

The `moremath` package*

Marcel Ilg[†]

Released 2024-07-15

Abstract

The `moremath` package provides several document level commands to ease the typesetting and \LaTeX code readability of certain mathematical constructs. It provides complementary commands to all operators defined by `amsmath`, the commands typeset delimiters that may be automatically, manually or not scaled at all. The commands also accept optional sub- and superscripts to the operators.

Furthermore it provides several commands to typeset gradient, divergence, curl, and Laplace operators, for which there are also versions with delimiters. Those commands also accept an optional subscript and their appearance can be modified using key-value options.

Additionally commands are provided for producing row and column vectors, as well as (anti-)diagonal matrices, utilizing `mathtools` `matrix*` family of environments.

Most of the document level commands defined by this package can also be disabled using a package load-time option to avoid clashes with commands defined by other packages.

Note: This Package is Still in its Initial Development Phase

Do not expect a stable interface until version 1.0.0 has been reached! While I try to keep the interface stable and backwards compatible, I am unable to promise this.

Contents

I	Document Author Documentation	4
1	Introduction	4
2	Dependencies	4
3	Package Options	5
3.1	Package Load Time Options	5
3.2	General Options	5
3.2.1	Options Affecting Vector Calculus Operators	5
3.2.2	Options Affecting Matrices and Vectors	6

*This document corresponds to `moremath` v0.4.0, dated 2024-07-15.

[†]mister01x (at) web (dot) de

4	Delimited Operators	6
4.1	Delimited Operators Predefined by <code>moremath</code>	6
4.2	Declaring New Delimited Operators	8
5	Vector Calculus Operators	9
5.1	Gradient Operator Commands	9
5.1.1	Standalone Operator Command	9
5.1.2	Operators with Delimiters	10
5.2	Divergence Operator Commands	11
5.2.1	Standalone Operator Command	11
5.2.2	Operators with Delimiters	12
5.3	Curl Operator Commands	12
5.3.1	Standalone Operator Command	12
5.3.2	Operators with Delimiters	13
5.4	Laplace Operator Commands	13
5.4.1	Standalone Operator Command	14
5.4.2	Operators with Delimiters	14
5.5	Commands Producing a d'Alembert operator	15
5.5.1	Standalone Operator Command	15
5.5.2	Operators with Delimiters	15
6	Row- and Column Vectors	15
6.1	Small Versions for Inline Math	17
7	Shorthands for Simple Matrices	17
7.1	(Anti-)diagonal Matrices	18
7.1.1	Small Versions for Inline Math	19
7.2	Identity Matrices	20
8	Shorthands for Absolute Value and Norm	22
9	Vertically Centering Math Along the Math Axis	22
II	Documentation for Class and Package Authors	23
1	Setting of Options	23
2	Delimited Operators	23
3	Vector Calculus Typesetting	24
3.1	Functions Producing Standalone Operators with Subscripts	24
3.2	Functions Producing Operators with Delimiters	24
4	Formatting Matrices and Vectors	25
5	Vertically Centering Math Along the Math Axis	26
III	Implementation	26
1	Initial Setup	26

2	Key-value interfaces	26
2.1	Package load time options	26
2.2	Keys controlling appearance	28
2.3	Functions for Setting Options	30
3	Package Initialization	31
4	Centering Math Symbols Along the Math-Axis	31
5	Declaring Paired Delimiters for Internal Use	33
6	Delimited Operators	34
6.1	Document Level Commands	36
7	Vector Calculus Macros	41
7.1	Macros Providing Symbols of Operators	41
7.2	Macros Producing an Operator	43
7.3	Producing Delimited Vector Calculus Operators	44
7.4	Document Level Commands	47
7.4.1	Standalone Operators	48
7.4.2	Operators with Delimiters	50
8	Macros Producing Matrices and Vectors	50
8.1	Producing Row and Column Vectors	50
8.2	Shorthands for Simple Matrices	52
8.2.1	(Anti-)diagonal matrices	52
8.2.2	Identity Matrices	54
8.3	Document Level Commands	55
8.3.1	Row and Column Vectors	56
8.3.2	(Anti-)diagonal Matrices	59
8.3.3	Identity Matrices	63
9	Shorthand Macros for Absolute Value and Norm	65
	Copyright and License	66
	References	66
	Change History	67

Part I

Document Author Documentation

1 Introduction

When typesetting mathematics in \LaTeX it is very common to often encounter, code patterns such as `\sin \left(\frac{x}{2} \right)`. While this above code works as expected its not especially easy for human readers of this code to immediately understand what that code is going to do.

This package tries to ease readability of such commonly used constructs by providing commands that (hopefully) increase readability, as well as speeding up writing math in \LaTeX . Using `moremath` the above code can be simplified to `\psin*\frac{x}{2}`, which is shorter and better “human readable”. The package also provides the possibility for users to define such delimited operators, on their own.

There are also other things that can be improved when typesetting math. One prominent example is the typesetting of row and column vectors, which require the use of a `matrix` environment. For this case `moremath` provides commands which accept a comma separated list of vector entries, obliterating the need for a `matrix` environment to typeset vectors.

The same goes for (anti-)diagonal matrices, only the elements of the diagonal are of importance here. Therefore the production of those matrices may also be simplified into a single command, which again improves readability of the code compared to a mostly empty `matrix` environment.

Another feature of this package is the definition of several (delimited) vector calculus operators such as gradient, divergence and curl operators, this does not only improve the semantics of the math code (`\mathop{\nabla} \mathbf{f}` vs. `\grad \mathbf{f}`), but also tries to provide the correct spacing between the operator and its arguments.

The `moremath` package is written using \LaTeX3 and provides a small \LaTeX3 interface for class and package writers.

The source code for `moremath` is hosted on GitHub at

<https://github.com/Mister00X/moremath>

If you have any issues or found a bug, feel free to register an issue there.

2 Dependencies

The main dependency of this package is `mathtools` [Hög+24]. Optionally the `bm` [CMT23] may be loaded.

If the option `no-vector` has *not* been given as a package load time option, this package also loads the `amssymb` [The] package.

3 Package Options

3.1 Package Load Time Options

The options described in this section *must* be given at package load time i.e. as package options. All options which are not described below will be passed to `mathtools` [Høg+24], see its documentation for more information.

`bm` The `bm` option loads the `bm` [CMT23] package which provides a better version of the `\boldsymbol` command.

`no-vector` The options `no-vector`, `no-abs-shorthands`, `no-operators`, `no-crvector`, and `no-abs-shorthands` `matrix` disable the definition of the predefined commands described in sections 5, 8, 4, `no-operators` 6, and 7 respectively.

`no-crvector` The option `nopredef` achieves the same as the three `no-⟨functionality⟩` options `no-matrix` above. It accepts multiple values, valid option values are: `vector`, `abs`, `operators`, `no-matrix` `crvector`, `matrix`, and `all`. The values `abs`, `operators`, and `vector` disable the predefined document level commands for delimited operators, vector calculus and the shorthands for absolute value and norm respectively. The values `crvector` and `matrix` disable the shorthand commands for row and column vectors, and simple matrices respectively. The special value `all` disables all of them.

The option accepts multiple values which can be given as a comma separated list, or as multiple key-value options, like in the examples below:

```
\usepackage[nopredef={vector,abs,operators}]{moremath}
```

This is equivalent to

```
\usepackage[nopredef=all]{moremath}
```

and to:

```
\usepackage[nopredef=vector,nopredef=abs,nopredef=operators]{moremath}
```

The command `\NewDelimitedOperator` is not affected by any of the above settings.

3.2 General Options

The options described in this section *must not* be given as package options, instead they should be set using `\moremathsetup` or given as optional argument to the commands described later.

```
\moremathsetup {kv list}
```

Updated: 2024-07-15 Sets the options specified in the `⟨key-value list⟩`, the assignment is local to the current group. If a `⟨value⟩` contains a comma it needs to be wrapped in braces. This command may be used anywhere in the document after `moremath` has been loaded.

3.2.1 Options Affecting Vector Calculus Operators

`nabla` The option `nabla` sets the symbol to use by the document level commands described in section 5 to use for the nabla. It accepts a list of `⟨tokens⟩`. Its default value is `\nabla`.
`arrownabla` The option `arrownabla` puts a small arrow over the gradient operator symbol. Its default value is `false`.
`boldnabla` The option `boldnabla` makes the nabla symbol bold. If the `bm` package option has

been given the `\boldsymbol` command from the `bm` package is used for the bold symbol, otherwise the `amsmath` [The23] version is used.

- `grad-op` The option `grad-op` may be used to overwrite, the built in version of the gradient operator, it accepts a `<token list>`. Use at your own responsibility.
- `laplacian-symb` The option `laplacian-symb` sets the symbol to use by the document level commands described in section 5 to use for the Laplace operator. It accepts a list of `<tokens>`.
- `delta-laplace` The option `delta-laplace` replaces the Laplace operator symbol (by default ∇^2) with a uppercase delta (Δ). Its default value is `false`.
- `arrowlaplace` The option `arrowlaplace` if set to `true` makes the Laplace operator look like this: $\vec{\nabla}^2$.
- `laplacian` Like the option `grad-op` above the option `laplacian` may be used to overwrite the built-in version of the Laplace operator. Use at your own responsibility.
- `dalembert-symb` Like the option `nabla` this sets the symbol to use by the document level commands described in section 5.5 to use as symbol for the d'Alembert operator. It accepts a list of `<tokens>`. Its default value is `\square`.
- `vcenter` The option `vcenter` controls if certain mathematical symbols of the operators described in section 5 should be vertically centered along the math-axis. The default value of this option is `true`.

3.2.2 Options Affecting Matrices and Vectors

The options in this section only affect the commands described in sections 6 and 7. To set them with `\moremathsetup` it is necessary to add the prefix `matrix /` to these options, so that the resulting command looks like `\moremathsetup{matrix / <option>}`. When using these options inside the optional argument of the commands described in sections 6 and 7, the prefix `matrix /` must be omitted.

- `delimiter` The option `delimiter` determines the delimiters used for the matrices, valid values are `p` for parenthesis, `b` for brackets, `B` for braces, `v` for single vertical lines (“|”), `V` for double vertical lines (“||”) or empty for no delimiters. The default value is `{}` (empty).
- `fill` The fill option determines the values an (anti-)diagonal matrix is filled with, outside the diagonal. The default is again empty.
- `align` This option determines the alignment of the numbers inside the matrix. The value of this option gets passed to the optional argument of the `matrix*` or `smallmatrix*` family of environments defined by `mathtools` [Høg+24]. Valid values for both types of those environments are `l` for left alignment, `r` for right alignment and `c` for centered alignment. The default is `c`.

TeXhackers note: The non-`small` versions of the commands described in the sections 6 and 7, accept “[...] any column type valid in the usual `array` environment.” [Høg+24]

4 Delimited Operators

4.1 Delimited Operators Predefined by `moremath`

- `no-operators` If the package load time option `no-operators` is not given this package defines several delimited mathematical operators.

Table 1: Operator commands defined by the `amsmath` [The23] package.

<code>\arccos</code>	<code>arccos</code>	<code>\deg</code>	<code>deg</code>	<code>\lg</code>	<code>lg</code>	<code>\projlim</code>	<code>projlim</code>
<code>\arcsin</code>	<code>arcsin</code>	<code>\det</code>	<code>det</code>	<code>\lim</code>	<code>lim</code>	<code>\sec</code>	<code>sec</code>
<code>\arctan</code>	<code>arctan</code>	<code>\dim</code>	<code>dim</code>	<code>\liminf</code>	<code>lim inf</code>	<code>\sin</code>	<code>sin</code>
<code>\arg</code>	<code>arg</code>	<code>\exp</code>	<code>exp</code>	<code>\limsup</code>	<code>lim sup</code>	<code>\sinh</code>	<code>sinh</code>
<code>\cos</code>	<code>cos</code>	<code>\gcd</code>	<code>gcd</code>	<code>\ln</code>	<code>ln</code>	<code>\sup</code>	<code>sup</code>
<code>\cosh</code>	<code>cosh</code>	<code>\hom</code>	<code>hom</code>	<code>\log</code>	<code>log</code>	<code>\tan</code>	<code>tan</code>
<code>\cot</code>	<code>cot</code>	<code>\inf</code>	<code>inf</code>	<code>\max</code>	<code>max</code>	<code>\tanh</code>	<code>tanh</code>
<code>\coth</code>	<code>coth</code>	<code>\injlim</code>	<code>injlim</code>	<code>\min</code>	<code>min</code>		
<code>\csc</code>	<code>csc</code>	<code>\ker</code>	<code>ker</code>	<code>\Pr</code>	<code>Pr</code>		
		<code>\varinjlim</code>	\varinjlim	<code>\varliminf</code>	\varliminf	<code>\lim</code>	\lim
		<code>\varprojlim</code>	\varprojlim	<code>\varlimsup</code>	\varlimsup	<code>\lim</code>	\lim

```

\parccos \parccos [size cmd] {contents}
\barccos \parccos [size cmd] ^{superscript} {contents}
\Barccos \parccos [size cmd] _{subscript} {contents}
\varccos \parccos [size cmd] ^{superscript}_ {subscript} {contents}
\Varccos \parccos* {contents}
\parccos* ^{superscript} {contents}
\parccos* _{subscript} {contents}
\parccos* ^{superscript}_ {subscript} {contents}
\langle prefix \rangle op name [size cmd] {contents}
\langle prefix \rangle op name [size cmd] ^{superscript} {contents}
\langle prefix \rangle op name [size cmd] _{subscript} {contents}
\langle prefix \rangle op name [size cmd] ^{superscript}_ {subscript} {contents}
\langle prefix \rangle op name* {contents}
\langle prefix \rangle op name* ^{superscript} {contents}
\langle prefix \rangle op name* _{subscript} {contents}
\langle prefix \rangle op name* ^{superscript}_ {subscript} {contents}

```

For all of the operators predefined by `amsmath` [The23], which are shown in table 1, `moremath` declares delimited versions. The name of those commands follows the scheme `\langle prefix \rangle \langle op \rangle`, where `\langle prefix \rangle` is one of `p`, `b`, `B`, `v`, or `V` and `\langle op \rangle` is the name of one of the operators shown in table 1.

The `\langle prefix \rangle`s `p`, `b`, `B`, `v`, and `V` stand for parenthesis, brackets, braces, single vertical lines (“|”), and double vertical lines (“||”) respectively.

The commands accept a `\langle size cmd \rangle` optional argument, which is usually one of `\big`, `\Big`, `\bigg` and `\Bigg`. These `\langle size cmd \rangle`s are used to change the size of the delimiters.

The commands also accept sub- and superscripts, which have to be issued *after* the optional argument (if present), but before the mandatory argument `\langle contents \rangle`.

The starred variant uses automatic scaling for the delimiters depending on the height of its contents.

Examples The following examples showcase the use of those predefined delimited operators:

1. Different delimited operators without any scaling:

`\l`

```

\pcos{x} \times \bcos{y}
\times \Bcos{z} \times \vcos{a}
\times \Vcos{b}
\]

```

$$\cos(x) \times \cos[y] \times \cos\{z\} \times \cos|a| \times \cos\|b\|$$

2. Delimited operator with automatic scaling:

```

\[
\pcos*\{\frac{x^2}{2}\}
\]

```

$$\cos\left(\frac{x^2}{2}\right)$$

3. Delimited operator with manual scaling:

```

\[
\pcos[\Big]{\frac{x^2}{2}}
\]

```

$$\cos\left(\frac{x^2}{2}\right)$$

4. Delimited operator with subscript:

```

\[
\plog_{10}\{1+x\}
\]

```

$$\log_{10}(1+x)$$

5. Delimited operator with superscript:

```

\[
\pcos^2{x}
\]

```

$$\cos^2(x)$$

6. Delimited operator with both sub- and superscript and manual scaling:

```

\[
\pcos[\Big]^2_x\{\frac{x}{2}\}
\]

```

$$\cos_x^2\left(\frac{x}{2}\right)$$

4.2 Declaring New Delimited Operators

```

\DeclareDelimitedOperator \DeclareDelimitedOperator{<new op>}{<op>}{<delim>}

```

Creates a new delimited operator, the name of the new command will be `<new op>`. The `<op>` is the command of the operator to use, which is usually a command declared with `\DeclareMathOperator`. `<delim>` is the command to use as paired delimiter, it is expected to behave like a paired delimiter declared by `mathtools` [Høg+24] `\DeclarePairedDelimiter`.

Example: Creating a New Delimited Operator The following code creates a new operator and a paired delimiter and uses it afterwards to declare a paired operator.

```

\documentclass{scrartcl}

\DeclareMathOperator{\glorb}{glorb}
\DeclarePairedDelimiter{\inparen}{\lparen}{\rparen}
\DeclarePairedOperator{\pglorb}{\glorb}{\inparen}

\begin{document}

```



```
\[
  \pglorb{a}
\]
\end{document}
```

The result then looks like this:

$$\text{glorb}(a)$$

5 Vector Calculus Operators

no-vector The commands in this section are only declared if the option `no-vector` has not been given to the package as a load time option.

vcenter The option `vcenter` controls if the symbols for the operators described below should be vertically centered along the math axis. Its default value is `true`.

This option only shows its effects if other options like `arrownabla`, `arrowlaplace`, or `boldnabla` are set to `true`. Like in the example below:

```
\begin{gather*}
  \grad f(x) \quad \grad[arrownabla,vcenter=false] f(x)
  \quad \grad[arrownabla,vcenter=true] f(x) \\
  \laplacian f(x) \quad \laplacian[arrowlaplace,vcenter=false] f(x)
  \quad \laplacian[arrowlaplace,vcenter=true] f(x)
\end{gather*}
```

$$\begin{array}{ccc} \nabla f(x) & \vec{\nabla} f(x) & \vec{\nabla} f(x) \\ \nabla^2 f(x) & \vec{\nabla}^2 f(x) & \vec{\nabla}^2 f(x) \end{array}$$

All of the commands described in this section take key-value options as optional argument, which are described in section 3.2.1

5.1 Gradient Operator Commands

5.1.1 Standalone Operator Command

<code>\grad</code>	<code>\grad [<i>kv opts</i>]</code>
<code>\grad [<i>kv opts</i>] _{<i>subscript</i>}</code>	

Updated: 2024-07-08

The `\grad` command produces a gradient operator looking like this “ ∇ ” by default. The optional argument `kv opts` accepts the key-value options described in section 3.2.1, whitespace between the command name and [`kv opts`] is *not allowed*. An optional subscript using `_` may be given after the optional argument.

Examples of Use

1. Standalone gradient operator (with and without subscript):

```
\[
  \grad f(x), \quad \grad_{x} f(x) \quad \quad \quad \nabla f(x), \quad \nabla_x f(x)
\]
```

2. Bold version of the gradient operator:

```
\[
  \grad[boldnabla] f(x)            $\nabla f(x)$ 
\]
```

3. Gradient operator with an arrow:

```
\[
  \grad[arrownabla] f(x)          $\vec{\nabla} f(x)$ 
\]
```

5.1.2 Operators with Delimiters

```
\pgrad \langle delim \rangle grad [size cmd] {content}
\bgrad \langle delim \rangle grad [kv opts] {content}
\Bgrad \langle delim \rangle grad [size cmd] _{subscript} {content}
\vgrad \langle delim \rangle grad [kv opts] _{subscript} {content}
\Vgrad \langle delim \rangle grad* [kv opts] {content}
\biggrad \langle delim \rangle grad* [kv opts] _{subscript} {content}
```

The `\langle delim \rangle grad` family of commands produces gradient operator which is followed by `\langle contents \rangle` inside delimiters. The delimiter is determined by the first letter of the command `\langle delim \rangle`, which may be `p` for parenthesis, `b` for brackets, `B` for braces, `v` for a single vertical line (“|”), or `V` for a double vertical line (“||”).

The commands accept either a `\langle size command \rangle` as optional argument, which gets passed to `mathtools` [Hög+24] paired delimiter or a list of `\langle key-value option \rangle`s, the `\langle kv opts \rangle` *must* be given using the complete syntax, i.e. `\langle key \rangle = \langle value \rangle`, shorthands for options with an implicit default value (`arrownabla`), will not work here. The `\langle size command \rangle` is usually one of `\big`, `\Big`, `\bigg` and `\Bigg`. Valid `\langle kv opts \rangle` are all options described in section 3.2.1 and the key `scale` which accepts a `\langle size cmd \rangle`.

Note:

Do not mix `\langle kv opts \rangle` and `\langle size cmd \rangle`, use either `\pgrad[\big]{f(x)}` or `\pgrad[arrownabla=true,scale=\big]{f(x)}`.

An optional `\langle subscript \rangle` may be given between the optional argument, and `\langle contents \rangle`. One use case for this subscript is to write formulae using the so called Feynman-notation, where the gradient operator acts only on one variable.

The starred version of the commands automatically scale the delimiters with its contents.

Examples:

1. Gradient operator with non-scaled delimiters:

```
\[
  \pgrad{1+\vec{x}}                $\nabla(1 + \vec{x})$ 
\]
```

2. Bold version:

```
\[
  \pgrad[boldnabla=true]{1+\vec{x}}  $\nabla(1 + \vec{x})$ 
\]
```

3. Gradient operator with automatically scaled delimiters:

$$\begin{array}{l} \backslash[\\ \quad \backslash\text{pgrad}*\{\frac{1}{x}\} \\ \backslash] \end{array} \quad \nabla\left(\frac{1}{x}\right)$$

4. Manually scaled version:

$$\begin{array}{l} \backslash\text{begin}\{\text{gather}*\} \\ \quad \backslash\text{pgrad}[\backslash\text{Big}]\{\frac{1}{x}\}\backslash\backslash[.5ex] \\ \quad \backslash\text{pgrad}[\text{scale}=\backslash\text{Big}]\{\frac{1}{x}\} \\ \backslash\text{end}\{\text{gather}*\} \end{array} \quad \begin{array}{l} \nabla\left(\frac{1}{x}\right) \\ \\ \nabla\left(\frac{1}{x}\right) \end{array}$$

5. Feynman-Notation:

$$\begin{array}{l} \backslash[\\ \quad \backslash\text{pgrad}_{\{x\}}\{x+y+z\} \\ \backslash] \end{array} \quad \nabla_x(x+y+z)$$

5.2 Divergence Operator Commands

5.2.1 Standalone Operator Command

<code>\divergence</code>	<code>\divergence [⟨kv opts⟩]</code>
	<code>\divergence [⟨kv opts⟩] _{\langle subscript \rangle}</code>

Updated: 2024-07-08

The `\divergence` command produces the divergence operator “ $\nabla\cdot$ ”, its usage is analogous to the use of the `\grad` command, which is described in section 5.1.1.

Examples

1. Standalone divergence operator

$$\begin{array}{l} \backslash[\\ \quad \backslash\text{divergence } f(x) \\ \backslash] \end{array} \quad \nabla \cdot f$$

2. Standalone divergence operator with an arrow over the gradient operator

$$\begin{array}{l} \backslash[\\ \quad \backslash\text{divergence}[\text{arrownabla}] f(x) \\ \backslash] \end{array} \quad \vec{\nabla} \cdot f(x)$$

3. Standalone divergence operator with subscript

$$\begin{array}{l} \backslash[\\ \quad \backslash\text{divergence}_{\{x\}} f(x) \\ \backslash] \end{array} \quad \nabla_x \cdot f(x)$$

5.2.2 Operators with Delimiters

```
\pdiv \langle delim \rangle div [size cmd] {content}
\bdiv \langle delim \rangle div [kv opts] {content}
\Bdiv \langle delim \rangle div [size cmd] _{subscript} {content}
\vdiv \langle delim \rangle div [kv opts] _{subscript} {content}
\Vdiv \langle delim \rangle div* [kv opts] {content}
_____ \langle delim \rangle div* [kv opts] _{subscript} {content}
```

The `\langle delim \rangle div` family of commands produces the divergence operator with its arguments placed inside delimiters. The usage of these commands is analogous to the `\langle delim \rangle grad` family of commands described in section 5.1.2.

Examples

1. Divergence operator with parenthesis and no scaling

```
\[
  \pdiv{1+x}
\]


$\nabla \cdot (1 + x)$


```

2. Bold version with manual scaling and subscript

```
\[
  \pdiv[boldnabla=true,scale=\Big]_x{1 + \frac{1}{x}}
\]


$\nabla_x \cdot \left(1 + \frac{1}{x}\right)$


```

3. Automatic scaling

```
\[
  \pdiv*{1 + \frac{1}{x}}
\]


$\nabla \cdot \left(1 + \frac{1}{x}\right)$


```

5.3 Curl Operator Commands

5.3.1 Standalone Operator Command

```
\curl [kv opts]
_____ \curl [kv opts] _{subscript}
```

Updated: 2024-07-08

The `\curl` command produces the curl operator “ $\nabla \times$ ”, its usage is analogous to the use of the `\grad` command described in section 5.1.1.

Examples

1. Standalone curl operator

```
\[
  \curl f(x)
\]


$\nabla \times f(x)$


```

2. Standalone curl operator with an arrow over the gradient operator

```
\[
\curl[arrownabla] f(x)
\]
```

$$\vec{\nabla} \times f(x)$$

3. Standalone curl operator with subscript

```
\[
\curl_{x} f(x,y)
\]
```

$$\nabla_x \times f(x,y)$$

5.3.2 Operators with Delimiters

```
\pcurly \langle delim \rangle curl [size cmd] {content}
\bcurl \langle delim \rangle curl [kv opts] {content}
\Bcurly \langle delim \rangle curl [size cmd] _{subscript} {content}
\vcurl \langle delim \rangle curl [kv opts] _{subscript} {content}
\Vcurly \langle delim \rangle curl* [kv opts] {content}
\vcurl* \langle delim \rangle curl* [kv opts] _{subscript} {content}
```

The `\langle delim \rangle curl` family of commands produce the curl operator with its arguments placed inside delimiters. The usage of these commands is analogous to the `\langle delim \rangle grad` family of commands described in section 5.1.2.

Examples

1. Curl operator with parenthesis without scaling

```
\[
\pcurly{1+x}
\]
```

$$\nabla \times (1 + x)$$

2. Bold version with manual scaling and subscript

```
\[
\pcurly[boldnabla=true,scale=\Big]_{x}{1 + \frac{1}{x}}
\]
```

$$\nabla_x \times \left(1 + \frac{1}{x}\right)$$

3. Automatic scaling

```
\[
\pcurly*{1 + \frac{1}{x}}
\]
```

$$\nabla \times \left(1 + \frac{1}{x}\right)$$

5.4 Laplace Operator Commands

This section describes commands which can be used to typeset a Laplace operator.

Like the commands described in sections 5.1, 5.2, and 5.3 the commands in this section accept key-value options via an optional argument. There is some deviation from the options compared to the above commands: The `arrownabla` option is ignored, instead the `arrowlaplace` option produces an arrow over the operator. The `boldnabla` option on the other hand is not ignored. Finally the `delta-laplace` option replaces the symbol used for the operator from ∇^2 to Δ

5.4.1 Standalone Operator Command

```
\laplacian      \laplacian [(kv opts)]
Updated: 2024-07-08 \laplacian [(kv opts)] _{subscript}
```

The `\laplacian` command produces a Laplace operator, which looks by default like this: ∇^2 .

Its interface is analogous to the `\grad`, `\divergence`, and `\curl` commands described above, with the difference in key-value options described at the start of this subsection.

5.4.2 Operators with Delimiters

```
\plaplacian \{delim\}laplacian [(size cmd)] {content}
\blaplacian \{delim\}laplacian [(kv opts)] {content}
\Blaplacian \{delim\}laplacian [(size cmd)] _{subscript} {content}
\vlaplacian \{delim\}laplacian [(kv opts)] _{subscript} {content}
\Wlaplacian \{delim\}laplacian* [(kv opts)] {content}
\Wlaplacian \{delim\}laplacian* [(kv opts)] _{subscript} {content}
```

Examples

1. Laplace operator delimited by parenthesis without scaling

```
\[
  \plaplacian{1+x}
\]

$$\nabla^2(1+x)$$

```

2. Version with arrow, manual scaling and subscript

```
\[
  \plaplacian[arrowlaplace=true,scale=\Big]_x{1 + \frac{1}{x}}
\]

$$\vec{\nabla}_x^2\left(1 + \frac{1}{x}\right)$$

```

3. Version with automatic scaling

```
\[
  \plaplacian*{1 + \frac{1}{x}}
\]

$$\nabla^2\left(1 + \frac{1}{x}\right)$$

```

4. Using a delta as symbol for the Laplacian

```
\[
  \plaplacian[delta-laplace=true]{1+x}
\]

$$\Delta(1+x)$$

```

5.5 Commands Producing a d’Alembert operator

5.5.1 Standalone Operator Command

<code>\quabla</code>	<code>\quabla</code> [<i>kv opts</i>]
	<code>\quabla</code> [<i>kv opts</i>] <i>_{subscript}</i>
<small>New: 2024-07-04</small>	
<small>Updated: 2024-07-08</small>	The <code>\quabla</code> command produces the d’Alembert operator “ \square ”. This command accepts an optional subscript.

The command is called `\quabla` because that’s shorter and easier to type than `\dalembertian`. If you want to have a command called `\dalembertian`, put the following in your documents preamble.

```
\NewCommandCopy\quabla\dalembertian
```

Example of Use:

```
\[
  \quabla f(x)
\]            $\square f(x)$ 
```

5.5.2 Operators with Delimiters

<code>\pquabla</code>	<code>\<delim>quabla</code> [<i>size cmd</i>] <i>{contents}</i>
<code>\bquabla</code>	<code>\<delim>quabla</code> [<i>kv opts</i>] <i>{contents}</i>
<code>\Bquabla</code>	<code>\<delim>quabla</code> [<i>size cmd</i>] <i>_{subscript}</i> <i>{contents}</i>
<code>\vquabla</code>	<code>\<delim>quabla</code> [<i>kv opts</i>] <i>_{subscript}</i> <i>{contents}</i>
<code>\Vquabla</code>	<code>\<delim>quabla*</code> [<i>kv opts</i>] <i>{contents}</i>
<small>New: 2024-07-04</small>	<code>\<delim>quabla*</code> [<i>kv opts</i>] <i>_{subscript}</i> <i>{contents}</i>

The `\<delim>quabla` family of commands produce a d’Alembert operator with *<contents>* placed inside delimiters. Their usage is analogous to the `\<delim>grad` family of commands described in section [5.1.2](#).

6 Row- and Column Vectors

`no-crvector` The command in this section are only declared if the option `no-crvector` has not been given as a package option.

Valid keys are `delimiter`, `fill`, and `align`, their usage is described in section [3.2.2](#).

```

\cvector \cvector [kv opts] {clist}
\rvector \rvector [kv opts] {clist}

```

The commands `\cvector` and `\rvector` produce row and column vectors respectively. Both of them accept key-value options as optional argument `<kv opts>`. Valid keys and values are described in section 3.2.2.

The mandatory argument `<clist>` is a comma-separated-list, whose elements are the entries of the column/row vector. If a comma has to appear inside an entry the entire entry has to be wrapped in braces.

The delimiter of the row or column vectors depends on the current value of the option `delimiter`, by default empty. The option `fill` has no effect on the commands and is simply ignored.

TeXhackers note: If you were to define your own `matrix*`-like environment called `mymatrix*`, which has an interface compatible to `mathtools`'s `matrix*` family of environments, you could make use of it by setting the value of `delimiter` to `my`.

Examples:

1. Column “vector” without delimiters:

```

\[
  \cvector{a_{1},a_{2},a_{3}}
\]

```

$$\begin{matrix} a_1 \\ a_2 \\ a_3 \end{matrix}$$

2. A column vectors delimited with parenthesis and different alignment:

```

\[
  \cvector[delimiter=p,align=c]{-1,2,3}
\]
\[
  \cvector[delimiter=p,align=r]{-1,2,3}
\]
\[
  \cvector[delimiter=p,align=l]{-1,2,3}
\]

```

$$\begin{matrix} \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} \\ \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} \\ \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} \end{matrix}$$

Row- and Column Vectors with Predefined Delimiters As vectors are commonly delimited by parenthesis, brackets, braces, etc. several shorthands producing delimited vectors are also available.

```

\pcvector \<delim>cvector [kv opts] {clist}
\bcvector \<delim>rvector [kv opts] {clist}

```

The `\<delim><c or r>vector` family of commands, accepts a list of key-value options as optional argument `<kv opts>`. Valid keys are described in section 3.2.2.

The mandatory argument `<clist>` is a comma-separated-list of the entries of the vector. If a comma needs to appear inside an entry of the vector, that entry has to be wrapped in braces.

`<delim>` may have the value `p` for parenthesis, `b` for brackets, `B` for braces, `v` for a single vertical line, or `V` for a double vertical line.

Example

```
\[
  \pcvector{a_{1},a_{2},a_{3}}
\]
```

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

6.1 Small Versions for Inline Math

As the `matrix` and `matrix*` family of environments is unsuitable, for inline math, `math-tools` [Høg+24] provides the `smallmatrix` and `smallmatrix*` family of environments. This package provides analogous commands for row and column vectors to be typeset in inline math mode.

```
\smallvector \smallvector [kv opts] {clist}
\smallrvector \smallrvector [kv opts] {clist}
```

`\smallvector` and `\smallrvector` produce column and row vectors suitable for inline math. Both commands accept the same optional key-value arguments as `\cvector` and `\rvector`.

Example:

An inline version of `\verb|\cvector|` looks like this `\(\smallvector{1,0}\)`.

An inline version of `\cvector` looks like this $\frac{1}{0}$.

```
\psmallvector \psmallvector [kv opts] {clist}
\bsmallvector \psmallrvector [kv opts] {clist}
\Bsmallvector \(\delim\small<c or r>vector [kv opts] {clist}
```

The `\(\delim\small<c or r>vector` family of commands like `\psmallvector` produce small inline math version of row and column vectors. Their interface is identical to the commands described above.

```
\psmallrvector
\bsmallrvector
\Bsmallrvector
\vsmallrvector
\Vsmallrvector
```

Example:

An inline version of `\verb|\pcvector|` looks like this `\(\psmallvector{1,0}\)`.

An inline version of `\pcvector` looks like this $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

7 Shorthands for Simple Matrices

`no-matrix` The commands in this section are only defined if the option `no-matrix` has not been given to the package at load-time.

7.1 (Anti-)diagonal Matrices

```
\diagmat      \diagmat [<kv opts>] {<diagonal>}
\antidiagmat  \antidiagmat [<kv opts>] {<diagonal>}
```

`\diagmat` and `\antidiagmat` produce a diagonal or anti-diagonal matrix respectively. The optional argument `<kv opts>` accepts the key value options described in section 3.2.2.

The key `fill` determines the contents of the matrix entries which are not part of the (anti-)diagonal, its default value is `{}` i.e. empty. The key `align` determines the alignment of the entries inside the matrix, valid values are usually `l`, `c`, and `r`, the default is `c`. The key `delimiter` determines the delimiter around the matrix, its default value is `{}` (none). Valid values for the delimiters are `p` for parenthesis, `b` for brackets, `B` for braces, `v` for a single vertical line (“|”), and `V` for a double vertical line (“||”). See the `mathtools` manual [Høg+24] for more information.

T_EXhackers note: The value of `delimiter` gets inserted inside the `\begin{#1matrix*}` and `\end{#1matrix*}` commands. Therefore it would be possible to define your own `matrix*` like environment, called for example `mymatrix*` and set `delimiter=my` to make use of it.

The mandatory argument `<diagonal>` has to be a comma separated list of the entries of the (anti-)diagonal. If an entry of the diagonal needs to contain a comma `,`, the entire entry has to be wrapped in braces.

Examples:

1. A diagonal and an anti-diagonal matrix without delimiters:

```
\begin{gather*}
\diagmat{1,2,3}\ [.5ex]
\antidiagmat{1,2,3}
\end{gather*}
```

$$\begin{matrix} & & & 1 & & & \\ & & & & 2 & & \\ & & & & & 3 & \\ & & & & & & 1 \\ & & & & & & & 2 \\ & & & & & & & & 3 \end{matrix}$$

2. A diagonal matrix delimited by parenthesis filled with zeros:

```
\[
\diagmat[delimiter=p,fill=0]{1,2,3}
\]
```

$$\left(\begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{matrix} \right)$$

3. A diagonal matrix filled with a symbol:

```
\[
\diagmat[fill=\square]{1,2,3}
\]
```

$$\begin{matrix} 1 & \square & \square \\ \square & 2 & \square \\ \square & \square & 3 \end{matrix}$$

4. Diagonal matrices with different alignment:

```
\[
\diagmat[delimiter=p,align=c,fill=0]{-1,-2,-3} \quad
\diagmat[delimiter=p,align=l,fill=0]{-1,-2,-3} \quad
\diagmat[delimiter=p,align=r,fill=0]{-1,-2,-3}
\]
```

$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -3 \end{pmatrix} \quad \begin{pmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -3 \end{pmatrix} \quad \begin{pmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -3 \end{pmatrix}$$

Shorthands for (Anti-)diagonal Matrices with Delimiters As matrices are more often than not written inside of delimiters, `moremath` provides several shorthands for producing those matrices without having to set the `delimiter` key explicitly.

```

\pdiagmat \pdiagmat [<kv opts>] {<diagonal>}
\bdiagmat \bdiagmat [<kv opts>] {<diagonal>}
\Bdiagmat \Bdiagmat [<kv opts>] {<diagonal>}
\vdiagmat \vdiagmat [<kv opts>] {<diagonal>}
\Vdiagmat \Vdiagmat [<kv opts>] {<diagonal>}

```

The `\(delim)diagmat` family of commands provides shorthands for producing a delimited (anti-)diagonal matrix without having to set the `delimiter` key explicitly every time. The pre-set `delimiter` may be overwritten by explicitly passing `delimiter` as a key-value option.

The optional argument `<kv opts>` accepts the same key-value arguments as `\diagmat`.

Example:

$$\begin{array}{l} \backslash[\\ \quad \backslashpdiagmat\{1,2,3\} \\ \backslash] \end{array} \quad \begin{pmatrix} 1 & & \\ & 2 & \\ & & 3 \end{pmatrix}$$

```

\pantidiagmat \pantidiagmat [<kv opts>] {<diagonal>}
\bantidiagmat \bantidiagmat [<kv opts>] {<diagonal>}
\Bantidiagmat \Bantidiagmat [<kv opts>] {<diagonal>}
\vantidiagmat \vantidiagmat [<kv opts>] {<diagonal>}
\Vantidiagmat \Vantidiagmat [<kv opts>] {<diagonal>}

```

The `\(delim)antidiagmat` family of commands behaves like the `\(delim)diagmat` commands described above, except they produce anti-diagonal matrices.

Example:

$$\begin{array}{l} \backslash[\\ \quad \backslashpantidiagmat\{1,2,3\} \\ \backslash] \end{array} \quad \begin{pmatrix} & & 1 \\ & 2 & \\ 3 & & \end{pmatrix}$$

7.1.1 Small Versions for Inline Math

`mathtools` defines special matrix environments for use in inline math mode, the `smallmatrix` and `smallmatrix*` family of environments. `moremath` provides for the case of typesetting an (anti-)diagonal matrix several commands which utilize these inline math versions.

```

\smallldiagmat \smallldiagmat [<kv opts>] {<diagonal>}
\smallantidiagmat \smallantidiagmat [<kv opts>] {<diagonal>}

```

The `\smallldiagmat` and `\smallantidiagmat` commands behave like their non-`small` counterpart described above, see their description for more information.

Example:

An inline version of a diagonal matrix looks like this
`\(\smallldiagmat{1,1}\)`.

An inline version of a diagonal matrix looks like this 1_1 .

<code>\psmallldiagmat</code>	<code>\psmallldiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\bsmallldiagmat</code>	<code>\bsmallldiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\Bsmallldiagmat</code>	<code>\Bsmallldiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\vsmallldiagmat</code>	<code>\vsmallldiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\Vsmallldiagmat</code>	<code>\Vsmallldiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>

Like the `\<delim>diagmat` commands described above there are also shorthand commands for producing an inline math version of a diagonal matrix with pre-set delimiters.

Example:

An inline math delimited diagonal matrix looks like this
`\(\psmalldiagmat{1,1}\)`.

An inline math delimited diagonal matrix looks like this $({}^1_1)$.

<code>\psmallantidiagmat</code>	<code>\psmallantidiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\bsmallantidiagmat</code>	<code>\bsmallantidiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\Bsmallantidiagmat</code>	<code>\Bsmallantidiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\vsmallantidiagmat</code>	<code>\vsmallantidiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>
<code>\Vsmallantidiagmat</code>	<code>\Vsmallantidiagmat</code>	<code>[[<kv opts>]</code>	<code>{<diagonal>}</code>

Like the `\<delim>antidiagmat` commands described above there are also shorthand commands for producing inline math versions of anti-diagonal matrices inside of delimiters.

Example:

A delimited version of an inline math anti-diagonal matrix looks like this
`\(\psmallantidiagmat{1,1}\)`.

A delimited version of an inline math anti-diagonal matrix looks like this
 $({}^1_1)$.

7.2 Identity Matrices

As identity matrices are always quadratic, one can further simplify the typesetting of identity matrices to a command, which constructs the identity matrix from the number of dimensions. The commands in this subsection perform this.

```

\idmat      \idmat  [(kv opts)] {(dimension)}
\smallidmat \smallidmat [(kv opts)] {(dimension)}

```

New: 2024-07-04 The `\idmat` command produces an identity matrix. The optional argument `(kv opts)` accepts any valid *matrix* key-value arguments, which are described in section 3.2.2. The mandatory argument `(dimension)` expects a *positive* integer, which specifies the dimensions of the identity matrix.

The `\smallidmat` command behaves like the `\idmat` command, except it produces a matrix suitable for inline math mode.

T_EXhackers note: The maximum number of columns is determined by the T_EX counter `MaxMatrixCols`, which has a default value of 10. See The LaTeX Project Team [The23] for more information.

```

\pidmat      \pidmat  [(kv opts)] {(dimension)}
\bidmat      \bidmat  [(kv opts)] {(dimension)}
\Bidmat      \Bidmat  [(kv opts)] {(dimension)}
\vidmat      \vidmat  [(kv opts)] {(dimension)}
\Vidmat      \Vidmat  [(kv opts)] {(dimension)}

```

New: 2024-07-04 Like the `\(delim)diagmat` commands, the `\(delim)idmat` commands produce an identity matrix with a pre-set delimiter around the matrix, i.e. they behave like `\idmat[delimiter = (delim)]{(dimension)}`. `(delim)` can be `p` for parenthesis, `b` for brackets, `B` for braces, `v` for a single vertical line (“|”), or `V` for a double vertical line (“||”).

Furthermore these commands accept the same `(kv opts)` as the `\idmat` command.

```

\psmallidmat \psmallidmat [(kv opts)] {(dimension)}
\bsmallidmat \bsmallidmat [(kv opts)] {(dimension)}
\Bsmallidmat \Bsmallidmat [(kv opts)] {(dimension)}
\vsmallidmat \vsmallidmat [(kv opts)] {(dimension)}
\Vsmallidmat \Vsmallidmat [(kv opts)] {(dimension)}

```

New: 2024-07-04 These commands are provide inline math suitable versions of the `\(delim)idmat` commands described above. See their description for more information.

The commands accept any of the key-value options described in section 3.2.2 as optional argument `(kv opts)`.

Examples of Use

1. Identity matrix without delimiters:

```

\[
\idmat{3}
\]

```

$$\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}$$

2. Identity matrices with different delimiters:

```

\[
\pidmat{3}\text{,}\quad \bidmat{3}\text{,}\quad
\Bidmat{3}\text{,}\quad \vidmat{3}\text{,}\quad
\Vidmat{3}
\]

```

$$\begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix}, \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}, \left\{ \begin{matrix} 1 & & \\ & 1 & \\ & & 1 \end{matrix} \right\}, \left| \begin{matrix} 1 & & \\ & 1 & \\ & & 1 \end{matrix} \right|, \left\| \begin{matrix} 1 & & \\ & 1 & \\ & & 1 \end{matrix} \right\|$$

3. Inline math versions of identity matrices with delimiters:

Inline math mode matrices look like this:

`\(\psmallidmat{2}\)`, `\(\bsmallidmat{2}\)`, `\(\Bsmallidmat{2}\)`,
`\(\vsmallidmat{2}\)`, `\(\Vsmallidmat{2}\)`.

Inline math mode matrices look like this: $(^1_1)$, $[^1_1]$, $\{^1_1\}$, $|^1_1|$, $\|^1_1\|$.

4. Identity matrix using the `fill` option for the non-diagonal elements

`\[` $\begin{pmatrix} 1 & * & * \\ * & 1 & * \\ * & * & 1 \end{pmatrix}$
`\pidmat[fill=\star]{3}`
`\]`

8 Shorthands for Absolute Value and Norm

`no-abs-shorthands` The commands in this section are only declared if the option `no-abs-shorthands` has not been given to the package as a load time option.

<code>\abs</code>	<code>\abs</code> [<i>size cmd</i>] { <i>content</i> }
<code>\norm</code>	<code>\abs*</code> { <i>content</i> }
	<code>\norm</code> [<i>size cmd</i>] { <i>content</i> }
	<code>\norm*</code> { <i>content</i> }

The commands `\abs` and `\norm`, produce $|\cdot|$ and $\|\cdot\|$ respectively. These commands are simply paired delimiters defined using `mathtools`' `\DeclarePairedDelimiter` command, instruction on their usage can therefore be found in Høgholm et al. [Høgh+24].

9 Vertically Centering Math Along the Math Axis

Sometimes it is useful to explicitly center a math symbol along the math axis. A prominent example is the case of `\mathop{\boldsymbol{\nabla}}\nolimits` vs. `\mathop{\nabla}\nolimits`, as displayed below.

$$\boldsymbol{\nabla} f(x) \quad \nabla f(x)$$

Here the bold nabla is slightly higher up than the non-bold version.

<code>\VCenterMath</code>	<code>\VCenterMath</code> { <i>contents</i> }
---------------------------	---

New: 2024-07-08



The command `\VCenterMath` centers *contents* along the current math axis, while adhering to the current math style. This command is not in a `TeX`-sense `\long`, i.e. it does not take `\par` tokens.

This command is somewhat dangerous as it utilizes the `\vcenter` primitive, which leaves math mode and requires to reenter it. *Use at your own responsibility.*

TeXhackers note: This command is a wrapper around `\vcenter`, `\hbox:n` and `\mathpalette`, it reenters math mode inside the `hbox`.

With `\VCenterMath` it is possible to fix the above example to:

```
\[
  \mathop{\VCenterMath{\boldsymbol{\nabla}}}\nolimits f(x)
  \quad \mathop{\nabla}\nolimits f(x)
\]
```

$$\nabla f(x) \quad \nabla f(x)$$

Part II

Documentation for Class and Package Authors

1 Setting of Options

```
\moremath_setup:n \moremath_setup:n {<kv opts>}
```

Updated: 2024-07-15 This functions sets the key-value options `<kv opts>` in the `moremath` namespace.

2 Delimited Operators

```
\moremath_delim_op_noscale:NNnnn \moremath_delim_op_noscale:NNnnn <math op> <delim>
\moremath_delim_op_autoscale:NNnnn   <sup t1> <sub t1> <t1>
```

```
Updated: 2024-07-15 \moremath_delim_op_autoscale:NNnnn <math op> <delim>
                    <sup t1> <sub t1> <t1>
```

The two functions `\moremath_delim_op_noscale:NNnnn` and `\moremath_delim_op_autoscale:NNnnn` provide a version of a delimited operator with no scaling or automatic scaling respectively. The token list arguments `<sup t1>` and `<sub t1>` take the super- and subscript of the operator respectively. Both of those token lists may be empty.

The first argument to the function is a control sequence `<math op>`, which should be an operator declared with `\DeclareMathOperator`. The second argument `<delim>` has to be a control sequence of a paired delimiter, as defined by `mathtools`'s `\DeclarePairedDelimiter`.

The final argument `<t1>` is a list of arbitrary token to put inside the delimiters.

```
\moremath_delim_op_manuscale:NNnnnn \moremath_delim_op_manuscale:NNnnnn <math op> <delim> <scale cmd>
\moremath_delim_op_manuscale:NNVnnn   <sup> <sub> <t1>
```

Updated: 2024-07-15

The function `\moremath_delim_op_manuscale:NNnnnn` typesets a math operator with manual scaling of its delimiters. Its arguments are identical to `\moremath_delim_op_noscale:NNnnn` except for the argument `<scale cmd>` which has to be a token list containing a `<size cmd>` that can be understood by `mathtools`, usually `\big`, `\Big`, `\bigg` and `\Bigg`.

```
\moremath_new_delim_op_command:NNN \moremath_new_delim_op_command:NNN <new csname> <operator> <delim>
\moremath_new_delim_op_command:cNN
```

Updated: 2024-07-15

The function `\moremath_new_delim_op_command:NNN` defines a new document level command called `<new csname>`, utilizing the operator `<operator>` and the delimiter `<delim>`. `<operator>` has to be a control sequence referring to an operator, as declared by `\DeclareMathOperator`. `<delim>` has to be a paired delimiter as defined by `mathtools'` `\DeclarePairedDelimiter`.

3 Vector Calculus Typesetting

3.1 Functions Producing Standalone Operators with Subscripts

```
\moremath_gradient_operator:n \moremath_gradient_operator:n <{subscript}>
\moremath_divergence_operator:n \moremath_divergence_operator:n <{subscript}>
\moremath_curl_operator:n \moremath_curl_operator:n <{subscript}>
\moremath_laplace_operator:n \moremath_laplace_operator:n <{subscript}>
\moremath_dalembert_operator:n \moremath_dalembert_operator:n <{subscript}>
```

Updated: 2024-07-15

Each of these functions return a vector calculus operator, with an (optional) subscript. The argument `<subscript>`, a token list, may be empty, in which case no subscript is produced.

The actual operator produced depends on the current settings of the key-value options described in section 3 inside the current group.

3.2 Functions Producing Operators with Delimiters

```
\moremath_delim_nabla_op_noscale:NNnn \moremath_delim_nabla_op_noscale:NNnn <operator> <delim>
\moremath_delim_nabla_op_autoscale:NNnn <{sub}> <{contents}>
\moremath_delim_nabla_op_autoscale:NNnn <operator> <delim>
\moremath_delim_nabla_op_autoscale:NNnn <{sub}> <{contents}>
```

Updated: 2024-07-15

The usage of these functions is similar to the functions `\moremath_delim_op_noscale:NNnn` and `\moremath_delim_op_autoscale:NNnn` except for the missing superscript.

The first argument `<operator>` has to be a function which accepts one argument (the subscript). This is usually one of the `\moremath_<op>_operator:n` commands.

The argument `<sub>`, which may be empty, is a token list to pass as the subscript to `\moremath_<op>_operator:n`.

The last argument `<contents>` is a token list to put inside the delimiters.

```
\moremath_delim_nabla_op_manuscale:NNnnn \moremath_delim_nabla_op_manuscale:NNnnn <operator> <delim>
\moremath_delim_nabla_op_manuscale:NNVnn <{scale cmd}> <{sub}> <{contents}>
```

Updated: 2024-07-15

The usage of function is similar to the above functions except for the additional argument `<scale cmd>` which is a token list with a `<size cmd>` that should be understood by `mathtools` (e.g. `\big`, `\Big`, etc.).

4 Formatting Matrices and Vectors

```

\moremath_column_vector:nn      \moremath_column_vector:nn <{delim spec tl}> <{clist}>
\moremath_row_vector:nn        \moremath_row_vector:nn <{delim spec tl}> <{clist}>
\moremath_column_smallvector:nn \moremath_column_smallvector:nn <{delim spec tl}> <{clist}>
\moremath_row_smallvector:nn   \moremath_row_smallvector:nn <{delim spec tl}> <{clist}>

```

Updated: 2024-07-15

These functions produce a column or row vector. The `small<c or r>vector` versions are intended for inline math mode.

The first argument `<delim spec tl>` should be the specification of a delimiter, as used in the naming of the `matrix*` environments by `mathtools`.

The second argument `<clist>`, is a comma-separated list of entries of the vector.

Of this commands the `c` versions produce column vectors, the `r` versions produce row vectors.

The alignment of the entries inside the vector depends on the current value of the key `align`.

```

\moremath_diagonal_matrix:nn      \moremath_diagonal_matrix:nn <{delim tl}> <{clist}>
\moremath_diagonal_matrix:(nV|Vn|VV) \moremath_antidiagonal_matrix:nn <{delim tl}> <{clist}>
\moremath_antidiagonal_matrix:nn  \moremath_diagonal_smallmatrix:nn <{delim tl}> <{clist}>
\moremath_antidiagonal_matrix:(nV|Vn|VV) \moremath_antidiagonal_smallmatrix:nn <{delim tl}>
\moremath_diagonal_smallmatrix:nn  <{clist}>
\moremath_diagonal_smallmatrix:(nV|Vn|VV)
\moremath_antidiagonal_smallmatrix:nn
\moremath_antidiagonal_smallmatrix:(nV|Vn|VV)

```

Updated: 2024-07-15

These functions produce (anti-)diagonal matrices. The argument `<delim tl>` is a token list, intended to specify the delimiter of the matrix or no delimiter if the `<tl>` is empty no delimiters are produced.

The argument `<clist>` is a list of comma separated values which specify the entries of the diagonal. Other properties of the matrix to produce like alignment (`align`) and the tokens to insert for non-diagonal entries (`fill`) depend on the current setting of the keys described in section 3.2.2.

The `smallmatrix` versions are intended for use in inline math mode.

```

\moremath_id_matrix:n           \moremath_id_matrix:n <{dimensions}>
\moremath_id_matrix:V          \moremath_id_smallmatrix:n <{dimensions}>
\moremath_id_smallmatrix:n
\moremath_id_smallmatrix:V

```

New: 2024-07-04

Updated: 2024-07-15

These functions produce a identity matrix with `<dimensions>` rows and columns. `<dimensions>` is expected to be a *positive* integer. The `smallmatrix` version utilizes `mathtools`' `smallmatrix*` environment, which makes it suitable for inline math mode.

The delimiter around the matrix is determined by the current value of the key `matrix / delimiter` inside the current group.

As these functions utilize `\moremath_diagonal_matrix:VV` or `\moremath_diagonal_smallmatrix:VV` and make use of temporary variables it is advised to put them into their own group, i.e. surround them by `\group_begin:` and `\group_end:`.

5 Vertically Centering Math Along the Math Axis

As already said in section 9 sometimes it is needed to explicitly center some math code.

```
\moremath_vcenter:n \moremath_vcenter:n {<contents>}
```

New: 2024-07-08
Updated: 2024-07-15

The function `\moremath_vcenter:n` places `<contents>` inside a `hbox`, which is centered along the math axis by `\vcenter`. The contents inside the `hbox` are set in math mode. The function also applies the “outer” math style to the contents of the `hbox`. This function is not `\long`, as it sets its contents inside a `hbox`.

TeXhackers note: This function is a wrapper around `\vcenter`, `\hbox:n`, and `\mathpalette`. Inside the `hbox` math mode is reentered.

Part III Implementation

Start the DocStrip guards.

```
1 {*package}  
   Set the internal prefix for this package so that DocStrip knows what to do.  
2 <@@=moremath>
```

1 Initial Setup

First set the required version of L^AT_EX, we need at least

```
3 \NeedsTeXFormat{LaTeX2e}[2022-11-01]
```

for key-value option handling, xparse-like document commands (especially for using the = option specification) and hooks.

Then identify this package as `moremath`.

```
4 \ProvidesExplPackage{moremath}  
5 {2024-07-15}{v0.4.0}{More Math Macros}
```

2 Key-value interfaces

To make use of key-value interfaces we need to define a few keys.

2.1 Package load time options

To allow for the conditional definition of macros at load time we define a few keys.

But before doing so we define a few messages to write the package options to the log file. One message to issue if the `bm` option has been given:

```
6 \msg_new:nnn { moremath } {load / bm}  
7 {  
8   Option~'bm'~given.\  
9   Loading~the~bm~package~\msg_line_context:.  
10 }
```

And a more generic message to issue for the other options, which all disable certain parts of the library.

```

11 \msg_new:nnn {moremath} { load / disabling }
12 {
13   Option~'#1'~given.\
14   Disabling~#2~\msg_line_context:.
15 }

```

To store the values of several switch type key-value options we declare several boolean variables.

```

\l__moremath_predef_vector_op_bool 16 \bool_new:N \l__moremath_predef_vector_op_bool
\l__moremath_predef_operators_bool 17 \bool_new:N \l__moremath_predef_operators_bool
\l__moremath_predef_crvector_bool 18 \bool_new:N \l__moremath_predef_crvector_bool
\l__moremath_predef_matrix_bool 19 \bool_new:N \l__moremath_predef_matrix_bool
\l__moremath_predef_abs_bool 20 \bool_new:N \l__moremath_predef_abs_bool

```

The variables `\l__moremath_predef_vector_op_bool`, `\l__moremath_predef_operators_bool`, `\l__moremath_predef_abs_bool`, `\l__moremath_predef_matrix_bool` and `\l__moremath_predef_crvector_bool` control if the predefined macros for vector calculus, delimited operators, the matrix shorthands, the row and column vectors, and the shorthands for absolute value and norm shall be defined.

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

bm Now we define package load time keys:

```

no-vector 21 \keys_define:nn { moremath / load }
no-operators 22 {
no-abs-shorthands We provide an option to conditionally load the bm-package [CMT23] to provide better
no-matrix looking bold symbols.
no-crvector 23 bm .code:n = {
nopredef 24 \msg_info:nn {moremath} {load / bm}
25 \RequirePackage{bm}
26 },
27 bm .value_forbidden:n = true,
28 bm .usage:n = load,

```

Then we define options for en-/disabling predefined macros of this package to avoid name clashes.

```

29 no-vector .bool_set_inverse:N = \l__moremath_predef_vector_op_bool,
30 no-vector .default:n = true,
31 no-vector .initial:n = false,
32 no-vector .usage:n = load,
33 no-operators .bool_set_inverse:N = \l__moremath_predef_operators_bool,
34 no-operators .default:n = true,
35 no-operators .initial:n = false,
36 no-operators .usage:n = load,
37 no-abs-shorthands .bool_set_inverse:N = \l__moremath_predef_abs_bool,
38 no-abs-shorthands .default:n = true,
39 no-abs-shorthands .initial:n = false,
40 no-abs-shorthands .usage:n = load,
41 no-matrix .bool_set_inverse:N = \l__moremath_predef_matrix_bool,
42 no-matrix .initial:n = false,
43 no-matrix .default:n = true,
44 no-matrix .usage:n =load,

```

```

45 no-crvector .bool_set_inverse:N = \l__moremath_predef_crvector_bool,
46 no-crvector .default:n = true,
47 no-crvector .initial:n = false,
48 no-crvector .usage:n = load,
49 nopredef .multichoice:,
50 nopredef / operators .meta:nn = { moremath / load }
51 {
52   no-operators = true
53 },
54 nopredef / vector .meta:nn = { moremath / load }
55 {
56   no-vector = true
57 },
58 nopredef / abs .meta:nn = { moremath / load }
59 {
60   no-abs-shorthands = true,
61 },
62 nopredef / matrix .meta:nn = { moremath / load }
63 {
64   no-matrix = true,
65 },
66 nopredef / crvector .meta:nn = {moremath / load}
67 {
68   no-crvector = true,
69 },
70 nopredef / all .meta:nn = { moremath / load }
71 {
72   no-operators = true,
73   no-vector = true,
74   no-abs-shorthands = true
75 },
76 nopredef .usage:n = load,

```

Unknown package options get passed to mathtools.

```

77 unknown .code:n = {\PassOptionsToPackage{\CurrentOption}{mathtools}},
78 unknown .usage:n = load,
79 }

```

2.2 Keys controlling appearance

We declare several variables to store the values of the keys affecting appearance.

```

\l__moremath_nabla_tl
\l__moremath_nabla_arrow_bool
\l__moremath_nabla_arrow_bold_bool
\l__moremath_grad_op_tl
\l__moremath_laplacian_symb_tl
\l__moremath_laplacian_delta_bool
\l__moremath_laplacian_arrow_bool
\l__moremath_laplacian_tl
\l__moremath_dalembert_symb_tl

```

```

80 \tl_new:N \l__moremath_nabla_tl
81 \bool_new:N \l__moremath_nabla_arrow_bool
82 \bool_new:N \l__moremath_nabla_arrow_bold_bool
83 \tl_new:N \l__moremath_grad_op_tl

```

The symbol to use as “nabla” is stored in `\l__moremath_nabla_tl`. The variables `\l__moremath_nabla_arrow_bool` and `\l__moremath_nabla_arrow_bold_bool` determine if the nabla-symbol shall have an arrow over itself and/or be bold respectively.

The variable `\l__moremath_grad_op_tl` contains a user provided token list to overwrite the built-in gradient operator of the package.

```

84 \tl_new:N \l__moremath_laplacian_symb_tl
85 \bool_new:N \l__moremath_laplacian_delta_bool
86 \bool_new:N \l__moremath_laplacian_arrow_bool

```

```
87 \tl_new:N \l__moremath_laplacian_tl
```

The token list variable `\l__moremath_laplacian_symb_tl` stores the tokens to be used for the Laplace operator. The boolean variable `\l__moremath_laplacian_delta_bool` determines if a delta should be used instead of ∇^2 for the laplacian symbol. If the boolean variable `\l__moremath_laplacian_arrow_bool` a small arrow will be placed over the Laplace operator symbol. If the user wants to overwrite the symbol used for the Laplacian, the user provided list of tokens is stored in the variable `\l__moremath_laplacian_tl`.

Finally we define a variable to hold the symbol to use for the d'Alembert operator.

```
88 \tl_new:N \l__moremath_dalembert_symb_tl
```

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

`\l__moremath_vcenter_bool` Additionally we declare a variable to decide if certain math symbols shall be centered explicitly along the math axis.

```
89 \bool_new:N \l__moremath_vcenter_bool
```

If `\l__moremath_vcenter_bool` is true, the symbols of certain math operators, should be centered explicitly.

(End of definition for `\l__moremath_vcenter_bool`.)

```

nabla First define keys for the vector calculus macros.
arrownabla 90 \keys_define:nn { moremath }
boldnabla 91 {
grad-op First we define keys for use with the gradient and gradient based operators.
laplacian-symb 92 % Symbol to use for the nabla
delta-laplace 93 nabla .tl_set:N = \l__moremath_nabla_tl,
arrownabla 94 nabla .initial:n = {\nabla},
laplacian 95 nabla .value_required:n = true,
dalembert-symb 96 % shall the nabla have an arrow over it
arrownabla 97 arrownabla .bool_set:N = \l__moremath_nabla_arrow_bool,
arrownabla 98 arrownabla .default:n = {true},
arrownabla 99 arrownabla .initial:n = {false},
100 % shall the nabla be bold
boldnabla 101 boldnabla .bool_set:N = \l__moremath_nabla_bold_bool,
boldnabla 102 boldnabla .default:n = {true},
boldnabla 103 boldnabla .initial:n = {false},

```

We also provide an override for the gradient operator

```

104 % Symbol to use for the gradient operator
105 grad-op .tl_set:N = \l__moremath_grad_op_tl,
106 grad-op .value_required:n = true,

```

Then we define keys for the laplacian.

```

107 % Symbol to use for the laplacian
108 laplacian-symb .tl_set:N = \l__moremath_laplacian_symb_tl,
109 laplacian-symb .initial:n = {\l__moremath_nabla_tl},
110 % shall the Laplace operator be a delta
111 delta-laplace .bool_set:N = \l__moremath_laplacian_delta_bool,
112 delta-laplace .initial:n = {false},
113 % shall the laplace operator have an arrow over itself
114 arrownabla .bool_set:N = \l__moremath_laplacian_arrow_bool,
115 arrownabla .default:n = {true},
116 arrownabla .initial:n = {false},

```

```

117 % overwrite the laplace operator
118 laplacian .tl_set:N = \l__moremath_laplacian_tl,
119 laplacian .value_required:n = true,

```

And keys for the d'Alembert operator.

```

120 dalembert-symb .tl_set:N = \l__moremath_dalembert_symb_tl,
121 dalembert-symb .initial:n = {\square},

```

vcenter The `vcenter` option will control the manual centering of certain math operators.

```

122 vcenter .bool_set:N = \l__moremath_vcenter_bool,
123 vcenter .initial:n = true,
124 vcenter .value_required:n = true,
125 }% \keys_define:nn

```

delimiter Then we define some keys for the matrix based environments:

```

fill 126 \keys_define:nn { moremath / matrix }
align 127 {

```

`\l__moremath_matrix_delim_tl` Every one of the following keys stores its value inside a token list variable.

```

\l__moremath_matrix_fill_tl 128 delimiter .tl_set:N = \l__moremath_matrix_delim_tl,
\l__moremath_matrix_align_tl 129 delimiter .initial:n = {},
130 fill .tl_set:N = \l__moremath_matrix_fill_tl,
131 fill .initial:n = {},
132 align .tl_set:N = \l__moremath_matrix_align_tl,
133 align .initial:n = {c},
134 align .value_required:n = true,
135 }

```

The keys `delimiter` and `fill` set the variables `\l__moremath_matrix_delim_tl` and `\l__moremath_matrix_fill_tl` respectively. `\l__moremath_matrix_delim_tl` determines the delimiter in use to surround matrices and `\l__moremath_matrix_fill_tl` determines the fill values of the `\diag`, `\smalldiag`, `\Xdiag` and `\Xsmalldiag` commands, which are used for non-diagonal matrix entries. The variable `\l__moremath_matrix_align_tl` contains the alignment specifier for use with the `matrix*` environment.

(End of definition for `\l__moremath_matrix_delim_tl`, `\l__moremath_matrix_fill_tl`, and `\l__moremath_matrix_align_tl`.)

2.3 Functions for Setting Options

Now we define a function for setting the options within the document

`\moremath_setup:n`

```

136 \cs_new_protected:Nn \moremath_setup:n
137 {
138   \keys_set:nn {moremath} {#1}
139 }

```

(End of definition for `\moremath_setup:n`. This function is documented on page 23.)

Additionally we provide the user with a version of this command.

`\moremathsetup`

```
140 \NewDocumentCommand \moremathsetup {m}
141 {
142   \moremath_setup:n {#1}
143 }
```

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

We also need a function for setting the package load time options. This function should set all given values for all key families and pass unknown options to `mathtools`.

`_moremath_load_time_setup:`

```
144 \cs_new_protected:Nn \_moremath_load_time_setup:
145 {
146   \ProcessKeyOptions[ moremath / load ]
147 }
```

(End of definition for `_moremath_load_time_setup:.`)

3 Package Initialization

We now “initialize” the package by processing the package options, all unknown options are passed to `mathtools` [Høg+24], which is loaded afterwards. Because certain `mathtools`-features are needed by this package, we need to require a version of at least 2004/06/05. As explained in section 2.1 this may also load `bm` if the `bm` package option has been given.

```
148 \_moremath_load_time_setup:
```

If the `no-vector` option has not been given during load time, we also need the `amssymb` package [The] for the `\square` command. We first define a message to inform the user about this.

```
149 \msg_new:nnn { moremath } { load / loading-amssymb }
150 {
151   Vector~calculus~commands~enabled.\\
152   Loading~amssymb~package~\msg_line_context:.
153 }
```

Then we conditionally load the package.

```
154 \bool_if:NT \l_moremath_predef_vector_op_bool
155 {
156   \msg_info:nn { moremath } { load / loading-amssymb }
157   \RequirePackage{amssymb}
158 }
```

Finally we load our most important dependency `mathtools`.

```
159 \RequirePackage{mathtools}[2004/06/05]
```

4 Centering Math Symbols Along the Math-Axis

Certain math constructs such cause \TeX to not center the operator along the math axis, like case of `\mathop{\nabla}\nolimits` vs. `\mathop{\boldsymbol{\nabla}}\nolimits`, as can be seen below.

$$\nabla f(x) \quad \text{vs.} \quad \boldsymbol{\nabla} f(x)$$

As can be seen the bold nabla symbol is slightly higher up than the non-bold version. Because of this it is sometimes useful to manually center some math symbols, along the math axis.¹

`\moremath_vcenter:n` The function `\moremath_vcenter:n` is a wrapper around the `\vcenter` T_EX primitive. It takes a single argument.

#1 : A $\langle t1 \rangle$ containing math mode material to center along the math axis.
This argument is typeset in math mode.

The function uses the `\mathpalette` primitive to switch to the right math style. The function is not in a T_EX-sense long, i.e. it does not take `\par` tokens.

As this function might be useful not only for internal use by `moremath`, we declare it as a public function here.

```
160 \cs_new_protected_nopar:Nn \moremath_vcenter:n
161 {
162   \mathpalette \__moremath_vcenter:Nn {#1}
163 }
```

(End of definition for `\moremath_vcenter:n`. This function is documented on page 26.)

`__moremath_vcenter:Nn` As `\mathpalette` needs as its first argument a macro which takes two arguments (the first is a math style switch and the second the contents).² We define an internal helper function for `\moremath_vcenter:n` for `\mathpalette` to call.

The function `__moremath_vcenter:Nn` takes two arguments:

#1 : The math style macro, which is passed to this function by `\mathpalette`.
#2 : The $\langle t1 \rangle$ to typeset inside the `\vcenter`.

Because of the properties of `\vcenter`, it switches to vertical mode, we need to put the contents to typeset inside a horizontal box. Because of this we also need to reenter math mode, and because of this we need to remove the spacing inserted by entering math mode by setting `\mathsurround` to zero.

```
164 \cs_new_protected_nopar:Nn \__moremath_vcenter:Nn
165 {
166   \vcenter
167   {
168     \hbox:n
169     {
170       $
171       \mathsurround=0pt
172       #1 {#2}
173       $
174     }
175   }
176 }
```

(End of definition for `__moremath_vcenter:Nn`.)

¹The code in this section is inspired by <http://www.tug.org/TUGboat/Articles/tb22-4/tb72per1S.pdf>

²See T_EXSE answer <https://tex.stackexchange.com/a/34412>

Document Level Command

Although unlikely there might arise the need for a document author to center some math along the math axis. For this purpose we are going to define a new document level command.

But first we are going to declare some messages to use by the command. The first message informs the user that the command is not available, because its $\langle csname \rangle$ is already taken.

```
177 \msg_new:nnn { moremath } { csname-already-defined }
178 {
179   Control-sequence-’ #1 ’~is-already~ defined.\\
180   Skipping-definition~\msg_line_context:
181 }
```

The second message should be issued if a command that only works in math mode was given outside of it.

```
182 \msg_new:nnnn { moremath } { vcenter / only-in-math-mode }
183 {
184   Command-#1~used~outside~math~mode~\msg_line_context:.
185 }
186 {
187   The-command-#1~may~only~be~used~inside~math~mode.
188 }
```

\VCenterMath Now to the document level command for centering math along the math axis.

```
189 \cs_if_free:NTF \VCenterMath
190 {
191   \NewDocumentCommand \VCenterMath { m }
192   {
193     \mode_if_math:TF
194     {
195       \moremath_vcenter:n {#1}
196     }{% \mode_if_math:TF FALSE BRANCH
197       \msg_error:nnn { moremath } { vcenter / only-in-math-mode } {\VCenterMath}
198     }
199   }
200 }{% \cs_if_free:NTF \VCenterMath FALSE BRANCH
201   \msg_warning:nnn { moremath } { csname-already-defined } {\VCenterMath}
202 }
```

(End of definition for $\backslash\text{VCenterMath}$. This function is documented on page 22.)

5 Declaring Paired Delimiters for Internal Use

$\backslash\text{__moremath_inparent:w}$ As we are going to use `mathtools`' *paired delimiters* at several places throughout this package, we define *paired delimiters* for internal use, as functions with weird syntax.

```
\_moremath_inbrack:w
\_moremath_inbrace:w
\_moremath_in_vert:w
\_moremath_in_Vert:w
203 \DeclarePairedDelimiter{\_moremath_inparent:w}{\lparen}{\rparen}
204 \DeclarePairedDelimiter{\_moremath_inbrack:w}{\lbrack}{\rbrack}
205 \DeclarePairedDelimiter{\_moremath_inbrace:w}{\lbrace}{\rbrace}
206 \DeclarePairedDelimiter{\_moremath_in_vert:w}{\lvert}{\rvert}
207 \DeclarePairedDelimiter{\_moremath_in_Vert:w}{\lVert}{\rVert}
```

(End of definition for $\backslash\text{__moremath_inparent:w}$ and others.)

6 Delimited Operators

We need three different functions for providing the delimited operators. But as we share a lot of code between those, we define an additional helper function beforehand.

```

\__moremath_operator:Nnn
The function \__moremath_operator:Nnn takes care of the operator part of the new
delimiter. It allows the operator to have super- and subscripts. It takes three arguments.
#1 : The <csname> of the operator to use.
#2 : A <token list>, which is used as the superscript operator.
      This argument may be empty
#3 : A <t1>, which is used as the subscript operator.
      The <t1> may be empty.

208 \cs_new_protected:Nn \__moremath_operator:Nnn
209 {
210   #1
211   % add superscript if present
212   \t1_if_empty:nF {#2} {^#{#2}}
213   % add subscript if present
214   \t1_if_empty:nF {#3} { \c_math_subscript_token {#3} }
215 }

```

(End of definition for __moremath_operator:Nnn.)

We now define three version of the delimited operators.

```

\moremath_delim_op_noscale:NNnnn
\moremath_delim_op_autoscale:NNnnn
\moremath_delim_op_noscale:NNnnn is provides a delimited operator without any scaling
of the delimiters and \moremath_delim_op_autoscale:NNnnn provides a version with
automatic scaling of the delimiters. Both of them take five arguments:
#1 : The <csname> of the operator to use.
      Any operator declared with amsmath's \operatorname and/or \DeclareMathOperator
      is valid for this.
#2 : The <csname> of a paired delimiter declared by mathtools' [Høg+24] \DeclarePairedDelimiter.
#3 : A <t1> to use as the superscript of the operator.
#4 : A <t1> to use as the subscript of the operator.
#5 : A <t1> to insert inside the delimiters.

216 \cs_new_protected_nopar:Nn \moremath_delim_op_noscale:NNnnn
217 {
218   \__moremath_operator:Nnn #1 {#3} {#4}
219   % #2 is the paired delimiter
220   #2 {#5}
221 }
222
223 \cs_new_protected_nopar:Nn \moremath_delim_op_autoscale:NNnnn
224 {
225   \__moremath_operator:Nnn #1 {#3} {#4}
226   % #2 is the paired delimiter
227   #2 * {#5}
228 }

```

(End of definition for \moremath_delim_op_noscale:NNnnn and \moremath_delim_op_autoscale:NNnnn. These functions are documented on page 23.)

```

\moremath_delim_op_manuscale:NNnnn
\moremath_delim_op_manuscale:NNnnn provides a delimited operator with manual
scaling. This version takes six arguments:

```

- #1 : The $\langle csname \rangle$ of the operator to use.
- #2 : The $\langle csname \rangle$ of the paired delimiter declared by `mathtools`' [\[Høg+24\]](#) `\DeclarePairedDelimiter`.
- #3 : A $\langle t1 \rangle$ containing the scaling macro i.e. `\big`, `\Big`, `\Bigg`,...
- #4 : A $\langle t1 \rangle$ containing the superscript of the operator.
The $\langle t1 \rangle$ may be empty
- #5 : A $\langle t1 \rangle$ containing the subscript of the operator
The $\langle t1 \rangle$ may be empty.
- #6 : A $\langle t1 \rangle$ to insert inside the delimiters

```

229 \cs_new_protected_nopar:Nn \moremath_delim_op_manuscale:NNnnnn
230 {
231   \__moremath_operator:Nnn #1 {#4} {#5}
232   % #2 is the paired delimiter
233   #2 [ #3 ] {#6}
234 }

```

We also provide a variant for the scaling macro.

```

235 \cs_generate_variant:Nn \moremath_delim_op_manuscale:NNnnnn {NNVnnn}

```

(End of definition for `\moremath_delim_op_manuscale:NNnnnn`. This function is documented on page 23.)

For the creation of document level commands we also create a function, so that the user is also able to declare new delimited operators. But before we do so we create a message to inform the user if a $\langle csname \rangle$ was already taken.

```

236 \msg_new:nnn { moremath } { delimop / already-defined-skip }
237 {
238   Control-sequence-'#1'-is-already-defined.\
239   Skipping-definition-of-delimited-operator-'#1'~\msg_line_context:.
240 }

```

We also create a message to inform the user about conflicting options given to the command.

```

241 \msg_new:nnn { moremath } { delimop / auto-manu-scale-conflict }
242 {
243   Both-star-and-scale-cmd-given-to-'#1'.\
244   Automatic-scaling-will-be-preferred,~the-size-command-will-be-
245   ignored-\msg_line_context:.
246 }

```

`\moremath_new_delim_op_command:NNN`
`\moremath_new_delim_op_command:cNN`

`\moremath_new_delim_op_command:NNN` takes three arguments:

- #1 : The $\langle csname \rangle$ to be defined.
- #2 : The $\langle csname \rangle$ of the operator to use, which should be an operator declared with `\DeclareMathOperator`.
- #3 : The $\langle csname \rangle$ of the delimiter to use, which should have been declared with `\DeclarePairedDelimiter`.

```

247 \cs_new_protected:Nn \moremath_new_delim_op_command:NNN
248 {
249   \cs_if_free:NTF #1
250   {
251     \exp_args:NNe \NewDocumentCommand #1
252     { s o E{ ^ \char_generate:nn {'_} {8} }{{}{}} m }
253     {
254       \tl_if_novalue:nTF {##2}
255       {

```

```

256 % second argument empty
257 \bool_if:nTF {##1}
258 {
259 % star given
260 \moremath_delim_op_autoscale:NNnnn #2 #3 {##3} {##4} {##5}
261 }{
262 % star not given
263 \moremath_delim_op_noscale:NNnnn #2 #3 {##3} {##4} {##5}
264 }
265 }{
266 % second argument present
267 % star given?
268 \bool_if:nTF {##1}
269 {
270 % Warn if both star and scaling factor are present
271 \msg_warning:nnn { moremath } { delimop / auto-manu-scale-conflict }
272 {#1}
273 \moremath_delim_op_autoscale:NNnnn #2 #3 {##3} {##4} {##5}
274 }{ % FALSE BRANCH
275 \moremath_delim_op_manuscale:NNnnnn #2 #3 {##2} {##3} {##4} {##5}
276 }
277 }
278 }
279 }{ % \cs_if_free:nTF #1 FALSE BRANCH
280 \msg_warning:nnn { moremath } { delimop / already-defined-skip }
281 {#1}
282 }
283 }
284
285 \cs_generate_variant:Nn \moremath_new_delim_op_command:NNN {cNN}

```

(End of definition for `\moremath_new_delim_op_command:NNN`. This function is documented on page 24.)

6.1 Document Level Commands

`\DeclareDelimitedOperator` Finally provide the user with a command to declare an additional delimited operator.

```

286 \NewDocumentCommand\DeclareDelimitedOperator { m m m }
287 {
288 \msg_redirect_name:nnn { moremath } { delimop / already-defined-skip } { error }
289 \moremath_new_delim_op_command:NNN #1 #2 #3
290 \msg_redirect_name:nnn { moremath } { delimop / already-defined-skip } {}
291 }

```

(End of definition for `\DeclareDelimitedOperator`. This function is documented on page 8.)

As `amsmath` [The23] pre-defines the following operators it is only sensible to also define delimited versions of them:

<code>\arccos</code>	arccos	<code>\deg</code>	deg	<code>\lg</code>	lg	<code>\projlim</code>	projlim
<code>\arcsin</code>	arcsin	<code>\det</code>	det	<code>\lim</code>	lim	<code>\sec</code>	sec
<code>\arctan</code>	arctan	<code>\dim</code>	dim	<code>\liminf</code>	lim inf	<code>\sin</code>	sin
<code>\arg</code>	arg	<code>\exp</code>	exp	<code>\limsup</code>	lim sup	<code>\sinh</code>	sinh
<code>\cos</code>	cos	<code>\gcd</code>	gcd	<code>\ln</code>	ln	<code>\sup</code>	sup
<code>\cosh</code>	cosh	<code>\hom</code>	hom	<code>\log</code>	log	<code>\tan</code>	tan
<code>\cot</code>	cot	<code>\inf</code>	inf	<code>\max</code>	max	<code>\tanh</code>	tanh
<code>\coth</code>	coth	<code>\injlim</code>	injlim	<code>\min</code>	min		
<code>\csc</code>	csc	<code>\ker</code>	ker	<code>\Pr</code>	Pr		
		<code>\varinjlim</code>	\varinjlim	<code>\varliminf</code>	\varliminf	<code>\lim</code>	\lim
		<code>\varprojlim</code>	\varprojlim	<code>\varlimsup</code>	\varlimsup	<code>\lim</code>	\lim

`_moremath_new_delim_op_cmds:nN` creates document level macros of the form `\<prefix><op name>`. It declares five versions `\p<op name>`, `\b<op name>`, `\B<op name>`, `\v<op name>` and `\V<op name>`.

The function takes two arguments:

- #1 : A token list which contains `<op name>`
- #2 : The `<csname>` of the operator to use.

```

292 \cs_new_protected:Nn \_moremath_new_delim_op_cmds:nN
293 {
294   \moremath_new_delim_op_command:cNN {p #1} #2 \_moremath_inparent:w
295   \moremath_new_delim_op_command:cNN {b #1} #2 \_moremath_inbrack:w
296   \moremath_new_delim_op_command:cNN {B #1} #2 \_moremath_inbrace:w
297   \moremath_new_delim_op_command:cNN {v #1} #2 \_moremath_in_vert:w
298   \moremath_new_delim_op_command:cNN {V #1} #2 \_moremath_in_Vert:w
299 }
```

(End of definition for `_moremath_new_delim_op_cmds:nN`.)

The decision if the following macros are defined depends on a package load time option.

```

300 \bool_if:NTF \l_moremath_predef_operators_bool
301 {
```

We define the commands for the operators already declared by `amsmath`.

```

\parccos For \arccos,
\barccos 302 \_moremath_new_delim_op_cmds:nN {arccos} \arccos
\Barccos
\varccos (End of definition for \parccos. This function is documented on page 7.)
\Varccos
\parcsin \arcsin,
\barcsin 303 \_moremath_new_delim_op_cmds:nN {arcsin} \arcsin
\Barcsin (End of definition for \parccos. This function is documented on page 7.)
\varcsin
\Varcsin
\parctan \arctan,
\barctan 304 \_moremath_new_delim_op_cmds:nN {arctan} \arctan
\Barctan (End of definition for \parccos. This function is documented on page 7.)
\varctan
\Varctan
\parg \arg
\bararg 305 \_moremath_new_delim_op_cmds:nN {arg} \arg
\Barg
\varg
\Varg
```

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

```
\pcos \cos
\bcos 306 \_moremath_new_delim_op_cmds:nN {cos} \cos
\Bcos
\vcos (End of definition for \parccos. This function is documented on page 7.)
\vcos
\pcosh \cosh,
\bcosh 307 \_moremath_new_delim_op_cmds:nN {cosh} \cosh
\Bcosh
\vcosh (End of definition for \parccos. This function is documented on page 7.)
\vcosh
\pcot \cot,
\bcot 308 \_moremath_new_delim_op_cmds:nN {cot} \cot
\Bcot
\vcot (End of definition for \parccos. This function is documented on page 7.)
\vcot
\pcoth \coth,
\bcoth 309 \_moremath_new_delim_op_cmds:nN {coth} \coth
\Bcoth
\vcoth (End of definition for \parccos. This function is documented on page 7.)
\vcoth
\pcsc \csc,
\bcsc 310 \_moremath_new_delim_op_cmds:nN {csc} \csc
\Bcsc
\vcsc (End of definition for \parccos. This function is documented on page 7.)
\vcsc
\pdeg \deg,
\bdeg 311 \_moremath_new_delim_op_cmds:nN {deg} \deg
\Bdeg
\vdeg (End of definition for \parccos. This function is documented on page 7.)
\vcsc
\pdet \det,
\bdet 312 \_moremath_new_delim_op_cmds:nN {det} \det
\Bdet
\vdet (End of definition for \parccos. This function is documented on page 7.)
\vcsc
\pdim \dim,
\bdim 313 \_moremath_new_delim_op_cmds:nN {dim} \dim
\Bdim
\vdim (End of definition for \parccos. This function is documented on page 7.)
\vcsc
\pexp \exp,
\bexp 314 \_moremath_new_delim_op_cmds:nN {exp} \exp
\Bexp
\vexp (End of definition for \parccos. This function is documented on page 7.)
\vcsc
\pgcd \gcd,
\bgcd 315 \_moremath_new_delim_op_cmds:nN {gcd} \gcd
\Bgcd
\vgcd (End of definition for \parccos. This function is documented on page 7.)
\vcsc
\phom \hom,
\bhom 316 \_moremath_new_delim_op_cmds:nN {hom} \hom
\Bhom
\vhom (End of definition for \parccos. This function is documented on page 7.)
\Vhom
```

```

\pinf \inf,
\binf 317 \_moremath_new_delim_op_cmds:nN {inf} \inf
\Binf
\vinf (End of definition for \parccos. This function is documented on page 7.)
\pinf \Vinf
\pinjlim \injlim,
\binjlim 318 \_moremath_new_delim_op_cmds:nN {injlim} \injlim
\Binjlim
\vinjlim (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \Vpinjlim
\pker \ker,
\bker 319 \_moremath_new_delim_op_cmds:nN {ker} \ker
\Bker
\vker (End of definition for \parccos. This function is documented on page 7.)
\pker \Vpker
\plg \lg,
\blg 320 \_moremath_new_delim_op_cmds:nN {lg} \lg
\Blg
\vlg (End of definition for \parccos. This function is documented on page 7.)
\plg \Vplg
\plim \lim,
\blim 321 \_moremath_new_delim_op_cmds:nN {lim} \lim
\Blim
\vlim (End of definition for \parccos. This function is documented on page 7.)
\plim \Vplim
\pliminf \liminf,
\bliminf 322 \_moremath_new_delim_op_cmds:nN {liminf} \liminf
\Bliminf
\vliminf (End of definition for \parccos. This function is documented on page 7.)
\pliminf \Vpliminf
\plimsup \limsup,
\blimsup 323 \_moremath_new_delim_op_cmds:nN {limsup} \limsup
\Blimsup
\vlimsup (End of definition for \parccos. This function is documented on page 7.)
\plimsup \Vplimsup
\pln \ln
\bln 324 \_moremath_new_delim_op_cmds:nN {ln} \ln
\Bln
\vln (End of definition for \parccos. This function is documented on page 7.)
\pln \Vpln
\plog \log,
\blog 325 \_moremath_new_delim_op_cmds:nN {log} \log
\Blog
\vlog (End of definition for \parccos. This function is documented on page 7.)
\plog \Vplog
\pmax \max,
\bmax 326 \_moremath_new_delim_op_cmds:nN {max} \max
\Bmax
\vmax (End of definition for \parccos. This function is documented on page 7.)
\pmax \Vpmax
\pmin \min,
\blmin 327 \_moremath_new_delim_op_cmds:nN {min} \min
\Blmin
\vmin (End of definition for \parccos. This function is documented on page 7.)
\pmin \Vpmin

```

```

\pPr \Pr,
\bPr 328 \_moremath_new_delim_op_cmds:nN {Pr} \Pr
\BPr
\vPr (End of definition for \parccos. This function is documented on page 7.)
\pprojlim \VPr \projlim,
\bprojlim 329 \_moremath_new_delim_op_cmds:nN {projlim} \projlim
\Bprojlim
\vprojlim (End of definition for \parccos. This function is documented on page 7.)
\Vprojlim \psec
\bsec \sec,
\Bsec 330 \_moremath_new_delim_op_cmds:nN {sec} \sec
\vsec (End of definition for \parccos. This function is documented on page 7.)
\Vsec \psin \sin,
\bsin 331 \_moremath_new_delim_op_cmds:nN {sin} \sin
\Bsin
\vsin (End of definition for \parccos. This function is documented on page 7.)
\Vsin \psinh \sinh,
\bsinh 332 \_moremath_new_delim_op_cmds:nN {sinh} \sinh
\Bsinh
\vsinh (End of definition for \parccos. This function is documented on page 7.)
\Vsinh \psup \sup,
\bsup 333 \_moremath_new_delim_op_cmds:nN {sup} \sup
\Bsup
\vsup (End of definition for \parccos. This function is documented on page 7.)
\Vsup \ptan \tan,
\btan 334 \_moremath_new_delim_op_cmds:nN {tan} \tan
\Btan
\vtan (End of definition for \parccos. This function is documented on page 7.)
\Vtan \ptanh and \tanh.
\btanh 335 \_moremath_new_delim_op_cmds:nN {tanh} \tanh
\Btanh
\vtanh (End of definition for \parccos. This function is documented on page 7.)
\pvarinjlim We also provide delimited versions of \varinjlim, \varprojlim, \varliminf, and
\bvarinjlim \varlimsup.
\Bvarinjlim 336 \_moremath_new_delim_op_cmds:nN {varinjlim} \varinjlim
\vvarinjlim 337 \_moremath_new_delim_op_cmds:nN {varprojlim} \varprojlim
\Vvarinjlim 338 \_moremath_new_delim_op_cmds:nN {varliminf} \varliminf
\pvarprojlim 339 \_moremath_new_delim_op_cmds:nN {varlimsup} \varlimsup
\bvarprojlim (End of definition for \parccos. This function is documented on page 7.)
\Bvarprojlim
\vvarprojlim 340 }{
\Bvarprojlim 341 \msg_info:nnnn {moremath} {load /disabling} {no-operators}
\vvarprojlim 342 {
\pvarliminf 343 predefined-delimited-operator-macros
\bvarliminf 344 }
\Bvarliminf 345 } % End of the conditional
\vvarliminf
\Vvarliminf
\pvarlimsup
\bvarlimsup
\Bvarlimsup
\vvarlimsup
\Vvarlimsup

```


7 Vector Calculus Macros

For providing macros which help with vector differentials, we first need some setup functions.

7.1 Macros Providing Symbols of Operators

Sometimes a symbol should be centered explicitly, as this will depend on the current setting of `vcenter`, i.e. the current value of `\l__moremath_vcenter_bool`, we provide a helper function which puts its argument

#1 : A list of *tokens*

inside a `\vcenter` by means of `\moremath_vcenter:n` or not depending on the current value of `\l__moremath_vcenter_bool`.

```

346 \cs_new_protected_nopar:Nn \__moremath_maybe_vcenter:n
347 {
348   \bool_if:NTF \l__moremath_vcenter_bool
349   {
350     \moremath_vcenter:n {#1}
351   }{
352     #1
353   }
354 }
```

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

`__moremath_gradient_operator_get:` This function returns the gradient operator depending on the current setting of the keys.

```

355 \cs_new_protected:Nn \__moremath_gradient_operator_get:
356 {
357   \tl_if_empty:NTF \l__moremath_grad_op_tl
358   {
359     \mathop
360     {
```

Otherwise we first need to check if the operator shall have an arrow over it. Afterwards if the nabla shall be bold.

```

361       \bool_if:NTF \l__moremath_nabla_arrow_bool
362       {
363         \vec
364         {
```

In case `\l__moremath_nabla_arrow_bool` is true we should *maybe* center the symbol.

```

365           \__moremath_maybe_vcenter:n
366           {
367             \bool_if:NT \l__moremath_nabla_bold_bool
368             {
369               \boldsymbol
370             }
371             \l__moremath_nabla_tl
372           }
373         }
374       }{
```

Like in the case above we should *maybe* center the symbol if `\l__moremath_nabla_`
`bold_bool` is true.

```

375     \bool_if:NTF \l__moremath_nabla_bold_bool
376     {
377         \__moremath_maybe_vcenter:n
378         {
379             \boldsymbol
380             \l__moremath_nabla_tl
381         }
382     }{
383         \l__moremath_nabla_tl
384     }
385 }
386 }% \mathop
387 \nolimits
388 }{

```

If the user provided its own implementation of the operator, we simply return it.

```

389     \l__moremath_grad_op_tl
390 }
391 }

```

(End of definition for `__moremath_gradient_operator_get:.`)

`__moremath_laplace_operator_get:` Then we define a function for returning the laplace operator symbol, depending on the currently set keys. The function wraps the symbol for the operator inside `\mathop` to provide the right spacing.

```

392 \cs_new_protected:Nn \__moremath_laplace_operator_get:
393 {
394     \tl_if_empty:NTF \l__moremath_laplacian_tl
395     {
396         \mathop
397         {
398             \bool_if:NTF \l__moremath_laplacian_delta_bool
399             {
400                 \Delta
401             }{
402                 \bool_if:NTF \l__moremath_laplacian_arrow_bool
403                 {
404                     \vec{
405                         \__moremath_maybe_vcenter:n
406                         {
407                             \bool_if:NT \l__moremath_nabla_bold_bool {\boldsymbol}
408                             \l__moremath_laplacian_symb_tl
409                         }
410                     }
411                 }{
412                     \bool_if:NTF \l__moremath_nabla_bold_bool
413                     {
414                         \__moremath_maybe_vcenter:n
415                         {
416                             \boldsymbol
417                             \l__moremath_laplacian_symb_tl
418                         }

```

```

419         }{
420         \l__moremath_laplacian_symb_tl
421         }
422     }
423 }
424 }\nolimits
425 \bool_if:NF \l__moremath_laplacian_delta_bool
426 {
427     \c_math_superscript_token
428     {
429         2
430     }
431 }
432 }{
433 \l__moremath_laplacian_tl
434 }
435 }

```

(End of definition for `__moremath_laplace_operator_get:.`)

`__moremath_dalembert_operator_get:` This function returns the d'Alembert operator depending on the currently set keys. The symbol for the d'Alembert operator is wrapped inside `\mathop` to provide proper spacing.

```

436 \cs_new_protected:Nn \__moremath_dalembert_operator_get:
437 {
438     \mathop
439     {
440         \l__moremath_dalembert_symb_tl
441     }% \mathop
442     \nolimits
443 }

```

(End of definition for `__moremath_dalembert_operator_get:.`)

7.2 Macros Producing an Operator

After we have defined the symbols it is now time to provide a function which produces the entire gradient operator

`\moremath_gradient_operator:n` This function takes one arguments, the subscript to use with the operator.

```

444 \cs_new_protected_nopar:Nn \moremath_gradient_operator:n
445 {
446     \__moremath_gradient_operator_get:
447     \tl_if_empty:nF {#1} {\c_math_subscript_token {#1}}
448 }

```

Like for the gradient operator we do the same for the laplacian

```

449 \cs_new_protected_nopar:Nn \moremath_laplace_operator:n
450 {
451     \__moremath_laplace_operator_get:
452     \tl_if_empty:nF {#1} {\c_math_subscript_token {#1}}
453 }

```

(End of definition for `\moremath_gradient_operator:n` and `\moremath_laplace_operator:n`. These functions are documented on page 24.)

`\moremath_divergence_operator:n`
`\moremath_curl_operator:n`

Using the already defined gradient operator it is possible to define a function which acts as an operator suitable for representing the divergence operator and the curl operator. Like the gradient operator this functions take one argument
#1 : the subscript of the gradient operator.

```
454 \cs_new_protected_nopar:Nn \moremath_divergence_operator:n
455 {
```

The braces around `\moremath_gradient_operator:n` are necessary to avoid issues with the spacing between `\cdot` and the following `\mathopen` from any braces.³ Example:

With braces: $\nabla \cdot (f(x))$, without braces: $\nabla \cdot f(x)$

```
456 {
457   \moremath_gradient_operator:n {#1}
458 }
459 \cdot
460 }
461
462 \cs_new_protected_nopar:Nn \moremath_curl_operator:n
463 {
464   {
465     \moremath_gradient_operator:n {#1}
466   }
467   \times
468 }
```

(End of definition for `\moremath_divergence_operator:n` and `\moremath_curl_operator:n`. These functions are documented on page 24.)

`\moremath_dalembert_operator:n`

This function produces the d'Alembert operator, including an optional subscript. The function takes one argument:

#1 : A $\langle t1 \rangle$ with the contents of the subscript of the operator.

This variable may be empty, in this case no subscript (not even an empty one) is produced.

```
469 \cs_new_protected:Nn \moremath_dalembert_operator:n
470 {
471   \__moremath_dalembert_operator_get:
472   \tl_if_empty:nF {#1}
473   {
474     \c_math_subscript_token {#1}
475   }
476 }
```

(End of definition for `\moremath_dalembert_operator:n`. This function is documented on page 24.)

7.3 Producing Delimited Vector Calculus Operators

`\moremath_delim_nabla_op_noscale:NNnn`
`\moremath_delim_nabla_op_autoscale:NNnn`

These functions produce a vector calculus operator with $\langle contents \rangle$ inside delimiters. The `autoscale` variant scales the delimiters with the size of $\langle contents \rangle$, the `noscale` variant does no scaling at all.

³See: <https://tex.stackexchange.com/a/223914>

- #1 : The $\langle csname \rangle$ of the operator to use.
This should have the same form as `\moremath_gradient_operator:n`, i.e. the function passed as $\langle csname \rangle$ should accept one argument to typeset as subscript.
- #2 : The $\langle csname \rangle$ of the *paired delimiter* to use.
The paired delimiter has to be one declared using `mathtools' \DeclarePairedDelimiter` command.
- #3 : A $\langle t1 \rangle$ with the subscript of the operator.
- #4 : A $\langle t1 \rangle$ with the contents to typeset inside the delimiters
- We begin with the `noscale` version.

```

477 \cs_new_protected_nopar:Nn \moremath_delim_nabla_op_noscale:NNnn
478 {
479   #1 {#3} #2{#4}
480 }

```

Then we create the version with automatic scaling.

```

481 \cs_new_protected_nopar:Nn \moremath_delim_nabla_op_autoscale:NNnn
482 {
483   #1 {#3} #2 * {#4}
484 }

```

(End of definition for `\moremath_delim_nabla_op_noscale:NNnn` and `\moremath_delim_nabla_op_autoscale:NNnn`. These functions are documented on page 24.)

`\moremath_delim_nabla_op_manuscale:NNnn`

This function produces a vector calculus operator like `\moremath_delim_nabla_op_noscale:NNnn` and `\moremath_delim_nabla_op_autoscale:NNnn`, but with manual scaling. It takes five arguments:

- #1 : The $\langle csname \rangle$ of the operator to use.
- #2 : The $\langle csname \rangle$ of the paired delimiter.
- #3 : A $\langle token list \rangle$ containing the $\langle scale cmd \rangle$ like `\big`, `\Big`, `\bigg`, etc.
- #4 : A $\langle t1 \rangle$ with the subscript of the operator.
- #5 : A $\langle t1 \rangle$ with the contents to typeset inside the delimiters.

```

485 \cs_new_protected_nopar:Nn \moremath_delim_nabla_op_manuscale:NNnnn
486 {
487   #1 {#4} #2 [#3] {#5}
488 }

```

`\moremath_delim_nabla_op_manuscale:NNVnn`

We also declare a variant for passing a variable with the $\langle scale cmd \rangle$ instead of a $\langle token list \rangle$.

```

489 \cs_generate_variant:Nn \moremath_delim_nabla_op_manuscale:NNnnn {NNVnn}

```

(End of definition for `\moremath_delim_nabla_op_manuscale:NNnnn` and `\moremath_delim_nabla_op_manuscale:NNVnn`. These functions are documented on page 24.)

To ease the definition of those macros we define a function for defining the delimited versions.

Of course we also need a way to declare a user interface for this functions. For this purpose we first need some helpers for setting necessary parameters.

`_moremath_parse_kv_args:nN`

- This helper macro takes two arguments,
- #1 : A token list of the key-value arguments.
- #2 : The $\langle csname \rangle$ of a token list to put the scale value in.

The function sets the given keys but before it does so it searches for a key named `scale` and puts it into the second argument. For this to work it is necessary that all values have = in them.

```
490 \cs_new_protected:Nn \__moremath_parse_kv_args:nN
491 {
```

We first put the key-value arguments inside a property list. Afterwards we check if the key `scale` has been given. If yes we pop it and assign it to the second argument. Otherwise we do nothing.

```
492 \prop_set_from_keyval:Nn \l_tmpa_prop {#1}
493
494 \prop_pop:NnNT \l_tmpa_prop {scale} #2 {}
495
496 \keys_set:ne {moremath} {\prop_to_keyval:N \l_tmpa_prop}
497 }
```

(End of definition for `__moremath_parse_kv_args:nN`.)

We also need a warning message for conflicting arguments, to inform the user that one of his options is going to be ignored.

```
498 \msg_new:nnn { moremath } { vector-calc / scale-star-conflict }
499 {
500 Both~star~and~scaling~factor~given~to~'~#1'~.\\
501 Automatic~scaling~will~be~preferred,~the~size~command~'~#2'~will~be~
502 ignored~\msg_line_context:.
503 }
```

```
\__moremath_new_delim_nabla_doc_cmd:NNN
\__moremath_new_delim_nabla_doc_cmd:cNN
```

This function takes three arguments:

- #1** : The *<csname>* of the to be defined command
- #2** : The *<csname>* of the operator to use.
This should be a function like `\moremath_gradient_operator:n`.
- #3** : The *<csname>* of the delimiter function to use.
This should be a macro/command created with `mathtools\DeclarePairedDelimiter` command.

Its purpose is to create a new document level command, for the delimited vector calculus operators.

```
504 \cs_new_protected:Nn \__moremath_new_delim_nabla_doc_cmd:NNN
505 {
506 \cs_if_free:NTF #1
507 {
508 \exp_args:NNe \NewDocumentCommand #1
509 {
510 s = {scale} o E { \char_generate:nn {'_'} {8} } { { } } m
511 }
512 { % command code
513 \group_begin:
514 % optional arguments given?
515 \tl_if_novalue:nF {##2}
516 {
517 \__moremath_parse_kv_args:nN {##2} \l_tmpa_tl
518 }
519 % star given?
520 \bool_if:nTF {##1}
521 {
```

```

522     % scale factor given?
523     \tl_if_empty:NF \l_tmpa_tl
524     {
525         \msg_warning:nnnV { moremath } { vector-calc / scale-star-conflict }
526         {#1} \l_tmpa_tl
527     }
528     \moremath_delim_nabla_op_autoscale:NNnn #2 #3 {##3} {##4}
529 }{ % \bool_if:nTF {##1} FALSE BRANCH
530     % scale factor given?
531     \tl_if_empty:NTF \l_tmpa_tl
532     {
533         \moremath_delim_nabla_op_noscale:NNnn #2 #3 {##3} {##4}
534     }{ % FALSE BRANCH
535         \moremath_delim_nabla_op_manuscale:NNVnn #2 #3 \l_tmpa_tl {##3} {##4}
536     }
537 } % \bool_if:nTF {##1}
538 \group_end:
539 }
540 }{ % \cs_if_free:NTF #1 FALSE BRANCH
541     \msg_warning:nnn { moremath } { vector-calc / already-defined-skip } {#1}
542 } % \cs_if_free:NTF #1
543 }
544 %
545 \cs_generate_variant:Nn \__moremath_new_delim_nabla_doc_cmd:NNN {cNN}

```

(End of definition for `__moremath_new_delim_nabla_doc_cmd:NNN`.)

`__moremath_new_nabla_doc_cmds:nN` This internal function creates five different document level commands at once, using `__moremath_new_delim_nabla_doc_cmd:cNN`. It takes two arguments:

- #1 : The a $\langle suffix\ tl \rangle$ for constructing the command names.
The resulting commands will have the form $\langle p \rangle \langle suffix \rangle$, $\langle b \rangle \langle suffix \rangle$, $\langle B \rangle \langle suffix \rangle$, $\langle v \rangle \langle suffix \rangle$ and $\langle V \rangle \langle suffix \rangle$.
- #2 : The $\langle csname \rangle$ of the operator to use

```

546 \cs_new_protected:Nn \__moremath_new_nabla_doc_cmds:nN
547 {
548     \__moremath_new_delim_nabla_doc_cmd:cNN { p #1 } #2 \__moremath_inparent:w
549     \__moremath_new_delim_nabla_doc_cmd:cNN { b #1 } #2 \__moremath_inbrack:w
550     \__moremath_new_delim_nabla_doc_cmd:cNN { B #1 } #2 \__moremath_inbrace:w
551     \__moremath_new_delim_nabla_doc_cmd:cNN { v #1 } #2 \__moremath_in_vert:w
552     \__moremath_new_delim_nabla_doc_cmd:cNN { V #1 } #2 \__moremath_in_Vert:w
553 }

```

(End of definition for `__moremath_new_nabla_doc_cmds:nN`.)

7.4 Document Level Commands

The predefined macros for vector calculus are also guarded by a package option to be conditionally disabled by the user.

```

554 \bool_if:NTF \l__moremath_predef_vector_op_bool
555 {

```

7.4.1 Standalone Operators

The user might want to use also a non-delimited version of the vector calculus operators, we provide them with a standalone version of those.

As the definition of a new document command can fail if the $\langle csname \rangle$ clashes with some already defined macro, we define an error message to use when defining document level commands.

```
556 \msg_new:nnn { moremath } { vector-calc / already-defined-skip }
557 {
558   Control-sequence- $\#1$ '-is-already-defined.\
559   Skipping-definition-\msg_line_context:.
560 }
```

\grad Now we provide the user with document level commands for $\text{\moremath}_{\langle op \rangle}_-$
 \divergence operator:n.

```

 $\text{\curl}$ 
 $\text{\laplacian}$ 
561 \cs_if_free:NTF \grad
562 {
563   \exp_args:NNe \NewDocumentCommand \grad { !o E{ \char_generate:nn {'_}{8} }{}} }
564   {
565     \group_begin:
566     \tl_if_novalue:nF {#1}
567     {
568       \keys_set:nn {moremath} {#1}
569     }
570     \moremath_gradient_operator:n {#2}
571     \group_end:
572   }
573 }{
574   \msg_warning:nnn {moremath} { vector-calc / already-defined-skip } {\grad}
575 }
576
577 \cs_if_free:NTF \divergence
578 {
579   \exp_args:NNe \NewDocumentCommand \divergence
580   { !o E{ \char_generate:nn {'_}{8} }{}} }
581   {
582     \group_begin:
583     \tl_if_novalue:nF {#1}
584     {
585       \keys_set:nn {moremath} {#1}
586     }
587     \moremath_divergence_operator:n {#2}
588     \group_end:
589   }
590 }{
591   \msg_warning:nnn {moremath} { vector-calc / already-defined-skip } {\divergence}
592 }
593
594 \cs_if_free:NTF \curl
595 {
596   \exp_args:NNe \NewDocumentCommand \curl
597   { !o E{ \char_generate:nn {'_}{8} }{}} }
598   {
599     \group_begin:

```



```

600   \tl_if_novalue:nF {#1}
601   {
602     \keys_set:nn {moremath} {#1}
603   }
604   \moremath_curl_operator:n {#2}
605   \group_end:
606 }
607 }{
608   \msg_warning:nnn {moremath} {vector-calc / already-defined-skip} {\curl}
609 }
610
611 \cs_if_free:NTF \laplacian
612 {
613   \exp_args:NNe \NewDocumentCommand \laplacian
614   { !o E{ \char_generate:nn {'_}{8} }{}} }
615   {
616     \group_begin:
617     \tl_if_novalue:nF {#1}
618     {
619       \keys_set:nn {moremath} {#1}
620     }
621     \moremath_laplace_operator:n {#2}
622     \group_end:
623   }
624 }{
625   \msg_warning:nnn {moremath} { vector-calc / already-defined-skip }
626   {\laplacian}
627 }

```

We now also define a command to use as a standalone d'Alembert operator. As the name `\dalembertian` is a bit cumbersome to type out, I've decided to use one of its alternate names `\quabla`

```

628 \cs_if_free:NTF \quabla
629 {
630   \exp_args:NNe \NewDocumentCommand \quabla
631   { !o E{ \char_generate:nn {'_}{ 8 } }{}} }
632   {
633     \group_begin:
634     \tl_if_novalue:nF {#1}
635     {
636       \keys_set:nn { moremath } {#1}
637     }
638     \moremath_dalembert_operator:n {#2}
639     \group_end:
640   }
641 }{ % \cs_if_free:NTF \quabla FALSE BRANCH
642   \msg_warning:nnn { moremath } { vector-calc / already-defined-skip }
643   {\quabla}
644 }

```

(End of definition for `\grad` and others. These functions are documented on page 9.)

7.4.2 Operators with Delimiters

```

\pgrad Now lets declare the delimited gradient operators. We provide five versions using paren-
\bgrad thesis, brackets, braces, single \vert, and double \Vert.
\Bgrad 645 \_moremath_new_nabla_doc_cmds:nN {grad} \moremath_gradient_operator:n
\vgrad
\Vgrad (End of definition for \pgrad and others. These functions are documented on page 10.)

\pdiv Now we do the same for the divergence operator.
\bdiv 646 \_moremath_new_nabla_doc_cmds:nN {div} \moremath_divergence_operator:n
\Bdiv
\vdiv (End of definition for \pdiv and others. These functions are documented on page 12.)
\Vdiv
\pcurl Now to the curl macros.
\bcurl 647 \_moremath_new_nabla_doc_cmds:nN {curl} \moremath_curl_operator:n
\Bcurl
\vcurl (End of definition for \pcurl and others. These functions are documented on page 13.)
\Vcurl
\plaplacian Next we take care of the laplacian
\blaplacian 648 \_moremath_new_nabla_doc_cmds:nN {laplacian} \moremath_laplace_operator:n
\Blaplacian
\vlaplacian (End of definition for \plaplacian and others. These functions are documented on page 14.)
\Vlaplacian
\pquabla Finally we define delimited commands for the d'Alembert operator.
\bquabla 649 \_moremath_new_nabla_doc_cmds:nN {quabla} \moremath_dalembert_operator:n
\Bquabla
\vquabla (End of definition for \pquabla and others. These functions are documented on page 15.)
\Vquabla
    If the user issued no-vector as a package loading option, write this to the log file.
650 }{ % IF NOT \l__moremath_predef_vector_op_bool
651 \msg_info:nnnn {moremath} { load /disabling } {no-vector}
652 {
653     predefined-vector~calculus~macros
654 }
655 } % END \l__moremath_predef_vector_op_bool

```

8 Macros Producing Matrices and Vectors

8.1 Producing Row and Column Vectors

The functions in this section are used to generate row and column vectors utilizing math-tools [Høg+24] `matrix*` and `smallmatrix*` environments.

```

\l__moremath_vector_entries_seq For generating row or column vectors it is necessary to store the entries inside a variable.
\l__moremath_vector_entries_seq is used for this purpose.

```

```
656 \seq_new:N \l__moremath_vector_entries_seq
```

(End of definition for `\l__moremath_vector_entries_seq`.)

Then we need a function for formatting the actual entries of the vector. We need two version one for the row vector and one for the column vector version.

```

\_moremath_seq_to_column_vector:N
\_moremath_seq_to_row_vector:N

```

These functions take one argument

#1 : A sequence which should be converted to the contents of the single column/row matrix.

They format the input suitable to be put inside a `matrix*` environment. We begin with the column vector version.

```
657 \cs_new_protected_nopar:Nn \__moremath_seq_to_column_vector:N
658 {
659   \seq_use:Nn #1 {\}
660 }
```

Then we get to the row vector.

```
661 \cs_new_protected_nopar:Nn \__moremath_seq_to_row_vector:N
662 {
663   \seq_use:Nn #1 {&}
664 }
```

(End of definition for `__moremath_seq_to_column_vector:N` and `__moremath_seq_to_row_vector:N`.)

In the next step we construct the single row/column matrix from the user provided input.

`\moremath_column_vector:nn` `\moremath_row_vector:nn` The two commands `\moremath_column_vector:n` and `\moremath_row_vector:n` construct the matrix, they both take two arguments.

- #1** : The delimiter specifier.
 This should be one of the prefixes of the `<prefix>matrix*`, environments.
 Another possibility is to issue `small` as this parameter to get an inline matrix.
- #2** : The comma separated contents of the matrix.

```
665 \cs_new_protected_nopar:Nn \moremath_column_vector:nn
666 {
667   \seq_clear:N \l__moremath_vector_entries_seq
668   \seq_set_from_clist:Nn \l__moremath_vector_entries_seq {#2}
669
670   \exp_args:NnNV \begin{#1 matrix*} [ \l__moremath_matrix_align_tl ]
671     \__moremath_seq_to_column_vector:N \l__moremath_vector_entries_seq
672   \end{#1 matrix*}
673 }
674
675 \cs_new_protected_nopar:Nn \moremath_row_vector:nn
676 {
677   \seq_clear:N \l__moremath_vector_entries_seq
678   \seq_set_from_clist:Nn \l__moremath_vector_entries_seq {#2}
679
680   \exp_args:NnNV \begin{#1 matrix*} [ \l__moremath_matrix_align_tl ]
681     \__moremath_seq_to_row_vector:N \l__moremath_vector_entries_seq
682   \end{#1 matrix*}
683 }
```

(End of definition for `\moremath_column_vector:nn` and `\moremath_row_vector:nn`. These functions are documented on page 25.)

`\moremath_column_smallvector:nn` `\moremath_row_smallvector:nn` To make the code more readable, we define functions specifically for creating row and column vectors using the `smallmatrix*` family of environments. These functions take the same arguments as the non-small versions above.

```
684 \cs_new_protected_nopar:Nn \moremath_column_smallvector:nn
685 {
686   \moremath_column_vector:nn {#1 small} {#2}
687 }
688
```

```

689 \cs_new_protected_nopar:Nn \moremath_row_smallvector:nn
690 {
691   \moremath_row_vector:nn {#1 small} {#2}
692 }

```

(End of definition for `\moremath_column_smallvector:nn` and `\moremath_row_smallvector:nn`. These functions are documented on page 25.)

8.2 Shorthands for Simple Matrices

The construction of several simple matrices like diagonal matrices, can be simplified as there is no need to use the `matrix` environment.⁴

`\l_moremath_mat_diag_entries_seq` We construct the matrices row by row, therefore we need to store the currently constructed row inside a variable. The same is true for the diagonal entries which also need to be stored somewhere. We therefore declare two sequence variables `\l_moremath_mat_diag_entries_seq` and `\l_moremath_mat_row_entries_seq` for this purpose

```

693 \seq_clear_new:N \l_moremath_mat_diag_entries_seq
694 \seq_clear_new:N \l_moremath_mat_row_entries_seq

```

(End of definition for `\l_moremath_mat_diag_entries_seq` and `\l_moremath_mat_row_entries_seq`.)

8.2.1 (Anti-)diagonal matrices

We split the construction of the matrix into multiple parts, utilizing internal helper functions.

`_moremath_constr_diagmat_row:n` `_moremath_constr_diagmat_row:n` and `_moremath_constr_antidiagmat_row:n` take one integer argument:

#1 : The number of the current row.

They both construct a matrix row and place it inside the input stream. They take the content of the (anti-)diagonal from the variable `\l_moremath_mat_diag_entries_seq` and use the variable `\l_moremath_mat_row_entries_seq` to store the current row.

```

695 \cs_new_protected:Nn \_moremath_constr_diagmat_row:n
696 {
697   \seq_clear:N \l_moremath_mat_row_entries_seq

```

As all diagonal matrices $M \in A^{m \times n}$, where A is a field, are quadratic i.e. $A^{m \times n} \equiv A^{n \times n}$ the length of the diagonal sequence equals the number of rows and columns of the matrix. We exploit this fact here.

```

698   \int_step_inline:nn {\seq_count:N \l_moremath_mat_diag_entries_seq}
699   {
700     \int_compare:nTF { #1 == ##1 }
701     {
702       \seq_put_right:Nx \l_moremath_mat_row_entries_seq
703       {
704         \seq_item:Nn \l_moremath_mat_diag_entries_seq { #1 }
705       }
706     }{ % false branch
707       \seq_put_right:NV \l_moremath_mat_row_entries_seq \l_moremath_matrix_fill_tl
708     } % \int_compare:nTF { #1 == ##1 }

```

⁴The code in this section is heavily inspired by the following answer on T_EXSE: <https://tex.stackexchange.com/a/539741>

```

709 }
710 \seq_use:Nn \l__moremath_mat_row_entries_seq { & } \\
711 }
712
713 % Anti-diagonal version
714 \cs_new_protected:Nn \__moremath_constr_antidiagmat_row:n
715 {
716   \seq_clear:N \l__moremath_mat_row_entries_seq
717   \int_step_inline:nn {\seq_count:N \l__moremath_mat_diag_entries_seq}
718   {
719     \int_compare:nTF { #1 == ##1 }
720     {
721       % as this is a anti diagonal matrix we put in the elements from the
722       % left so that the first entry is the right most entry
723       \seq_put_left:Nx \l__moremath_mat_row_entries_seq
724       {
725         \seq_item:Nn \l__moremath_mat_diag_entries_seq { #1 }
726       }
727     }{ % false branch
728       \seq_put_left:NV \l__moremath_mat_row_entries_seq \l__moremath_matrix_fill_tl
729     } % \int_compare:nTF { #1 == ##1 }
730   }
731   \seq_use:Nn \l__moremath_mat_row_entries_seq { & } \\
732 }

```

(End of definition for `__moremath_constr_diagmat_row:n` and `__moremath_constr_antidiagmat_row:n`.)

`\moremath_diagonal_matrix:nn`
`\moremath_antidiagonal_matrix:nn`
`\moremath_diagonal_smallmatrix:nn`
`\moremath_antidiagonal_smallmatrix:nn`

The `\moremath_⟨a or d⟩_matrix:nn` and `\moremath_⟨a or d⟩_smallmatrix:nn` family of functions produce a matrix based on `mathtools` [Høg+24] `matrix*` environment. The functions take two arguments.

#1 : The delimiter specifier.

This should be one of the prefixes of the `⟨prefix⟩matrix*` environments.

#2 : The comma separated contents of the (anti-)diagonal.

These functions also use the values of the variables `\l__moremath_matrix_fill_tl` and `\l__moremath_matrix_align_tl`. And modifies the variable `\l__moremath_mat_diag_entries_seq`.

```

733 \cs_new_protected:Nn \moremath_diagonal_matrix:nn
734 {
735   \seq_set_from_clist:Nn \l__moremath_mat_diag_entries_seq { #2 }
736   \exp_args:NnNV \begin{#1 matrix*} [ \l__moremath_matrix_align_tl ]
737     \int_step_function:nN { \seq_count:N \l__moremath_mat_diag_entries_seq }
738     \__moremath_constr_diagmat_row:n
739   \end{#1 matrix*}
740 }
741
742 % Anti-diagonal matrix
743 \cs_new_protected:Nn \moremath_antidiagonal_matrix:nn
744 {
745   \seq_set_from_clist:Nn \l__moremath_mat_diag_entries_seq { #2 }
746   \exp_args:NnNV \begin{ #1 matrix* } [ \l__moremath_matrix_align_tl ]
747     \int_step_function:nN { \seq_count:N \l__moremath_mat_diag_entries_seq }
748     \__moremath_constr_antidiagmat_row:n
749   \end{ #1 matrix* }

```

```

750 }
751
752 % Small versions
753 \cs_new_protected:Nn \moremath_diagonal_smallmatrix:nn
754 {
755   \moremath_diagonal_matrix:nn {#1 small} {#2}
756 }
757
758 \cs_new_protected:Nn \moremath_antidiagonal_smallmatrix:nn
759 {
760   \moremath_antidiagonal_matrix:nn {#1 small} {#2}
761 }

```

For convenience we also define some variants of the above functions.

```

762 \cs_generate_variant:Nn \moremath_diagonal_matrix:nn { n V }
763 \cs_generate_variant:Nn \moremath_diagonal_matrix:nn { V n }
764 \cs_generate_variant:Nn \moremath_diagonal_matrix:nn { V V }
\moremath_diagonal_matrix:nV 765 \cs_generate_variant:Nn \moremath_antidiagonal_matrix:nn { n V }
\moremath_diagonal_matrix:Vn 766 \cs_generate_variant:Nn \moremath_antidiagonal_matrix:nn { V n }
\moremath_diagonal_matrix:VV 767 \cs_generate_variant:Nn \moremath_antidiagonal_matrix:nn { V V }
  \moremath_antidiagonal_matrix:nV 768 \cs_generate_variant:Nn \moremath_diagonal_smallmatrix:nn { n V }
  \moremath_antidiagonal_matrix:Vn 769 \cs_generate_variant:Nn \moremath_diagonal_smallmatrix:nn { V n }
  \moremath_antidiagonal_matrix:VV 770 \cs_generate_variant:Nn \moremath_diagonal_smallmatrix:nn { V V }
  \moremath_diagonal_smallmatrix:nV 771 \cs_generate_variant:Nn \moremath_antidiagonal_smallmatrix:nn { n V }
  \moremath_diagonal_smallmatrix:Vn 772 \cs_generate_variant:Nn \moremath_antidiagonal_smallmatrix:nn { V n }
  \moremath_diagonal_smallmatrix:VV 773 \cs_generate_variant:Nn \moremath_antidiagonal_smallmatrix:nn { V V }
\moremath_antidiagonal_smallmatrix:nV
\moremath_antidiagonal_smallmatrix:Vn
\moremath_antidiagonal_smallmatrix:VV

```

(End of definition for `\moremath_diagonal_matrix:nn` and others. These functions are documented on page 25.)

8.2.2 Identity Matrices

As we already have functions available for producing diagonal matrices, it makes only sense to also provide a shorthand for producing an identity matrix, i.e. a diagonal matrix with “1” along the diagonal.

`_moremath_generate_one_filled_clist:Nn` The function `_moremath_generate_one_filled_clist:Nn` produces a `<clist>`, consisting only of “1” as entries. This function takes two arguments:

#1 : The *csname* of a `<clist var>` to store the `<clist>` into.

#2 : An `<int>` to represent the number of entries to produce.

```

774 \cs_new_protected_nopar:Nn \_moremath_generate_one_filled_clist:Nn
775 {
776   \seq_clear:N \l_tmpa_seq
777   \int_step_inline:nn {#2}
778   {
779     \seq_put_right:NV \l_tmpa_seq \c_one_int
780   }
781   \clist_set_from_seq:NN #1 \l_tmpa_seq
782 }

```

We also define a variant accepting an integer variable.

```

\_moremath_generate_one_filled_clist:NV 783 \cs_generate_variant:Nn \_moremath_generate_one_filled_clist:Nn { N V }

```

(End of definition for `_moremath_generate_one_filled_clist:Nn` and `_moremath_generate_one_filled_clist:NV`.)

`\l_moremath_id_entries_clist` As we want to utilize the `\moremath_diagonal_matrix:VV` and `\moremath_diagonal_smallmatrix:VV` functions for creating the identity matrix we declare an internal `\clist var` called `\l__moremath_id_entries_clist` for passing the `\clist` around.

```
784 \clist_new:N \l__moremath_id_entries_clist
```

(End of definition for `\l__moremath_id_entries_clist`.)

`\moremath_id_matrix:n` These functions are intended to produce an identity matrix from an integer expression.
`\moremath_id_smallmatrix:n` They take one argument.
#1 : The number of diagonal entries.

```
785 \cs_new_protected_nopar:Nn \moremath_id_matrix:n
786 {
787   \clist_clear:N \l__moremath_id_entries_clist
788   \_moremath_generate_one_filled_clist:Nn \l__moremath_id_entries_clist {#1}
789   \moremath_diagonal_matrix:VV \l__moremath_matrix_delim_tl \l__moremath_id_entries_clist
790 }
791 \cs_new_protected_nopar:Nn \moremath_id_smallmatrix:n
792 {
793   \clist_clear:N \l__moremath_id_entries_clist
794   \_moremath_generate_one_filled_clist:Nn \l__moremath_id_entries_clist {#1}
795   \moremath_diagonal_smallmatrix:VV \l__moremath_matrix_delim_tl \l__moremath_id_entries_clist
796 }
```

We also provide variants, which accepts a V-type argument:

```
797 \cs_generate_variant:Nn \moremath_id_matrix:n { V }
798 \cs_generate_variant:Nn \moremath_id_smallmatrix:n { V }
\moremath_id_matrix:V
\moremath_id_smallmatrix:V
```

(End of definition for `\moremath_id_matrix:n` and others. These functions are documented on page 25.)

8.3 Document Level Commands

Now we define document level commands for the previously defined functions.

But before we do so we define a message to be issued in case the targeted `\csname` is already defined elsewhere.

```
799 \msg_new:nnnn { moremath } { matrix / already-defined-doc-cmd-skip }
800 {
801   Control-sequence-’#1’-is-already-defined.\
802   Skipping-definition-\msg_line_context:.
803 }
804 {
805   The-control-sequence-’#1’-has-already\
806   been-defined-by-some-other-package.\
807   And-I-am-refusing-to-overwrite-the-existing-definition,\
808   therefore-I-am-skipping-the-definition-of-this-command.
809 }
```

8.3.1 Row and Column Vectors

We begin with the row and column vector functions. As with the other document level commands, we guard the definitions with a key value option, so that the user can disable them.

```
810 \bool_if:nTF \l_moremath_predef_crvector_bool
811 {
```

```

\cvector First we define the document level command for the bare column vector
\rvector
\smallcvector
\smallrvector
812 \cs_if_free:NTF \cvector
813 {
814   \NewDocumentCommand \cvector { o m }
815   {
816     \group_begin:
817     \tl_if_novalue:nF {#1}
818     {
819       \keys_set:nn { moremath / matrix } {#1}
820     }
821     \moremath_column_vector:nn {\l_moremath_matrix_delim_tl} {#2}
822     \group_end:
823   }
824 }{
825   % issue a warning message if the csname is already taken.
826   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
827   {
828     \cvector
829   }
830 } % \cs_if_free:NTF \cvector
```

and the row vector.

```
831 \cs_if_free:NTF \rvector
832 {
833   \NewDocumentCommand \rvector { o m }
834   {
835     \group_begin:
836     \tl_if_novalue:nF {#1}
837     {
838       \keys_set:nn { moremath / matrix } {#1}
839     }
840     \moremath_row_vector:nn {\l_moremath_matrix_delim_tl} {#2}
841     \group_end:
842   }
843 }{
844   % warn if csname is already taken
845   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip}
846   {
847     \rvector
848   }
849 } % \cs_if_free:NTF \rvector
```

Then we define the smaller inline versions of those commands.

```
850 \cs_if_free:NTF \smallcvector
851 {
852   \NewDocumentCommand \smallcvector { o m }
853   {
```



```

854 \group_begin:
855 \tl_if_novalue:nF {#1}
856 {
857   \keys_set:nn {moremath / matrix} {#1}
858 }
859 \moremath_column_smallvector:nn {\l__moremath_matrix_delim_tl} {#2}
860 \group_end:
861 }
862 }{
863 % Issue a warning message if the csname is already taken
864 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
865 {
866   \smallcvector
867 }
868 } % \cs_if_free:NTF \smallcvector
869
870 \cs_if_free:NTF \smallrvector
871 {
872   \NewDocumentCommand \smallrvector { o m }
873   {
874     \group_begin:
875     \tl_if_novalue:nF {#1}
876     {
877       \keys_set:nn { moremath / matrix } {#1}
878     }
879     \moremath_row_smallvector:nn {\l__moremath_matrix_delim_tl} {#2}
880     \group_end:
881   }
882 }{
883 % warn if csname is taken
884 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
885 {
886   \smallrvector
887 }
888 }

```

(End of definition for `\cvector` and others. These functions are documented on page 16.)

Commands with Pre-defined Delimiters Next we define several shorthands to for the commonly used delimiters, to avoid code duplication, we first create some helper functions which define those functions.

The function `__moremath_new_vector_shorth_doc_cmd:NNn` creates a new vector shorthand, command. It takes three arguments:

- #1** : The *⟨csname⟩* to be defined.
- #2** : The *⟨function⟩* to use for this shorthand.
This should be one of the `\moremath_⟨type⟩_⟨size⟩vector:nn` like commands.
- #3** : The *⟨delimiter⟩* to use.
Usually one of `p, b, B, v, V`.

```

889 \cs_new_protected:Nn \__moremath_new_vector_shorth_doc_cmd:NNn
890 {
891   \cs_if_free:NTF #1
892   {

```

```

893 \NewDocumentCommand #1 { o m }
894 {
895   \group_begin:
896   % set the delimiter key pre-set for this function
897   \keys_set:nn {moremath / matrix } {delimiter = #3}
898   \tl_if_novalue:nF {##1}
899   {
900     \keys_set:nn {moremath / matrix } {##1}
901   }
902   #2 {\l__moremath_matrix_delim_tl} {##2}
903   \group_end:
904 }
905 }{
906 % warn if csname is taken
907 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
908 {
909   #1
910 }
911 }
912 }

```

(End of definition for `__moremath_new_vector_shorth_doc_cmd:NNn.`)

`\pcvector` `\bcvector` Now we define shorthands for all of the matrix types so that the user does not have to specify `delimiter=(delim)` every time. We begin with the column vector.

```

\Bcvector 913 % parenthesis
\vcvector 914 \__moremath_new_vector_shorth_doc_cmd:NNn \pcvector \moremath_column_vector:nn {p}
\Vcvector 915 % brackets
\prvector 916 \__moremath_new_vector_shorth_doc_cmd:NNn \bcvector \moremath_column_vector:nn {b}
\brvector 917 % braces
\Brvector 918 \__moremath_new_vector_shorth_doc_cmd:NNn \Bcvector \moremath_column_vector:nn {B}
\vrvector 919 % single vert
\Vrvector 920 \__moremath_new_vector_shorth_doc_cmd:NNn \vcvector \moremath_column_vector:nn {v}
          921 % double vert
          922 \__moremath_new_vector_shorth_doc_cmd:NNn \Vcvector \moremath_column_vector:nn {V}

```

Now to the row vectors.

```

923 % parenthesis
924 \__moremath_new_vector_shorth_doc_cmd:NNn \prvector \moremath_row_vector:nn {p}
925 % brackets
926 \__moremath_new_vector_shorth_doc_cmd:NNn \brvector \moremath_row_vector:nn {b}
927 % braces
928 \__moremath_new_vector_shorth_doc_cmd:NNn \Brvector \moremath_row_vector:nn {B}
929 % single vert
930 \__moremath_new_vector_shorth_doc_cmd:NNn \vrvector \moremath_row_vector:nn {v}
931 % double vert
932 \__moremath_new_vector_shorth_doc_cmd:NNn \Vrvector \moremath_row_vector:nn {V}

```

(End of definition for `\pcvector` and others. These functions are documented on page 16.)

`\psmallvector` `\bsmallvector` We also define shorthands for the `\shortcvector` and `\shortrvector` versions.

```

\Bsmallvector 933 % column vectors
\psmallvector 934 % parenthesis
\Vsmallvector 935 \__moremath_new_vector_shorth_doc_cmd:NNn \psmallvector \moremath_column_smallvector:nn
\psmallrvector 936 {p}
\bsmallrvector
\Bsmallrvector
\vsmaillrvector
\Vsmalldrvector

```

```

937 % brackets
938 \_moremath_new_vector_shorth_doc_cmd:NNn \bsmallcvector \moremath_column_smallvector:nn
939 {b}
940 % braces
941 \_moremath_new_vector_shorth_doc_cmd:NNn \Bsmallcvector \moremath_column_smallvector:nn
942 {B}
943 % single vert
944 \_moremath_new_vector_shorth_doc_cmd:NNn \vsmallcvector \moremath_column_smallvector:nn
945 {v}
946 % double vert
947 \_moremath_new_vector_shorth_doc_cmd:NNn \Vsmallcvector \moremath_column_smallvector:nn
948 {V}
949 %
950 % row vectors
951 % parenthesis
952 \_moremath_new_vector_shorth_doc_cmd:NNn \psmallrvector \moremath_row_smallvector:nn {p}
953 % brackets
954 \_moremath_new_vector_shorth_doc_cmd:NNn \bsmallrvector \moremath_row_smallvector:nn {b}
955 % braces
956 \_moremath_new_vector_shorth_doc_cmd:NNn \Bsmallrvector \moremath_row_smallvector:nn {B}
957 % single vert
958 \_moremath_new_vector_shorth_doc_cmd:NNn \vsmallrvector \moremath_row_smallvector:nn {v}
959 % double vert
960 \_moremath_new_vector_shorth_doc_cmd:NNn \Vsmallrvector \moremath_row_smallvector:nn {V}
961
962
963 }{ % \bool_if:nTF \l__moremath_predef_crvector_bool FALSE PATH
964 \msg_info:nnnn {moremath} {load / disabling} {no-crvector}
965 {
966   commands~producing~row~and~column~vectors
967 }
968 } % \bool_if:nTF \l__moremath_predef_crvector_bool

```

(End of definition for `\psmallcvector` and others. These functions are documented on page 17.)

8.3.2 (Anti-)diagonal Matrices

Now to the (anti-)diagonal matrix shorthands, these are also guarded by a key value option.

```

969 \bool_if:nTF \l__moremath_predef_matrix_bool
970 {
971   \diagmat
972   \antidiagmat
973   \smalldiagmat
974   \smallantidiagmat
975   \cs_if_free:NTF \diagmat
976   {
977     \NewDocumentCommand \diagmat { o m }
978     {
979       \group_begin:
980       \tl_if_novalue:nF {#1}
981       {
982         \keys_set:nn { moremath / matrix } {#1}
983       }
984       \moremath_diagonal_matrix:Vn \l__moremath_matrix_delim_tl {#2}
985       \group_end:

```

```

982 }
983 }{
984 \msg_warning:nnn {moremath} {matrix / already-defined-doc-cmd-skip}
985 {
986 \diagmat
987 }
988 } % \cs_if_free:nTF \diagmat
989
990 \cs_if_free:NTF \antidiagmat
991 {
992 \NewDocumentCommand \antidiagmat { o m }
993 {
994 \group_begin:
995 \tl_if_novalue:nF {#1}
996 {
997 \keys_set:nn { moremath / matrix } {#1}
998 }
999 \moremath_antidiagonal_matrix:Vn \l__moremath_matrix_delim_tl {#2}
1000 \group_end:
1001 }
1002 }{
1003 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1004 {
1005 \antidiagmat
1006 }
1007 } % \cs_if_free:nTF \antidiagmat
1008
1009 \cs_if_free:NTF \smallldiagmat
1010 {
1011 \NewDocumentCommand \smallldiagmat { o m }
1012 {
1013 \group_begin:
1014 \tl_if_novalue:nF {#1}
1015 {
1016 \keys_set:nn { moremath / matrix } {#1}
1017 }
1018 \moremath_diagonal_smallmatrix:Vn \l__moremath_matrix_delim_tl {#2}
1019 \group_end:
1020 }
1021 }{
1022 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1023 {
1024 \smallldiagmat
1025 }
1026 }
1027
1028 \cs_if_free:NTF \smallantidiagmat
1029 {
1030 \NewDocumentCommand \smallantidiagmat { o m }
1031 {
1032 \group_begin:
1033 \tl_if_novalue:nF {#1}
1034 {
1035 \keys_set:nn { moremath / matrix } {#1}

```

```

1036   }
1037   \moremath_antidiagonal_smallmatrix:Vn \l__moremath_matrix_delim_tl {#2}
1038   \group_end:
1039 }
1040 }{
1041   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1042   {
1043     \smallantidiagmat
1044   }
1045 }

```

(End of definition for `\diagmat` and others. These functions are documented on page 18.)

(Anti-)diagonal Matrices with Pre-defined Delimiters As it is sort of cumbersome to always specify the delimiter key, we also provide commands with pre-defined delimiters.

To provide several shorthands for delimited matrices, we use a helper function to avoid code duplication. `__moremath_new_matrix_shorth_doc_cmd:NNn` takes three arguments:

- #1 : The `<csname>` to define
- #2 : The `<csname>` of the matrix function to use, which should have the signature `Vn`.
- #3 : The “predefined” delimiter of this version

```

1046 \cs_new_protected:Nn \__moremath_new_matrix_shorth_doc_cmd:NNn
1047 {
1048   \cs_if_free:NTF #1
1049   {
1050     \NewDocumentCommand #1 { o m }
1051     {
1052       \group_begin:
1053       \tl_if_empty:nF {#3}
1054       {
1055         \keys_set:nn { moremath / matrix }
1056         {
1057           delimiter = #3
1058         }
1059       } % \tl_if_empty:nF {#3}
1060       \tl_if_novalue:nF {##1}
1061       {
1062         \keys_set:nn { moremath / matrix } {##1}
1063       }
1064       #2 \l__moremath_matrix_delim_tl {##2}
1065       \group_end:
1066     }
1067   }{
1068     \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1069     {
1070       #1
1071     }
1072   }
1073 }

```

(End of definition for `__moremath_new_matrix_shorth_doc_cmd:NNn`.)

We now define the shorthand commands with predefined delimiters.

`\pdiagmat` We begin with the regular diagonal matrix
`\bdiagmat` 1074 `_moremath_new_matrix_shorth_doc_cmd:NNn \pdiagmat \moremath_diagonal_matrix:Vn {p}`
`\Bdiagmat` 1075 `_moremath_new_matrix_shorth_doc_cmd:NNn \bdiagmat \moremath_diagonal_matrix:Vn {b}`
`\vdiagmat` 1076 `_moremath_new_matrix_shorth_doc_cmd:NNn \Bdiagmat \moremath_diagonal_matrix:Vn {B}`
`\Vdiagmat` 1077 `_moremath_new_matrix_shorth_doc_cmd:NNn \vdiagmat \moremath_diagonal_matrix:Vn {v}`
1078 `_moremath_new_matrix_shorth_doc_cmd:NNn \Vdiagmat \moremath_diagonal_matrix:Vn {V}`

(End of definition for `\pdiagmat` and others. These functions are documented on page 19.)

`\pantidiagmat` Now for the anti-diagonal matrix commands.
`\bantidiagmat` 1079 `_moremath_new_matrix_shorth_doc_cmd:NNn \pantidiagmat`
`\Bantidiagmat` 1080 `\moremath_antidiagonal_matrix:Vn {p}`
`\vantidiagmat` 1081 `_moremath_new_matrix_shorth_doc_cmd:NNn \bantidiagmat`
`\Vantidiagmat` 1082 `\moremath_antidiagonal_matrix:Vn {b}`
1083 `_moremath_new_matrix_shorth_doc_cmd:NNn \Bantidiagmat`
1084 `\moremath_antidiagonal_matrix:Vn {B}`
1085 `_moremath_new_matrix_shorth_doc_cmd:NNn \vantidiagmat`
1086 `\moremath_antidiagonal_matrix:Vn {v}`
1087 `_moremath_new_matrix_shorth_doc_cmd:NNn \Vantidiagmat`
1088 `\moremath_antidiagonal_matrix:Vn {V}`

(End of definition for `\pantidiagmat` and others. These functions are documented on page 19.)

`\psmalldiagmat` We continue with the inline math versions based on the `smallmatrix*` environment.
`\bsmalldiagmat` 1089 `_moremath_new_matrix_shorth_doc_cmd:NNn \psmalldiagmat`
`\Bsmalldiagmat` 1090 `\moremath_diagonal_smallmatrix:Vn {p}`
`\vsmalldiagmat` 1091 `_moremath_new_matrix_shorth_doc_cmd:NNn \bsmalldiagmat`
`\Vsmalldiagmat` 1092 `\moremath_diagonal_smallmatrix:Vn {b}`
1093 `_moremath_new_matrix_shorth_doc_cmd:NNn \Bsmalldiagmat`
1094 `\moremath_diagonal_smallmatrix:Vn {B}`
1095 `_moremath_new_matrix_shorth_doc_cmd:NNn \vsmalldiagmat`
1096 `\moremath_diagonal_smallmatrix:Vn {v}`
1097 `_moremath_new_matrix_shorth_doc_cmd:NNn \Vsmalldiagmat`
1098 `\moremath_diagonal_smallmatrix:Vn {V}`

(End of definition for `\psmalldiagmat` and others. These functions are documented on page 20.)

`\psmallantidiagmat` We provide also anti-diagonal versions of the small matrices.
`\bsmallantidiagmat` 1099 `_moremath_new_matrix_shorth_doc_cmd:NNn \psmallantidiagmat`
`\Bsmallantidiagmat` 1100 `\moremath_antidiagonal_smallmatrix:Vn {p}`
`\vsmallantidiagmat` 1101 `_moremath_new_matrix_shorth_doc_cmd:NNn \bsmallantidiagmat`
`\Vsmallantidiagmat` 1102 `\moremath_antidiagonal_smallmatrix:Vn {b}`
1103 `_moremath_new_matrix_shorth_doc_cmd:NNn \Bsmallantidiagmat`
1104 `\moremath_antidiagonal_smallmatrix:Vn {B}`
1105 `_moremath_new_matrix_shorth_doc_cmd:NNn \vsmallantidiagmat`
1106 `\moremath_antidiagonal_smallmatrix:Vn {v}`
1107 `_moremath_new_matrix_shorth_doc_cmd:NNn \Vsmallantidiagmat`
1108 `\moremath_antidiagonal_smallmatrix:Vn {V}`

(End of definition for `\psmallantidiagmat` and others. These functions are documented on page 20.)

8.3.3 Identity Matrices

We also provide document level commands for producing an identity matrix. These commands are also guarded by the same variable as the other matrix commands (`\l__moremath_predef_matrix_bool`).

`\idmat` We provide two document level commands for producing the identity matrix, one for inline
`\smallidmat` math mode and one for display math mode.

We start with the display math mode version.

```

1109 \cs_if_free:NTF \idmat
1110 {
1111   \NewDocumentCommand \idmat { o m }
1112   {
1113     \group_begin:
1114     \tl_if_novalue:nF {#1}
1115     {
1116       \keys_set:nn { moremath / matrix } {#1}
1117     }
1118     \moremath_id_matrix:n {#2}
1119     \group_end:
1120   }
1121 }{ % \cs_if_free:NTF \idmat FALSE BRANCH
1122   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1123   {\idmat}
1124 }
```

Afterwards we continue with the inline math mode version.

```

1125 \cs_if_free:NTF \smallidmat
1126 {
1127   \NewDocumentCommand \smallidmat { o m }
1128   {
1129     \group_begin:
1130     \tl_if_novalue:nF {#1}
1131     {
1132       \keys_set:nn { moremath / matrix } {#1}
1133     }
1134     \moremath_id_smallmatrix:n {#2}
1135     \group_end:
1136   }
1137 }{ % \cs_if_free:NTF \smallidmat FALSE BRANCH
1138   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1139   {\smallidmat}
1140 }
```

(End of definition for `\idmat` and `\smallidmat`. These functions are documented on page 21.)

Identity Matrices with Pre-defined Delimiters We also define shorthands for the commonly used delimiters around matrices, to avoid code duplication, we first declare a helper function for this.

This function creates a new document level command for an identity matrix like command. It allows pre-setting $\langle kv\ opts \rangle$. The function takes three arguments.
#1 : The $\langle csname \rangle$ of the document level command to define

`_moremath_new_id_matrix_doc_cmd:NNn`

- #2 : The $\langle csname \rangle$ of the function to use
This is indented to be one of $\backslash\text{moremath_id_matrix:n}$ or $\backslash\text{moremath_id_smallmatrix:n}$
- #3 : $\langle kv\ opts \rangle$ to preset in the moremath / matrix namespace for this command
This is meant to be used for pre-setting the key delimiter .

```

1141 \cs_new_protected:NNn \_moremath_new_id_matrix_doc_cmd:NNn
1142 {
1143   \cs_if_free:NTF #1
1144   {
1145     \NewDocumentCommand #1 { o m }
1146     {
1147       \group_begin:
1148       \tl_if_empty:nF {#3}
1149       {
1150         \keys_set:nn { moremath / matrix } {#3}
1151       }
1152       \tl_if_novalue:nF {##1}
1153       {
1154         \keys_set:nn { moremath / matrix } {##1}
1155       }
1156       #2 {##2}
1157       \group_end:
1158     }
1159   }{ % \cs_if_free:NTF #1 FALSE BRANCH
1160     \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1161     {#1}
1162   }
1163 }

```

(End of definition for $\backslash_moremath_new_id_matrix_doc_cmd:NNn$.)

$\backslash\text{pidmat}$ We begin with the display math versions, starting with the version delimited by parenthesis,
 $\backslash\text{bidmat}$
 $\backslash\text{Bidmat}$ 1164 $\backslash_moremath_new_id_matrix_doc_cmd:NNn$ $\backslash\text{pidmat}$ $\backslash\text{moremath_id_matrix:n}$ { $\text{delimiter} = p$ }
 $\backslash\text{vidmat}$ continue with the bracketed version,
 $\backslash\text{Vidmat}$ 1165 $\backslash_moremath_new_id_matrix_doc_cmd:NNn$ $\backslash\text{bidmat}$ $\backslash\text{moremath_id_matrix:n}$ { $\text{delimiter} = b$ }
the version using braces,
1166 $\backslash_moremath_new_id_matrix_doc_cmd:NNn$ $\backslash\text{Bidmat}$ $\backslash\text{moremath_id_matrix:n}$ { $\text{delimiter} = B$ }
single vertical lines,
1167 $\backslash_moremath_new_id_matrix_doc_cmd:NNn$ $\backslash\text{vidmat}$ $\backslash\text{moremath_id_matrix:n}$ { $\text{delimiter} = v$ }
and finally double vertical lines.
1168 $\backslash_moremath_new_id_matrix_doc_cmd:NNn$ $\backslash\text{Vidmat}$ $\backslash\text{moremath_id_matrix:n}$ { $\text{delimiter} = V$ }

(End of definition for $\backslash\text{pidmat}$ and others. These functions are documented on page 21.)

$\backslash\text{psmallidmat}$ Now we also define shorthands for inline math mode. We start again defining the version
 $\backslash\text{bsmallidmat}$ using parenthesis,
 $\backslash\text{Bsmallidmat}$ 1169 $\backslash_moremath_new_id_matrix_doc_cmd:NNn$ $\backslash\text{psmallidmat}$ $\backslash\text{moremath_id_smallmatrix:n}$
 $\backslash\text{vsmallidmat}$ 1170 { $\text{delimiter} = p$ }
 $\backslash\text{Vsmallidmat}$

then brackets,

```
1171 \_moremath_new_id_matrix_doc_cmd:NNn \bsmallidmat \moremath_id_smallmatrix:n
1172 { delimiter = b }
```

then braces,

```
1173 \_moremath_new_id_matrix_doc_cmd:NNn \Bsmallidmat \moremath_id_smallmatrix:n
1174 { delimiter = B }
```

followed by single vertical lines,

```
1175 \_moremath_new_id_matrix_doc_cmd:NNn \vsmallidmat \moremath_id_smallmatrix:n
1176 { delimiter = v }
```

and finally double vertical lines.

```
1177 \_moremath_new_id_matrix_doc_cmd:NNn \Vsmallidmat \moremath_id_smallmatrix:n
1178 { delimiter = V }
```

(End of definition for \psmallidmat and others. These functions are documented on page 21.)

```
1179 }{ % \bool_if:nTF \l_moremath_predef_matrix_bool FALSE BRANCH
1180   \msg_info:nnnn {moremath} { load / disabling } { no-matrix }
1181   {
1182     (anti-)diagonal~matrix~commands
1183   }
1184 } % \bool_if:nTF \l_moremath_predef_matrix_bool
```

9 Shorthand Macros for Absolute Value and Norm

We first declare another warning message to inform the user of the case that, the $\langle csnames \rangle$ are already taken.

```
1185 \msg_new:nnnn { moremath } { abs-shorth / csname-already-defined-skip }
1186 {
1187   Control~sequence~'#1'~is~already~defined.\\
1188   Skipping~declaration~of~paired~delimiter~\msg_line_context:\\.\\
1189   Use~package~option~'no-abs-shorthands'~to~disable~the~paired\\
1190   delimiter~shorthands.
1191 }{
1192   The~control~sequence~'#1'~has~already~been\\
1193   defined~by~something~else.\\
1194   I~am~refusing~to~overwrite~its~existing~definition~and~instead~avoid\\
1195   declaring~a~paired~delimiter.\\
1196 }
```

As with the other parts these macros may be conditionally disabled.

```
1197 \bool_if:NTF \l_moremath_predef_abs_bool
1198 {
```

\abs These macros provide shorthands for $|\langle content \rangle|$ and $\|\langle content \rangle\|$.

```
\norm 1199 \cs_if_free:NTF \abs
1200 {
1201   \DeclarePairedDelimiter \abs {\lvert} {\rvert}
1202 }{
1203   % warn if the csname is taken
1204   \msg_warning:nnn { moremath } { abs-shorth / csname-already-defined-skip }
1205   {\abs}
1206 } % \cs_if_free:NTF \abs
```

```

1207
1208 \cs_if_free:NTF \norm
1209 {
1210   \DeclarePairedDelimiter \norm {\lVert} {\rVert}
1211 }{
1212   % warn if csname is already taken
1213   \msg_warning:nnn { moremath } { abs-shorth / csname-already-defined-skip }
1214   {\norm}
1215 } % \cs_if_free:NTF

(End of definition for \abs and \norm. These functions are documented on page 22.)

1216 }{
1217   \msg_info:nnnn {moremath} {load / disabling} {no-abs-shorthands}
1218   {
1219     '\abs'~and~'\norm'-macros
1220   }
1221 } % End of the conditional
1222 </package>

```

Copyright and License

The following copyright notice applies to the `moremath` package:

Copyright © 2024 Marcel Ilg

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

This work has the LPPL maintenance status “maintained”.

The Current Maintainer of this work is Marcel Ilg.

This work consists of the files listed in `MANIFEST.md`.

The file `MANIFEST.md` has to be distributed together with the package.

References

- [CMT23] David Carlisle, Frank Mittelbach, and The LaTeX Project Team. *The `bm` package. Access bold symbols in maths mode*. Version 1.2f. Dec. 19, 2023. URL: <https://ctan.org/pkg/bm> (visited on 07/04/2024).
- [Høg+24] Morten Høgholm et al. *The `mathtools` package. Mathematical tools to use with `amsmath`*. Version 1.30. Mar. 11, 2024. URL: <https://ctan.org/pkg/mathtools> (visited on 07/04/2024).
- [The] The American Mathematical Society. *The `amsfonts` package. TeX fonts from the American Mathematical Society*. Version 3.04. URL: <https://ctan.org/pkg/amsfonts> (visited on 07/04/2024).
- [The23] The LaTeX Project Team. *The `amsmath` package. AMS mathematical facilities for LaTeX*. Version 2.17o. May 13, 2023. URL: <https://ctan.org/pkg/amsmath> (visited on 07/04/2024).

Change History

v0.1.0 – 2024-06-28

General: Initial development release	.. 1
<code>align</code> : Added options for the matrix-based environments of <code>mathtools</code> .	30
<code>dalembert-symb</code> : Added options controlling appearance.	30
<code>\DeclareDelimitedOperator</code> : Added document level command.	36
<code>\laplacian</code> : Added <code>\grad</code> , <code>\divergence</code> , <code>\curl</code> and <code>\laplacian</code> commands.	49
<code>\moremath_antidiagonal_smallmatrix:VV</code> : Added variants.	54
<code>\moremath_antidiagonal_smallmatrix:nn</code> : Added functions producing (anti-)diagonal matrices. Also added versions for inline math.	54
<code>\moremath_curl_operator:n</code> : Added functions.	44
<code>\moremath_delim_nabla_op_autoscale:NNnn</code> : Added <code>noscale</code> and <code>autoscale</code> functions.	45
<code>\moremath_delim_nabla_op_manuscale:NNVnn</code> : Add variant.	45
<code>\moremath_delim_nabla_op_manuscale:NNnnn</code> : Add function.	45
<code>\moremath_delim_op_autoscale:NNnnn</code> : Added <code>noscale</code> and <code>autoscale</code> versions.	34
<code>\moremath_delim_op_manuscale:NNVnnn</code> : Function added.	35
<code>\moremath_laplace_operator:n</code> : Added functions.	43
<code>\l_moremath_matrix_align_tl</code> : Added internal variables.	30
<code>\moremath_new_delim_op_command:cNN</code> : Added function and variant.	36
<code>\moremath_row_smallvector:nn</code> : Added functions producing inline math row- and column vectors.	52
<code>\moremath_row_vector:nn</code> : Added functions producing row- and column vectors.	51
<code>\moremath_setup:n</code> : Added function for setting options.	30
<code>\moremathsetup</code> : Added document command for setting package options.	31

<code>nopredef</code> : Added package load-time options.	28
<code>\norm</code> : Added shorthands for absolute value and norm.	66
<code>\smallantidiagmat</code> : Added commands producing (anti-)diagonal matrices, including inline math versions.	61
<code>\smallrvector</code> : Added commands producing row and column vectors, including inline math versions.	57
<code>\Vantidiagmat</code> : Added commands for delimited antidiagonal matrices.	62
<code>\Varccos</code> : Added delimited document commands for all <code>amsmath</code> -defined operators.	37
<code>\Vcurl</code> : Added commands for delimited curl operators.	50
<code>\Vdiagmat</code> : Added commands for delimited diagonal matrices.	62
<code>\Vdiv</code> : Added commands for delimited divergence operators.	50
<code>\Vgrad</code> : Added commands for delimited gradient operators.	50
<code>\Vlaplacian</code> : Added commands for delimited Laplace operators.	50
<code>\Vrvector</code> : Added commands for row and column vectors with predefined delimiters.	58
<code>\Vsmallantidiagmat</code> : Added commands for delimited anti-diagonal matrices suitable for inline math.	62
<code>\Vsmalldiagmat</code> : Added commands for delimited diagonal matrices suitable for inline math.	62
<code>\Vsmallrvector</code> : Added commands for inline math row and column vectors with predefined delimiters.	59

v0.2.0 – 2024-07-04

General: Fixed example to match description.	13
Fixed syntax description of standalone operators not matching reality.	15
New: Load <code>amssymb</code> if the <code>no-vector</code> option has <i>not</i> been given.	31
<code>dalembert-symb</code> : Added: New option <code>dalembert-symb</code> .	30

<code>\moremath_antidiagonal_smallmatrix:VV:</code>	<code>\Vidmat:</code> Added document level commands <code>\pidmat</code> , <code>\bidmat</code> , <code>\Bidmat</code> , <code>\vidmat</code> , and <code>\Vidmat</code>	54	64
Added <code>nV</code> and <code>VV</code> variants.	<code>\Vquabla:</code> Added commands for a delimited d'Alembert operator.		50
<code>\moremath_dalembert_operator:n:</code>	Added new function <code>\moremath_dalembert_operator:n</code>	44	
Added function: <code>\moremath_dalembert_operator_get::</code>	<code>\Vsmallidmat:</code> Added document level commands <code>\psmallidmat</code> , <code>\bsmallidmat</code> , <code>\vsmallidmat</code> and <code>\Vsmallidmat</code>	43	65
<code>\l_moremath_dalembert_symb_tl:</code>	Added: New variable <code>\l_moremath_dalembert_symb_tl</code>	29	
Explicitly declared variables before using them to set keys.	General: Split the documentation into several parts.	29	1
<code>\moremath_generate_one_filled_clist:Nn:</code>	<code>\laplacian:</code> Changed: Spaces between the <code><csname></code> and the optional argument are now disallowed for <code>\grad</code> , <code>\divergence</code> , <code>\curl</code> and <code>\laplacian</code>	55	49
Added function.	<code>\moremath_vcenter:n:</code> Added new function.	55	32
<code>\l_moremath_id_entries_clist:</code>	<code>\quabla:</code> Changed: Spaces between the <code><csname></code> and the optional argument are now disallowed.		49
Added variable.	<code>vcenter:</code> Added new option.	55	30
<code>\moremath_id_smallmatrix:n:</code> Added functions <code>\moremath_id_matrix:n</code> and <code>\moremath_id_smallmatrix:n</code> and variants.	<code>\VCenterMath:</code> Added new document level command.	55	33
<code>\l_moremath_predef_abs_bool:</code>	v0.4.0 – 2024-07-15		
Explicitly declared boolean variables used to store key-value options.	General: Breaking Change: Rename the package and the <code><prefix></code> used for function names from <code>commath</code> to <code>moremath</code>	27	1
<code>\quabla:</code> Added <code>\quabla</code> command.		49	
<code>\smallidmat:</code> Added <code>\idmat</code> and <code>\smallidmat</code> document level commands.		63	

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	
<code>\</code>	<code>\cot</code>
8, 13, 151, 179, 238, 243, 500, 558,	7, 37, 38
659, 710, 731, 801, 805, 806, 807,	<code>\coth</code>
1187, 1188, 1189, 1192, 1193, 1194, 1195	7, 37, 38
	<code>\csc</code>
	7, 37, 38
	<code>\DeclareMathOperator</code>
	8, 23, 34, 35
	<code>\deg</code>
	7, 37, 38
	<code>\det</code>
	7, 37, 38
	<code>\dim</code>
	7, 37, 38
	<code>\exp</code>
	7, 37, 38
	<code>\gcd</code>
	7, 37, 38
	<code>\hom</code>
	7, 37, 38
	<code>\inf</code>
	7, 37, 39
	<code>\injl</code>
	7, 37, 39
	<code>\ker</code>
	7, 37, 39
	<code>\lg</code>
	7, 37, 39

<code>\lim</code>	7, 37, 39	<code>\Bdeg</code>	311
<code>\liminf</code>	7, 37, 39	<code>\bdeg</code>	311
<code>\limsup</code>	7, 37, 39	<code>\Bdet</code>	312
<code>\ln</code>	7, 37, 39	<code>\bdet</code>	312
<code>\log</code>	7, 37, 39	<code>\Bdiagmat</code>	19, 1074
<code>\max</code>	7, 37, 39	<code>\bdiagmat</code>	19, 1074
<code>\min</code>	7, 37, 39	<code>\Bdim</code>	313
<code>\Pr</code>	7, 37, 40	<code>\bdim</code>	313
<code>\projlim</code>	7, 37, 40	<code>\Bdiv</code>	12, 646
<code>\sec</code>	7, 37, 40	<code>\bdiv</code>	12, 646
<code>\sin</code>	7, 37, 40	<code>\begin</code>	670, 680, 736, 746
<code>\sinh</code>	7, 37, 40	<code>\Bexp</code>	314
<code>\sup</code>	7, 37, 40	<code>\bexp</code>	314
<code>\tan</code>	7, 37, 40	<code>\Bgcd</code>	315
<code>\tanh</code>	7, 37, 40	<code>\bgcd</code>	315
<code>\varinjlim</code>	7, 37, 40	<code>\Bgrad</code>	10, 645
<code>\varliminf</code>	7, 37, 40	<code>\bgrad</code>	10, 645
<code>\varlimsup</code>	7, 37, 40	<code>\Bhom</code>	316
<code>\varprojlim</code>	7, 37, 40	<code>\bhom</code>	316
amssymb commands:			
<code>\square</code>	6, 31	<code>\Bidmat</code>	21, 68, 1164
<code>\antidiagmat</code>	18, 971	<code>\bidmat</code>	21, 68, 1164
<code>\arccos</code>	302	<code>\Big</code>	7, 10, 23, 24, 35, 45
<code>\arcsin</code>	303	<code>\big</code>	7, 10, 23, 24, 35, 45
<code>\arctan</code>	304	<code>\Bigg</code>	7, 10, 23, 35
<code>\arg</code>	305	<code>\bigg</code>	7, 10, 23, 45
<code>arrowlaplace (option)</code>	6, 90	<code>\Binf</code>	317
<code>arrownabla (option)</code>	5, 90	<code>\binf</code>	317
B			
<code>\Bantidiagmat</code>	19, 1079	<code>\Binjlim</code>	318
<code>\bantidiagmat</code>	19, 1079	<code>\binjlim</code>	318
<code>\Barccos</code>	7, 302	<code>\Bker</code>	319
<code>\barccos</code>	7, 302	<code>\bker</code>	319
<code>\Barcsin</code>	303	<code>\Blaplacian</code>	14, 648
<code>\barcsin</code>	303	<code>\blaplacian</code>	14, 648
<code>\Barctan</code>	304	<code>\Blg</code>	320
<code>\barctan</code>	304	<code>\blg</code>	320
<code>\Barg</code>	305	<code>\Blim</code>	321
<code>\barg</code>	305	<code>\blim</code>	321
<code>\Bcos</code>	306	<code>\Bliminf</code>	322
<code>\bcos</code>	306	<code>\bliminf</code>	322
<code>\Bcosh</code>	307	<code>\Blimsup</code>	323
<code>\bcosh</code>	307	<code>\blimsup</code>	323
<code>\Bcot</code>	308	<code>\Bln</code>	324
<code>\bcot</code>	308	<code>\bln</code>	324
<code>\Bcoth</code>	309	<code>\Blog</code>	325
<code>\bcoth</code>	309	<code>\blog</code>	325
<code>\Bcsc</code>	310	<code>bm (option)</code>	5, 21
<code>\bcsc</code>	310	<code>\Bmax</code>	326
<code>\Bcurl</code>	13, 647	<code>\bmax</code>	326
<code>\bcurl</code>	13, 647	<code>\Bmin</code>	327
<code>\Bcvector</code>	16, 913	<code>\bmin</code>	327
<code>\bcvector</code>	16, 913	<code>boldnabla (option)</code>	5, 90
		<code>\boldsymbol</code>	5, 369, 379, 407, 416

bool commands:		\clist_set_from_seq:NN	781
\bool_if:NTF	154, 300, 348, 361, 367, 375, 398, 402, 407, 412, 425, 554, 1197	\cos	306
\bool_if:nTF	257, 268, 520, 529, 537, 810, 963, 968, 969, 1179, 1184	\cosh	307
\bool_new:N	16, 17, 18, 19, 20, 81, 82, 85, 86, 89	\cot	308
\BPr	328	\coth	309
\bPr	328	cs commands:	
\Bprojlim	329	\cs_generate_variant:Nn	235, 285, 489, 545, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 783, 797, 798
\bprojlim	329	\cs_if_free:NTF	189, 200, 249, 506, 540, 542, 561, 577, 594, 611, 628, 641, 812, 830, 831, 849, 850, 868, 870, 891, 971, 990, 1009, 1028, 1048, 1109, 1121, 1125, 1137, 1143, 1159, 1199, 1206, 1208, 1215
\Bquabla	15, 649	\cs_if_free:nTF	279, 988, 1007
\bquabla	15, 649	\cs_new_protected:Nn	136, 144, 208, 247, 292, 355, 392, 436, 469, 490, 504, 546, 695, 714, 733, 743, 753, 758, 889, 1046, 1141
\Brvector	16, 913	\cs_new_protected_nopar:Nn	160, 164, 216, 223, 229, 346, 444, 449, 454, 462, 477, 481, 485, 657, 661, 665, 675, 684, 689, 774, 785, 791
\brvector	16, 913	\csc	310
\Bsec	330	\curl	12, 14, 67, 68, 561
\bsec	330	\CurrentOption	77
\Bsin	331	\cvector	16, 17, 812
\bsin	331	D	
\Bsinh	332	dalembert-symb (option)	6, 90
\bsinh	332	\DeclareDelimitedOperator	8, 286
\Bsmallantidiagmat	20, 1099	\DeclareMathOperator	24
\bsmallantidiagmat	20, 1099	\DeclarePairedDelimiter	24, 203, 204, 205, 206, 207, 1201, 1210
\Bsmallcvector	17, 933	\deg	311
\bsmallcvector	17, 933	delimiter (option)	6, 126
\Bsmallldiagmat	20, 1089	\Delta	400
\bsmallldiagmat	20, 1089	delta-laplace (option)	6, 90
\Bsmallldimat	21, 68, 1169	\det	312
\bsmallldimat	21, 68, 1169	\diag	30
\Bsmallrvector	17, 933	\diagmat	18, 19, 971
\bsmallrvector	17, 933	\dim	313
\Bsup	333	\divergence	11, 14, 67, 68, 561
\bsup	333	E	
\Btan	334	\end	672, 682, 739, 749
\btan	334	\exp	314
\Btanh	335	exp commands:	
\btanh	335	\exp_args:NNe	251, 508, 563, 579, 596, 613, 630
\Bvarinjlim	336	\exp_args:NnNV	670, 680, 736, 746
\bvarinjlim	336		
\Bvarliminf	336		
\bvarliminf	336		
\Bvarlimsup	336		
\bvarlimsup	336		
\Bvarprojlim	336		
\bvarprojlim	336		
C			
\cdot	459		
char commands:			
\char_generate:nn	252, 510, 563, 580, 597, 614, 631		
clist commands:			
\clist_clear:N	787, 793		
\clist_new:N	784		

	F				
fill (option)					206, 1201
				M	
	G			\backslash mathop	359, 386, 396, 438, 441
\backslash gcd				\backslash mathpalette	32, 162
\backslash grad	9, 11, 12, 14, 67, 68,	561		\backslash mathsurround	171
grad-op (option)		6, 90		mathtools commands:	
group commands:				\backslash DeclarePairedDelimiter	
\backslash group_begin:					8, 22, 23, 34, 35, 45, 46
	25, 513, 565, 582, 599, 616,			\backslash max	326
	633, 816, 835, 854, 874, 895, 975,			\backslash min	327
	994, 1013, 1032, 1052, 1113, 1129, 1147			mode commands:	
\backslash group_end:				\backslash mode_if_math:TF	193, 196
	25, 538, 571, 588, 605, 622,			moremath commands:	
	639, 822, 841, 860, 880, 903, 981,			\backslash moremath_antidiagonal_matrix:nn	
	1000, 1019, 1038, 1065, 1119, 1135, 1157				25, 733, 743, 760, 762, 765, 766,
					767, 999, 1080, 1082, 1084, 1086, 1088
	H			\backslash moremath_antidiagonal_smallmatrix:nn	
hbox commands:					25, 733, 758, 762, 771, 772,
\backslash hbox:n		22, 26, 168			773, 1037, 1100, 1102, 1104, 1106, 1108
\backslash hom		316		\backslash moremath_column_smallvector:nn	
					25,
	I				684, 684, 859, 935, 938, 941, 944, 947
\backslash idmat		21, 68, 1109		\backslash moremath_column_vector:n	51
\backslash inf		317		\backslash moremath_column_vector:nn	25, 665,
\backslash injl		318			665, 686, 821, 914, 916, 918, 920, 922
int commands:				\backslash moremath_curl_operator:n	
\backslash int_compare:nTF		700, 708, 719, 729			24, 454, 462, 604, 647
\backslash int_step_function:nN		737, 747		\backslash moremath_dalembert_operator:n	
\backslash int_step_inline:nn		698, 717, 777			24, 68, 469, 469, 638, 649
\backslash c_one_int		779		\backslash moremath_delim_nabla_op_-	
				autoscale:NNnn	24, 45, 477, 481, 528
	K			\backslash moremath_delim_nabla_op_-	
\backslash ker		319		manuscale:NNnnn	
keys commands:					24, 485, 485, 489, 489, 535
\backslash keys_define:nn		21, 90, 125, 126		\backslash moremath_delim_nabla_op_-	
\backslash keys_set:nn		138, 496,		noscale:NNnn	24, 45, 477, 477, 533
	568, 585, 602, 619, 636, 819, 838,			\backslash moremath_delim_op_autoscale:NNnnn	
	857, 877, 897, 900, 978, 997, 1016,				23, 24, 34, 216, 223, 260, 273
	1035, 1055, 1062, 1116, 1132, 1150, 1154			\backslash moremath_delim_op_manuscale:NNnnnn	
					23, 34, 229, 229, 235, 275
	L			\backslash moremath_delim_op_noscale:NNnnn	
laplacian (option)		6, 90			23, 24, 34, 216, 216, 263
\backslash laplacian		14, 67, 68, 561		\backslash moremath_diagonal_matrix:nn	25,
laplacian-symb (option)		6, 90			55, 733, 733, 755, 762, 762, 763, 764,
\backslash lbrace		205			789, 980, 1074, 1075, 1076, 1077, 1078
\backslash lbrack		204		\backslash moremath_diagonal_smallmatrix:nn	
\backslash lg		320			25, 55, 733, 753, 762, 768, 769, 770,
\backslash lim		321			795, 1018, 1090, 1092, 1094, 1096, 1098
\backslash liminf		322		\backslash moremath_divergence_operator:n	
\backslash limsup		323			24, 454, 454, 587, 646
\backslash ln		324		\backslash moremath_gradient_operator:n	
\backslash log		325			24, 44-46, 444, 444, 457, 465, 570, 645
\backslash lparen		203			
\backslash lVert		207, 1210			

<code>\moremath_id_matrix:n</code>	<code>\l__moremath_laplacian_symb_tl</code> ..
..... 25, 64, 68, 785, 785, 797, 29, 80, 108, 408, 417, 420
797, 1118, 1164, 1165, 1166, 1167, 1168	<code>\l__moremath_laplacian_tl</code>
<code>\moremath_id_smallmatrix:n</code> 29, 80, 118, 394, 433
..... 25, 64, 68, 785, 791, 797,	<code>__moremath_load_time_setup:</code> ...
798, 1134, 1169, 1171, 1173, 1175, 1177 144, 144, 148
<code>\moremath_laplace_operator:n</code> ...	<code>\l__moremath_mat_diag_entries_-</code>
..... 24, 444, 449, 621, 648	<code>seq</code>
<code>\moremath_new_delim_op_command:NNN</code> 52, 53, 693,
..... 24, 35, 247,	698, 704, 717, 725, 735, 737, 745, 747
247, 285, 289, 294, 295, 296, 297, 298	<code>\l__moremath_mat_row_entries_seq</code>
<code>\moremath_row_smallvector:nn</code> 25, 52, 693,
684, 689, 879, 952, 954, 956, 958, 960	697, 702, 707, 710, 716, 723, 728, 731
<code>\moremath_row_vector:n</code>	<code>\l__moremath_mat_row_seq</code>
51	52
<code>\moremath_row_vector:nn</code> . 25, 665,	<code>\l__moremath_matrix_align_tl</code> ...
675, 691, 840, 924, 926, 928, 930, 932 30, 53, 128, 670, 680, 736, 746
<code>\moremath_setup:n</code> .. 23, 136, 136, 142	<code>\l__moremath_matrix_delim_tl</code> ...
<code>\moremath_vcenter:n</code> 30, 128, 789, 795, 821, 840,
..... 26, 32, 41, 160, 160, 195, 350	859, 879, 902, 980, 999, 1018, 1037, 1064
moremath internal commands:	<code>\l__moremath_matrix_fill_tl</code> ...
<code>__moremath_constr_antidiagmat_-</code> 30, 53, 128, 707, 728
<code>row:n</code>	<code>__moremath_maybe_vcenter:n</code>
52, 695, 714, 748 346, 346, 365, 377, 405, 414
<code>__moremath_constr_diagmat_row:n</code>	<code>\l__moremath_nabla_arrow_bold_-</code>
..... 52, 695, 695, 738	<code>bool</code>
<code>__moremath_dalembert_operator_-</code>	80
<code>get:</code>	<code>\l__moremath_nabla_arrow_bool</code> ...
68, 436, 436, 471 28, 41, 80, 97, 361
<code>\l__moremath_dalembert_symb_tl</code> ..	<code>\l__moremath_nabla_bold_bool</code> ...
..... 68, 80, 120, 440 28, 42, 101, 367, 375, 407, 412
<code>__moremath_generate_one_filled_-</code>	<code>\l__moremath_nabla_tl</code>
<code>clist:Nn</code> 28, 80, 93, 109, 371, 380, 383
..... 54, 774, 774, 783, 783, 788, 794	<code>__moremath_new_delim_nabla_doc_-</code>
<code>\l__moremath_grad_op_tl</code>	<code>cmd:NNN</code>
..... 28, 80, 105, 357, 389	504, 504, 545, 548, 549, 550, 551, 552
<code>__moremath_gradient_operator_-</code>	<code>__moremath_new_delim_op_cmds:nN</code>
<code>get:</code> 37, 292, 292, 302, 303,
355, 355, 446	304, 305, 306, 307, 308, 309, 310,
<code>\l__moremath_id_entries_clist</code> ...	311, 312, 313, 314, 315, 316, 317,
..... 55, 784, 787, 788, 789, 793, 794, 795	318, 319, 320, 321, 322, 323, 324,
<code>__moremath_in_Vert:w</code>	325, 326, 327, 328, 329, 330, 331,
..... 203, 207, 298, 552	332, 333, 334, 335, 336, 337, 338, 339
<code>__moremath_in_vert:w</code>	<code>__moremath_new_id_matrix_doc_-</code>
..... 203, 206, 297, 551	<code>cmd:NNn</code>
<code>__moremath_inbrace:w</code> 1141, 1141, 1164, 1165, 1166,
..... 203, 205, 296, 550	1167, 1168, 1169, 1171, 1173, 1175, 1177
<code>__moremath_inbrack:w</code>	<code>__moremath_new_matrix_shorth_-</code>
..... 203, 204, 295, 549	<code>doc_cmd:NNn</code> 61, 1046, 1046, 1074,
<code>__moremath_inparent:w</code>	1075, 1076, 1077, 1078, 1079, 1081,
..... 203, 203, 294, 548	1083, 1085, 1087, 1089, 1091, 1093,
<code>__moremath_laplace_operator_-</code>	1095, 1097, 1099, 1101, 1103, 1105, 1107
<code>get:</code>	<code>__moremath_new_nabla_doc_-</code>
392, 392, 451	<code>cmds:nN</code>
<code>\l__moremath_laplacian_arrow_-</code> 546, 546, 645, 646, 647, 648, 649
<code>bool</code>	<code>__moremath_new_vector_shorth_-</code>
29, 80, 114, 402	<code>doc_cmd</code>
<code>\l__moremath_laplacian_delta_-</code>	57
<code>bool</code>	
29, 80, 111, 398, 425	

<code>_moremath_new_vector_shorth_</code>	630, 814, 833, 852, 872, 893, 973,
<code>doc_cmd:NNn</code>	889, 914, 916, 918, 920, 922, 924, 926, 928, 930, 932, 935, 938, 941, 944, 947, 952, 954, 956, 958, 960
<code>_moremath_operator:Nnn</code>	34, 208, 208, 218, 225, 231
<code>_moremath_parse_kv_args:nN</code>	490, 490, 517
<code>\l_moremath_predef_abs_bool</code>	27, 16, 37, 1197
<code>\l_moremath_predef_crvector_</code>	
<code>bool</code>	27, 16, 45, 810, 963, 968
<code>\l_moremath_predef_matrix_bool</code>	27, 63, 16, 41, 969, 1179, 1184
<code>\l_moremath_predef_operators_</code>	
<code>bool</code>	27, 16, 33, 300
<code>\l_moremath_predef_vector_op_</code>	
<code>bool</code>	27, 16, 29, 154, 554, 650, 655
<code>_moremath_seq_to_column_</code>	
<code>vector:N</code>	657, 657, 671
<code>_moremath_seq_to_row_vector:N</code>	657, 661, 681
<code>_moremath_vcenter:Nn</code>	32, 162, 164, 164
<code>\l_moremath_vcenter_bool</code>	29, 41, 89, 122, 348
<code>\l_moremath_vector_entries_seq</code>	50, 656, 667, 668, 671, 677, 678, 681
<code>\moremathsetup</code>	5, 6, 140
msg commands:	
<code>\msg_error:nnn</code>	197
<code>\msg_info:nn</code>	24, 156
<code>\msg_info:nnnn</code> 341, 651, 964, 1180, 1217	
<code>\msg_line_context:</code>	9, 14, 152, 180, 184, 239, 245, 502, 559, 802, 1188
<code>\msg_new:nnn</code>	6, 11, 149, 177, 236, 241, 498, 556
<code>\msg_new:nnnn</code>	182, 799, 1185
<code>\msg_redirect_name:nnn</code>	288, 290
<code>\msg_warning:nnn</code>	201, 271, 280, 541, 574, 591, 608, 625, 642, 826, 845, 864, 884, 907, 984, 1003, 1022, 1041, 1068, 1122, 1138, 1160, 1204, 1213
<code>\msg_warning:nnnn</code>	525
N	
<code>nabla</code> (option)	5, 90
<code>\nabla</code>	94
<code>\NeedsTeXFormat</code>	3
<code>\NewDelimitedOperator</code>	5
<code>\NewDocumentCommand</code>	140, 191, 251, 286, 508, 563, 579, 596, 613,
<code>no-abs-shorthands</code> (option)	5, 22, 21
<code>no-crvector</code> (option)	5, 15, 21
<code>no-matrix</code> (option)	5, 17, 21
<code>no-operators</code> (option)	5, 6, 21
<code>no-vector</code> (option)	5, 9, 21
<code>\nolimits</code>	387, 424, 442
<code>nopredef</code> (option)	5, 21
<code>\norm</code>	22, 1199, 1219
O	
<code>\operatorname</code>	34
options:	
<code>align</code>	6, 126
<code>arrowlaplace</code>	6, 90
<code>arrownabla</code>	5, 90
<code>bm</code>	5, 21
<code>boldnabla</code>	5, 90
<code>dalembert-symb</code>	6, 90
<code>delimiter</code>	6, 126
<code>delta-laplace</code>	6, 90
<code>fill</code>	6, 126
<code>grad-op</code>	6, 90
<code>laplacian</code>	6, 90
<code>laplacian-symb</code>	6, 90
<code>nabla</code>	5, 90
<code>no-abs-shorthands</code>	5, 22, 21
<code>no-crvector</code>	5, 15, 21
<code>no-matrix</code>	5, 17, 21
<code>no-operators</code>	5, 6, 21
<code>no-vector</code>	5, 9, 21
<code>nopredef</code>	5, 21
<code>vcenter</code>	6, 9, 122
P	
<code>\pantidiagmat</code>	19, 1079
<code>\parccos</code>	7, 302
<code>\parcsin</code>	303
<code>\parctan</code>	304
<code>\Parg</code>	305
<code>\PassOptionsToPackage</code>	77
<code>\pcos</code>	306
<code>\pcosh</code>	307
<code>\pcot</code>	308
<code>\pcoth</code>	309
<code>\pcsc</code>	310
<code>\pcurl</code>	13, 647
<code>\pvector</code>	16, 913
<code>\pdeg</code>	311
<code>\pdet</code>	312
<code>\pdiagmat</code>	19, 1074
<code>\pdim</code>	313
<code>\pdiv</code>	12, 646

<code>\pexp</code>	314
<code>\pgcd</code>	315
<code>\pgrad</code>	10, 645
<code>\phom</code>	316
<code>\pidmat</code>	21, 68, 1164
<code>\pinf</code>	317
<code>\pinjlim</code>	318
<code>\pker</code>	319
<code>\plaplacian</code>	14, 648
<code>\plg</code>	320
<code>\plim</code>	321
<code>\pliminf</code>	322
<code>\plimsup</code>	323
<code>\pln</code>	324
<code>\plog</code>	325
<code>\pmax</code>	326
<code>\pmin</code>	327
<code>\pPr</code>	328
<code>\pprojl</code>	329
<code>\pquabla</code>	15, 649
<code>\Pr</code>	328
<code>\ProcessKeyOptions</code>	146
<code>\projlim</code>	329
prop commands:	
<code>\prop_pop:NnNTF</code>	494
<code>\prop_set_from_keyval:Nn</code>	492
<code>\prop_to_keyval:N</code>	496
<code>\l_tmpa_prop</code>	492, 494, 496
<code>\ProvidesExplPackage</code>	4
<code>\rvector</code>	16, 913
<code>\psec</code>	330
<code>\psin</code>	331
<code>\psinh</code>	332
<code>\psmallantidiagmat</code>	20, 1099
<code>\psmallcvector</code>	17, 933
<code>\psmallldiagmat</code>	20, 1089
<code>\psmallldmat</code>	21, 68, 1169
<code>\psmallrvector</code>	17, 933
<code>\psup</code>	333
<code>\ptan</code>	334
<code>\ptanh</code>	335
<code>\pvarinjlim</code>	336
<code>\pvarliminf</code>	336
<code>\pvarlimsup</code>	336
<code>\pvarprojlim</code>	336
Q	
<code>\quabla</code>	15, 49, 68, 628
R	
<code>\rbrace</code>	205
<code>\rbrack</code>	204
<code>\RequirePackage</code>	25, 157, 159
<code>\rparen</code>	203
<code>\rvector</code>	16, 17, 812
<code>\rVert</code>	207, 1210
<code>\rvert</code>	206, 1201
S	
<code>\sec</code>	330
seq commands:	
<code>\seq_clear:N</code>	667, 677, 697, 716, 776
<code>\seq_clear_new:N</code>	693, 694
<code>\seq_count:N</code>	698, 717, 737, 747
<code>\seq_item:Nn</code>	704, 725
<code>\seq_new:N</code>	656
<code>\seq_put_left:Nn</code>	723, 728
<code>\seq_put_right:Nn</code>	702, 707, 779
<code>\seq_set_from_clist:Nn</code>	668, 678, 735, 745
<code>\seq_use:Nn</code>	659, 663, 710, 731
<code>\l_tmpa_seq</code>	776, 779, 781
<code>\shortcvector</code>	58
<code>\shortrvector</code>	58
<code>\sin</code>	331
<code>\sinh</code>	332
<code>\smallantidiagmat</code>	19, 971
<code>\smallcvector</code>	17, 812
<code>\smallldiag</code>	30
<code>\smallldiagmat</code>	19, 971
<code>\smallldmat</code>	21, 68, 1109
<code>\smallrvector</code>	17, 812
<code>\square</code>	121
<code>\sup</code>	333
T	
<code>\tan</code>	334
<code>\tanh</code>	335
TEX and LATEX 2 _ε commands:	
<code>\cdot</code>	44
<code>\mathop</code>	42, 43
<code>\mathopen</code>	44
<code>\mathpalette</code>	22, 26, 32
<code>\mathsurround</code>	32
<code>\par</code>	32
<code>\vcenter</code>	22, 26, 32, 41
<code>\Vert</code>	50
<code>\vert</code>	50
<code>\times</code>	467
tl commands:	
<code>\tl_if_empty:NTF</code>	357, 394, 523, 531
<code>\tl_if_empty:nTF</code>	212, 214, 447, 452, 472, 1053, 1059, 1148
<code>\tl_if_novalue:nTF</code>	254, 515, 566, 583, 600, 617, 634, 817, 836, 855, 875, 898, 976, 995, 1014, 1033, 1060, 1114, 1130, 1152
<code>\tl_new:N</code>	80, 83, 84, 87, 88
<code>\l_tmpa_tl</code>	517, 523, 526, 531, 535

token commands:		\Vhom	316
\c_math_subscript_token	214, 447, 452, 474	\vhom	316
\c_math_superscript_token	427	\Vidmat	21, 68, 1164
V		\vidmat	21, 68, 1164
\Vantidiagmat	19, 1079	\Vinf	317
\vantidiagmat	19, 1079	\vinf	317
\Varccos	7, 302	\Vinjlim	318
\varccos	7, 302	\vinjlim	318
\Varcsin	303	\Vker	319
\varcsin	303	\vker	319
\Varctan	304	\Vlaplacian	14, 648
\varctan	304	\vlaplacian	14, 648
\Varg	305	\Vlg	320
\varg	305	\vlg	320
\varinjlim	336	\Vlim	321
\varliminf	338	\vlim	321
\varlimsup	339	\Vliminf	322
\varprojlim	337	\vliminf	322
vcenter (option)	6, 9, 122	\Vlimsup	323
\vcenter	166	\vlimsup	323
\VCenterMath	22, 23, 189	\Vln	324
\Vcos	306	\vln	324
\vcos	306	\Vlog	325
\Vcosh	307	\vlog	325
\vcosh	307	\Vmax	326
\Vcot	308	\vmax	326
\vcot	308	\Vmin	327
\Vcoth	309	\vmin	327
\vcoth	309	\VPr	328
\Vcsc	310	\vPr	328
\vcsc	310	\Vprojlim	329
\Vcurl	13, 647	\vprojlim	329
\vcurl	13, 647	\Vquabla	15, 649
\Vcvector	16, 913	\vquabla	15, 649
\vcvector	16, 913	\Vrvector	16, 913
\Vdeg	311	\vrvector	16, 913
\vdeg	311	\Vsec	330
\Vdet	312	\vsec	330
\vdet	312	\Vsin	331
\Vdiagmat	19, 1074	\vsin	331
\vdiagmat	19, 1074	\Vsinh	332
\Vdim	313	\vsinh	332
\vdim	313	\Vsmallantidiagmat	20, 1099
\Vdiv	12, 646	\vsmallantidiagmat	20, 1099
\vdiv	12, 646	\Vsmallcvector	17, 933
\Vvec	363, 404	\vsmallcvector	17, 933
\Vexp	314	\Vsmallldiagmat	20, 1089
\vexp	314	\vsmallldiagmat	20, 1089
\Vgcd	315	\Vsmallidmat	21, 68, 1169
\vgcd	315	\vsmallidmat	21, 68, 1169
\Vgrad	10, 645	\Vsmallrvector	17, 933
\vgrad	10, 645	\vsmallrvector	17, 933
		\Vsup	333
		\vsup	333

<code>\Vtan</code>	<u>334</u>	<code>\Vvarliminf</code>	<u>336</u>
<code>\vtan</code>	<u>334</u>	<code>\vvarliminf</code>	<u>336</u>
<code>\Vtanh</code>	<u>335</u>	<code>\Vvarlimsup</code>	<u>336</u>
<code>\vtanh</code>	<u>335</u>	<code>\vvarlimsup</code>	<u>336</u>
<code>\Vvarinjlim</code>	<u>336</u>	<code>\Vvarprojlim</code>	<u>336</u>
<code>\vvarinjlim</code>	<u>336</u>	<code>\vvarprojlim</code>	<u>336</u>