-project.org/publications/2020-egreg-TUB-tb127gregorio-math.pdf.

[8] D. Arseneau. ‘The cancel package: Place lines through maths formulae’ (2013). CTAN: cancel.

[9] D. Carlisle. ‘The slashed package: Put a slash through characters’ (1987). CTAN: slashed.

[10] *LaTeX3 Project*. ‘Hypertext marks in LaTeX: a manual for hyperref: Extensive support for hypertext in LaTeX’ (1995). CTAN: hyperref. GitHub: latex3/hyperref.

[11] *LaTeX3 Project*. ‘The xparse package: A generic document command parser’ (1999). CTAN: xparse.

[12] K. Hosny, W. Robertson, P. Stephani, and J. Wright. ‘The unicode-math package: Unicode mathematics support forXeTeX and LuaTeX’ (2006). CTAN: unicode-math.

[13] *American Mathematical Society*. ‘The amssymb package’ (1995). CTAN: amsfonts.

[14] *LaTeX3 Project*. ‘The xfrac package: Split-level fractions’ (2004). CTAN: xfrac.

[15] P. Lehman and J. Wright. ‘The etoolbox package: e-TeX tools for LaTeX’ (2007). CTAN: etoolbox.

Change History

v1.0

General: First release on CTAN: \mathdef, overlines, accents, operators, fractions, units, differentials, delimiters, probabilities, brackets, bra-ket notation, matrix elements, and vectors. 1

v1.1

General: Added optional defaults to \mathdef, \widebar, centred colons, and separate bra-ket spacing, loaded amssymb, and improved flat fractions and slashed symbols. 1

v1.2

General: Added \imaginaryunit, \unit and \unitfrac in math mode, optional integral placement, \floor and \ceil, and refined the spacing of differentials and flat fractions. 1

v1.3

General: Switched underlining from soulutf8 to soul, stopped redefining \int, added \slashedpartialdifferential and \spd, and defined \row and \column with \mathdef. 1

v1.4

General: Applied starred command definitions consistently. 1

v1.5

General: Raised the LaTeX format requirement and expanded the documentation, README, and examples. 1

v1.6

General: Removed the patch of nfssext-cfr, which ran before nfssext-cfr defined its macros. The hep-font package now keeps nfssext-cfr from overwriting \set. Fixed \mathdef macros, \trans, and \slashed in bookmarks and moving arguments, added PDF-string versions of operators, units, inverses, and fractions, gave all trigonometric functions the height of \sin, removed spurious spaces from \unitfrac, avoided the clash of amssymb with unicode-math when loaded after hep-math-font, and let the derivatives follow the math style, with \tfrac for the second star. 1