

The `moremath` package*

Marcel Ilg[†]

Released 2024-08-20

Abstract

The `moremath` package provides several document level commands to ease the typesetting and L^AT_EX 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` v0.5.0, dated 2024-08-20.

[†]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	27
2.1	Package load time options	27
2.2	Keys controlling appearance	29
2.3	Functions for Setting Options	31
3	Package Initialization	31
4	Abstractions for T_EX-primitives and L^AT_EX 2_ε-commands	32
4.1	T _E X-primitives	32
4.2	L ^A T _E X 2 _ε -commands	33
5	Centering Math Symbols Along the Math-Axis	34
6	Declaring Paired Delimiters for Internal Use	35
7	Delimited Operators	36
7.1	Document Level Commands	38
8	Vector Calculus Macros	43
8.1	Macros Providing Symbols of Operators	43
8.2	Macros Producing an Operator	45
8.3	Producing Delimited Vector Calculus Operators	46
8.4	Document Level Commands	49
8.4.1	Standalone Operators	50
8.4.2	Operators with Delimiters	52
9	Macros Producing Matrices and Vectors	52
9.1	Producing Row and Column Vectors	52
9.2	Shorthands for Simple Matrices	54
9.2.1	(Anti-)diagonal matrices	54
9.2.2	Identity Matrices	56
9.3	Document Level Commands	57
9.3.1	Row and Column Vectors	58
9.3.2	(Anti-)diagonal Matrices	61
9.3.3	Identity Matrices	65
10	Shorthand Macros for Absolute Value and Norm	67
	Copyright and License	68
	References	68
	Change History	69

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}

```


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:

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

4. Manually scaled version:

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

5. Feynman-notation:

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

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

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

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

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

3. Standalone divergence operator with subscript

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

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

$$\begin{array}{l} \backslash \\ \backslash\text{curl}[\text{arrownabla}] f(x) \\ \backslash \end{array} \qquad \vec{\nabla} \times f(x)$$

3. Standalone curl operator with subscript

$$\begin{array}{l} \backslash \\ \backslash\text{curl}_{\{x\}} f(x,y) \\ \backslash \end{array} \qquad \nabla_x \times f(x,y)$$

5.3.2 Operators with Delimiters

```

\pcurl \langle delim \rangle curl [size cmd] {content}
\bcurl \langle delim \rangle curl [kv opts] {content}
\Bcurl \langle delim \rangle curl [size cmd] _{subscript} {content}
\vcurl \langle delim \rangle curl [kv opts] _{subscript} {content}
\Vcurl \langle delim \rangle curl* [kv opts] {content}

```

```

\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

$$\begin{array}{l} \backslash \\ \backslash\text{pcurl}\{1+x\} \\ \backslash \end{array} \qquad \nabla \times (1+x)$$

2. Bold version with manual scaling and subscript

$$\begin{array}{l} \backslash \\ \backslash\text{pcurl}[\text{boldnabla}=\text{true},\text{scale}=\text{Big}]_{\{x\}}\{1 + \frac{1}{x}\} \\ \backslash \end{array}$$

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

3. Automatic scaling

$$\begin{array}{l} \backslash \\ \backslash\text{pcurl}*\{1 + \frac{1}{x}\} \\ \backslash \end{array} \qquad \nabla \times \left(1 + \frac{1}{x}\right)$$

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]
```

```
\laplacian [kv opts] _{subscript}
```

Updated: 2024-07-08

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}
```

```
\Vlaplacian \{delim\laplacian* [kv opts] {content}
```

```
\{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](#).

<code>\cvector</code>	<code>\cvector</code> [<i>kv opts</i>] { <i>clist</i> }
<code>\rvector</code>	<code>\rvector</code> [<i>kv opts</i>] { <i>clist</i> }

Updated: 2024-08-20

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:

<code>\[</code>		a_1
<code>\cvector{a_{1},a_{2},a_{3}}</code>		a_2
<code>\]</code>		a_3

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

<code>\[</code>		$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$
<code>\cvector[delimiter=p,align=c]{-1,2,3}</code>		
<code>\]</code>		
<code>\[</code>		$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$
<code>\cvector[delimiter=p,align=r]{-1,2,3}</code>		
<code>\]</code>		
<code>\[</code>		$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$
<code>\cvector[delimiter=p,align=l]{-1,2,3}</code>		
<code>\]</code>		

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.

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

Updated: 2024-08-20

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

$$\begin{array}{l} \backslash[\\ \quad \backslash\text{pcvector}\{a_{1},a_{2},a_{3}\} \\ \backslash] \end{array} \quad \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

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>}
```

Updated: 2024-08-20 `\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>}
```

`\vsmallvector` 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
```

Updated: 2024-08-20

Example:

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

An inline version of `\pcvector` looks like this $\left(\frac{1}{0}\right)$.

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
 $\langle \langle \smallldiagmat{1,1} \rangle \rangle$.

An inline version of a diagonal matrix looks like this $\langle \langle \smallldiagmat{1,1} \rangle \rangle$.

```

\psmalldiagmat \psmalldiagmat [(kv opts)] {diagonal}
\bsmalldiagmat \bsmalldiagmat [(kv opts)] {diagonal}
\Bsmalldiagmat \Bsmalldiagmat [(kv opts)] {diagonal}
\vsmalldiagmat \vsmalldiagmat [(kv opts)] {diagonal}
\Vsmalldiagmat \Vsmalldiagmat [(kv opts)] {diagonal}

```

Like the $\langle \langle \mathit{delim} \rangle \rangle \mathit{diagmat}$ 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
 $\langle \langle \psmalldiagmat{1,1} \rangle \rangle$.

An inline math delimited diagonal matrix looks like this $\langle \langle \psmalldiagmat{1,1} \rangle \rangle$.

```

\psmallantidiagmat \psmallantidiagmat [(kv opts)] {diagonal}
\bsmallantidiagmat \bsmallantidiagmat [(kv opts)] {diagonal}
\Bsmallantidiagmat \Bsmallantidiagmat [(kv opts)] {diagonal}
\vsmallantidiagmat \vsmallantidiagmat [(kv opts)] {diagonal}
\Vsmallantidiagmat \Vsmallantidiagmat [(kv opts)] {diagonal}

```

Like the $\langle \langle \mathit{delim} \rangle \rangle \mathit{antidiagmat}$ 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
 $\langle \langle \psmallantidiagmat{1,1} \rangle \rangle$.

A delimited version of an inline math anti-diagonal matrix looks like this
 $\langle \langle \psmallantidiagmat{1,1} \rangle \rangle$.

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

`\[` $\begin{matrix} 1 & \star & \star \\ \star & 1 & \star \\ \star & \star & 1 \end{matrix}$ `\]`
`\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.

`\abs` `\abs` [*size cmd*] {*content*}

`\norm` `\abs*` {*content*}

`\norm` [*size cmd*] {*content*}

`\norm*` {*content*}

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.

`\VCenterMath` `\VCenterMath` {*contents*}

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:NNnnn` and `\moremath_delim_op_autoscale:NNnnn` 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 this 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 {\langle delim spec t1 \rangle} {\langle clist \rangle}
\moremath_column_vector:Vn     \moremath_row_vector:nn {\langle delim spec t1 \rangle} {\langle clist \rangle}
\moremath_row_vector:nn       \moremath_column_smallvector:nn {\langle delim spec t1 \rangle} {\langle clist \rangle}
\moremath_row_vector:Vn       \moremath_row_smallvector:nn {\langle delim spec t1 \rangle} {\langle clist \rangle}
\moremath_column_smallvector:nn
\moremath_column_smallvector:Vn
\moremath_row_smallvector:nn
\moremath_row_smallvector:Vn

```

Updated: 2024-08-20

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 t1⟩` 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 {\langle delim t1 \rangle} {\langle clist \rangle}
\moremath_diagonal_matrix:(nV|Vn|VV)
\moremath_antidiagonal_matrix:nn \moremath_antidiagonal_matrix:nn {\langle delim t1 \rangle} {\langle clist \rangle}
\moremath_antidiagonal_matrix:(nV|Vn|VV) \moremath_diagonal_smallmatrix:nn {\langle delim t1 \rangle} {\langle clist \rangle}
\moremath_diagonal_smallmatrix:nn \moremath_antidiagonal_smallmatrix:nn {\langle delim t1 \rangle}
\moremath_diagonal_smallmatrix:(nV|Vn|VV)
\moremath_antidiagonal_smallmatrix:nn
\moremath_antidiagonal_smallmatrix:(nV|Vn|VV)

```

Updated: 2024-08-20

These functions produce (anti-)diagonal matrices. The argument `⟨delim t1⟩` is a token list, intended to specify the delimiter of the matrix or no delimiter if the `⟨t1⟩` 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.

<code>\moremath_id_matrix:n</code> <code>\moremath_id_matrix:V</code> <code>\moremath_id_smallmatrix:n</code> <code>\moremath_id_smallmatrix:V</code>	<code>\moremath_id_matrix:n</code> $\langle dimensions \rangle$ <code>\moremath_id_smallmatrix:n</code> $\langle dimensions \rangle$ These functions produce a identity matrix with $\langle dimensions \rangle$ rows and columns. $\langle dimensions \rangle$ is expected to be a <i>positive</i> integer. The <code>smallmatrix</code> version utilizes <code>mathtools</code> ' <code>smallmatrix*</code> environment, which makes it suitable for inline math mode. The delimiter around the matrix is determined by the current value of the key <code>matrix / delimiter</code> inside the current group.
--	---

New: 2024-07-04
Updated: 2024-07-15

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.

<code>\moremath_vcenter:n</code>	<code>\moremath_vcenter:n</code> $\langle contents \rangle$ The function <code>\moremath_vcenter:n</code> places $\langle contents \rangle$ inside a <code>hbox</code> , which is centered along the math axis by <code>\vcenter</code> . The contents inside the <code>hbox</code> are set in math mode. The function also applies the “outer” math style to the contents of the <code>hbox</code> .
----------------------------------	--

New: 2024-07-08
Updated: 2024-08-20

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.

Afterwards we provide rollback information, for the `latexrelease` rollback mechanism. (See Mittelbach [Mit18] for more information.)

For the `v0.x.x` versions I've decided on using the date of the first release that was uploaded to CTAN.

- 4 `\DeclareCurrentRelease{v0}{2024-07-15}`

Then identify this package as `moremath`.

```
5 \ProvidesExplPackage{moremath}
6 {2024-08-20}{v0.5.0}{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:

```
7 \msg_new:nnn { moremath } {load / bm}
8 {
9   Option~'bm'~given.\\
10  Loading~the~bm~package~\msg_line_context:.
11 }
```

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

```
12 \msg_new:nnn {moremath} { load / disabling }
13 {
14   Option~'#1'~given.\\
15   Disabling~#2~\msg_line_context:.
16 }
```

```
\l_moremath_predef_vector_op_bool
\l_moremath_predef_operators_bool
\l_moremath_predef_crvector_bool
\l_moremath_predef_matrix_bool
\l_moremath_predef_abs_bool
```

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

```
17 \bool_new:N \l__moremath_predef_vector_op_bool
18 \bool_new:N \l__moremath_predef_operators_bool
19 \bool_new:N \l__moremath_predef_crvector_bool
20 \bool_new:N \l__moremath_predef_matrix_bool
21 \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 22 \keys_define:nn { moremath / load }
no-operators 23 {
```

`no-abs-shorthands` We provide an option to conditionally load the `bm`-package [CMT23] to provide better looking bold symbols.

```
no-matrix
no-crvector
nopredef 24   bm .code:n = {
25     \msg_info:nn {moremath} {load / bm}
26     \RequirePackage{bm}
27   },
```

```

28  bm .value_forbidden:n = true,
29  bm. usage:n = load,

```

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

```

30  no-vector .bool_set_inverse:N = \l__moremath_predef_vector_op_bool,
31  no-vector .default:n = true,
32  no-vector .initial:n = false,
33  no-vector .usage:n = load,
34  no-operators .bool_set_inverse:N = \l__moremath_predef_operators_bool,
35  no-operators .default:n = true,
36  no-operators .initial:n = false,
37  no-operators .usage:n = load,
38  no-abs-shorthands .bool_set_inverse:N = \l__moremath_predef_abs_bool,
39  no-abs-shorthands .default:n = true,
40  no-abs-shorthands .initial:n = false,
41  no-abs-shorthands .usage:n = load,
42  no-matrix .bool_set_inverse:N = \l__moremath_predef_matrix_bool,
43  no-matrix .initial:n = false,
44  no-matrix .default:n = true,
45  no-matrix .usage:n = load,
46  no-crvector .bool_set_inverse:N = \l__moremath_predef_crvector_bool,
47  no-crvector .default:n = true,
48  no-crvector .initial:n = false,
49  no-crvector .usage:n = load,
50  nopredef .multichoice:,
51  nopredef / operators .meta:nn = { moremath / load }
52  {
53    no-operators = true
54  },
55  nopredef / vector .meta:nn = { moremath / load }
56  {
57    no-vector = true
58  },
59  nopredef / abs .meta:nn = { moremath / load }
60  {
61    no-abs-shorthands = true,
62  },
63  nopredef / matrix .meta:nn = { moremath / load }
64  {
65    no-matrix = true,
66  },
67  nopredef / crvector .meta:nn = {moremath / load}
68  {
69    no-crvector = true,
70  },
71  nopredef / all .meta:nn = { moremath / load }
72  {
73    no-operators = true,
74    no-vector = true,
75    no-abs-shorthands = true
76  },
77  nopredef .usage:n = load,

```

Unknown package options get passed to mathtools.

```

78   unknown .code:n = {\PassOptionsToPackage{\CurrentOption}{mathtools}},
79   unknown .usage:n = load,
80 }

```

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

```

```

81 \tl_new:N \l__moremath_nabla_tl
82 \bool_new:N \l__moremath_nabla_arrow_bool
83 \bool_new:N \l__moremath_nabla_arrow_bold_bool
84 \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.

```

85 \tl_new:N \l__moremath_laplacian_symb_tl
86 \bool_new:N \l__moremath_laplacian_delta_bool
87 \bool_new:N \l__moremath_laplacian_arrow_bool
88 \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.

```

89 \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.

```

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

```

```

100   arrownabla .initial:n = {false},
101   % shall the nabla be bold
102   boldnabla .bool_set:N = \l__moremath_nabla_bold_bool,
103   boldnabla .default:n = {true},
104   boldnabla .initial:n = {false},

```

We also provide an override for the gradient operator

```

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

```

Then we define keys for the laplacian.

```

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

```

And keys for the d'Alembert operator.

```

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

```

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

```

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

```

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

```

   fill 127 \keys_define:nn { moremath / matrix }
   align 128 {

```

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

```

\l__moremath_matrix_delim_tl 129 delimiter .tl_set:N = \l__moremath_matrix_delim_tl,
\l__moremath_matrix_fill_tl  130 delimiter .initial:n = {},
\l__moremath_matrix_align_tl 131 fill .tl_set:N = \l__moremath_matrix_fill_tl,
                               132 fill .initial:n = {},
                               133 align .tl_set:N = \l__moremath_matrix_align_tl,
                               134 align .initial:n = {c},
                               135 align .value_required:n = true,
                               136 }

```

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`

```

137 \cs_new_protected:Nn \moremath_setup:n
138 {
139   \keys_set:nn {moremath} {#1}
140 }
```

(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`

```

141 \NewDocumentCommand \moremathsetup {m}
142 {
143   \moremath_setup:n {#1}
144 }
```

(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:`

```

145 \cs_new_protected:Nn \__moremath_load_time_setup:
146 {
147   \ProcessKeyOptions[ moremath / load ]
148 }
```

(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` [Hög+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.

```

149 \__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.

```

150 \msg_new:nnn { moremath } { load / loading-amssymb }
151 {
152   Vector~calculus~commands~enabled.\
153   Loading~amssymb~package~\msg_line_context:.
154 }

```

Then we conditionally load the package.

```

155 \bool_if:NT \l__moremath_predef_vector_op_bool
156 {
157   \msg_info:nn { moremath } { load / loading-amssymb }
158   \RequirePackage{amssymb}
159 }

```

Finally we load our most important dependency `mathtools`.

```

160 \RequirePackage{mathtools}[2004/06/05]

```

4 Abstractions for T_EX-primitives and L^AT_EX 2_ε-commands

As this package makes use of certain T_EX-primitives and L^AT_EX 2_ε-commands, which are either part of the L^AT_EX-kernel or of some package. It is sensible to provide a (package internal) abstraction for those commands, so that they can be switched out once a L^AT_EX 3-command has been defined for them.

4.1 T_EX-primitives

`__moremath_tex_vcenter:n` `__moremath_tex_vcenter:n` is an abstraction for `\tex_vcenter:D` i.e. `\vcenter`. Its single argument is passed to `\tex_vcenter:D`.

```

161 \cs_new_protected:Nn \__moremath_tex_vcenter:n
162 {
163   \tex_vcenter:D {#1}
164 }

```

(End of definition for __moremath_tex_vcenter:n.)

`__moremath_tex_mathpalette:Nn` The function `__moremath_tex_mathpalette:Nn` is an abstraction for the `\mathpalette`, primitive. The function takes two arguments

#1 : The *<cname>* of the macro to pass to `\mathpalette`

This macro **must** accept **two** arguments: The first is a single token containing the current math-style, the second is the contents to “typeset”. See <https://tex.stackexchange.com/questions/34393/the-mysteries-of-mathpalette/34412#34412> for further information.

#2 : The *<tokens>* to forward to the macro passed as first argument.

```

165 \cs_new_protected:Nn \__moremath_tex_mathpalette:Nn
166 {
167   \mathpalette #1 {#2}
168 }

```

(End of definition for __moremath_tex_mathpalette:Nn.)

`_moremath_tex_mathsurround:n` The function `_moremath_tex_mathsurround:n` provides (hopefully) a correct abstraction for the `\mathsurround` primitive. It takes one argument
#1 : A $\langle length\ expr \rangle$ for the whitespace surrounding a math switch.

```

169 \cs_new_protected_nopar:Nn \_moremath_tex_mathsurround:n
170 {
171   \tex_mathsurround:D = #1
172 }

```

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

`_moremath_tex_mathop:n` The function `_moremath_tex_mathop:n` provides an abstraction for the `\mathop` T_EX-primitive. It accepts a single argument:
#1 : $\langle Tokens \rangle$ to treat as a math operator.

```

173 \cs_new_protected_nopar:Nn \_moremath_tex_mathop:n
174 {
175   \tex_mathop:D {#1}
176 }

```

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

4.2 L^AT_EX 2_ε-commands

`_moremath_matrix_star_begin:nn` The functions `_moremath_matrix_star_begin:nn` and `_moremath_matrix_star_end:n` provide an abstraction to the `matrix*` family of environments provided by `math-`
`_moremath_matrix_star_begin:nV` tools.
`_moremath_matrix_star_begin:Vn` `_moremath_matrix_star_begin:nn` takes two arguments:
`_moremath_matrix_star_begin:VV` **#1** : A $\langle t1 \rangle$ determining the type of the matrix.
`_moremath_matrix_star_end:n` This is used to further specify which `matrix*` environment to use.
`_moremath_matrix_star_end:V` The $\langle t1 \rangle$ may be empty.

#2 : A $\langle t1 \rangle$ specifying the column type.

```

177 \cs_new_nopar:Nn \_moremath_matrix_star_begin:nn
178 {
179   \begin{ #1 matrix* } [ #2 ]
180 }

```

We also create `nV`, `Vn`, and `VV` variants of this function:

```

181 \cs_generate_variant:Nn \_moremath_matrix_star_begin:nn { nV, Vn, VV }

```

The function `_moremath_matrix_star_end:n` takes only one argument
#1 : The $\langle t1 \rangle$ determining the type of the `matrix*` environment.
 The $\langle t1 \rangle$ **must** have the same value as used for `_moremath_matrix_star_begin:nn`.

```

182 \cs_new_nopar:Nn \_moremath_matrix_star_end:n
183 {
184   \end{ #1 matrix* }
185 }

```

We also provide a `V` type variant of this function.

```

186 \cs_generate_variant:Nn \_moremath_matrix_star_end:n {V}

```

(End of definition for `_moremath_matrix_star_begin:nn` and `_moremath_matrix_star_end:n`.)

5 Centering Math Symbols Along the Math-Axis

Certain math constructs such cause T_EX 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.

```

187 \cs_new_protected_nopar:Nn \moremath_vcenter:n
188 {
189   \__moremath_tex_mathpalette:Nn \__moremath_vcenter:Nn {#1}
190 }
```

(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.

```

191 \cs_new_protected_nopar:Nn \__moremath_vcenter:Nn
192 {
193   \__moremath_tex_vcenter:n
194   {
195     \hbox:n
196     {
197       $
198       \__moremath_tex_mathsurround:n {0pt}
199       #1 {#2}
200       $
201     }
202   }
203 }
```

(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.

```
204 \msg_new:nnn { moremath } { csname-already-defined }
205 {
206   Control-sequence-’ #1 ’~is-already~ defined.\\
207   Skipping-definition~\msg_line_context:
208 }
```

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

```
209 \msg_new:nnnn { moremath } { vcenter / only-in-math-mode }
210 {
211   Command-#1~used~outside~math~mode~\msg_line_context:.
212 }
213 {
214   The-command-#1~may~only~be~used~inside~math~mode.
215 }
```

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

```
216 \cs_if_free:NTF \VCenterMath
217 {
218   \NewDocumentCommand \VCenterMath { m }
219   {
220     \mode_if_math:TF
221     {
222       \moremath_vcenter:n {#1}
223     }{% \mode_if_math:TF FALSE BRANCH
224       \msg_error:nnn { moremath } { vcenter / only-in-math-mode } {\VCenterMath}
225     }
226   }
227 }{% \cs_if_free:NTF \VCenterMath FALSE BRANCH
228   \msg_warning:nnn { moremath } { csname-already-defined } {\VCenterMath}
229 }
```

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

6 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
230 \DeclarePairedDelimiter{\__moremath_inparent:w}{\lparen}{\rparen}
231 \DeclarePairedDelimiter{\__moremath_inbrack:w}{\lbrack}{\rbrack}
232 \DeclarePairedDelimiter{\__moremath_inbrace:w}{\lbrace}{\rbrace}
233 \DeclarePairedDelimiter{\__moremath_in_vert:w}{\lvert}{\rvert}
234 \DeclarePairedDelimiter{\__moremath_in_Vert:w}{\lVert}{\rVert}
```

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

7 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.

```
235 \cs_new_protected:Nn \_moremath_operator:Nnn
236 {
237   #1
238   % add superscript if present
239   \tl_if_empty:nF {#2} {^#{#2}}
240   % add subscript if present
241   \tl_if_empty:nF {#3} { \c_math_subscript_token {#3} }
242 }
```

(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` 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.

```
243 \cs_new_protected_nopar:Nn \moremath_delim_op_noscale:NNnnn
244 {
245   \_moremath_operator:Nnn #1 {#3} {#4}
246   % #2 is the paired delimiter
247   #2 {#5}
248 }
249
250 \cs_new_protected_nopar:Nn \moremath_delim_op_autoscale:NNnnn
251 {
252   \_moremath_operator:Nnn #1 {#3} {#4}
253   % #2 is the paired delimiter
254   #2 * {#5}
255 }
```

(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
```

`\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

256 \cs_new_protected_nopar:Nn \moremath_delim_op_manuscale:NNnnnn
257 {
258   \__moremath_operator:Nnn #1 {#4} {#5}
259   % #2 is the paired delimiter
260   #2 [ #3 ] {#6}
261 }

```

We also provide a variant for the scaling macro.

```

262 \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.

```

263 \msg_new:nnn { moremath } { delimop / already-defined-skip }
264 {
265   Control-sequence-' $\#1$ '-is-already-defined.\
266   Skipping-definition-of-delimited-operator-' $\#1$ '~\msg_line_context:.
267 }

```

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

```

268 \msg_new:nnn { moremath } { delimop / auto-manu-scale-conflict }
269 {
270   Both-star-and-scale-cmd-given-to-' $\#1$ '.\
271   Automatic-scaling-will-be-preferred,~the-size-command-will-be-
272   ignored-\msg_line_context:.
273 }

```

`\moremath_new_delim_op_command:NNN` \moremath_new_delim_op_command:NNN takes three arguments:

```

\moremath_new_delim_op_command:cNN
#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.

```

```

274 \cs_new_protected:Nn \moremath_new_delim_op_command:NNN
275 {
276   \cs_if_free:NTF #1
277   {
278     \exp_args:NNe \NewDocumentCommand #1
279     { s o E{ ^ \char_generate:nn {'_} {8} }{{}{}} m }
280     {
281       \tl_if_novalue:nTF {##2}
282       {

```

```

283 % second argument empty
284 \bool_if:nTF {##1}
285 {
286 % star given
287 \moremath_delim_op_autoscale:NNnnn #2 #3 {##3} {##4} {##5}
288 }{
289 % star not given
290 \moremath_delim_op_noscale:NNnnn #2 #3 {##3} {##4} {##5}
291 }
292 }{
293 % second argument present
294 % star given?
295 \bool_if:nTF {##1}
296 {
297 % Warn if both star and scaling factor are present
298 \msg_warning:nnn { moremath } { delimop / auto-manu-scale-conflict }
299 {#1}
300 \moremath_delim_op_autoscale:NNnnn #2 #3 {##3} {##4} {##5}
301 }{ % FALSE BRANCH
302 \moremath_delim_op_manuscale:NNnnnn #2 #3 {##2} {##3} {##4} {##5}
303 }
304 }
305 }
306 }{ % \cs_if_free:nTF #1 FALSE BRANCH
307 \msg_warning:nnn { moremath } { delimop / already-defined-skip }
308 {#1}
309 }
310 }
311
312 \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.)

7.1 Document Level Commands

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

```

313 \NewDocumentCommand\DeclareDelimitedOperator { m m m }
314 {
315 \msg_redirect_name:nnn { moremath } { delimop / already-defined-skip } { error }
316 \moremath_new_delim_op_command:NNN #1 #2 #3
317 \msg_redirect_name:nnn { moremath } { delimop / already-defined-skip } {}
318 }

```

(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>\varprojlim</code>	\varprojlim	<code>\varlimsup</code>	\varlimsup

`_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.

```

319 \cs_new_protected:Nn \_moremath_new_delim_op_cmds:nN
320 {
321   \moremath_new_delim_op_command:cNN {p #1} #2 \_moremath_inparent:w
322   \moremath_new_delim_op_command:cNN {b #1} #2 \_moremath_inbrack:w
323   \moremath_new_delim_op_command:cNN {B #1} #2 \_moremath_inbrace:w
324   \moremath_new_delim_op_command:cNN {v #1} #2 \_moremath_in_vert:w
325   \moremath_new_delim_op_command:cNN {V #1} #2 \_moremath_in_Vert:w
326 }

```

(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.

```

327 \bool_if:NTF \l_moremath_predef_operators_bool
328 {

```

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

```

\parccos For \arccos,
\barccos 329 \_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 330 \_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 331 \_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 332 \_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` 333 `_moremath_new_delim_op_cmds:nN {cos} \cos`
`\Bcos`
`\vcos` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vcos` `\cosh`,
`\pcosh`
`\bcosh` 334 `_moremath_new_delim_op_cmds:nN {cosh} \cosh`
`\Bcosh`
`\vcosh` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vcosh` `\cot`,
`\pcot`
`\bcot` 335 `_moremath_new_delim_op_cmds:nN {cot} \cot`
`\Bcot`
`\vcot` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vcot` `\coth`,
`\pcoth`
`\bcoth` 336 `_moremath_new_delim_op_cmds:nN {coth} \coth`
`\Bcoth`
`\vcoth` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vcoth` `\csc`,
`\pcsc`
`\bcsc` 337 `_moremath_new_delim_op_cmds:nN {csc} \csc`
`\Bcsc`
`\vcsc` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vcsc` `\deg`,
`\pdeg`
`\bdeg` 338 `_moremath_new_delim_op_cmds:nN {deg} \deg`
`\Bdeg`
`\vdeg` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vdeg` `\det`,
`\pdet`
`\bdet` 339 `_moremath_new_delim_op_cmds:nN {det} \det`
`\Bdet`
`\vdet` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vdet` `\dim`,
`\pdim`
`\bdim` 340 `_moremath_new_delim_op_cmds:nN {dim} \dim`
`\Bdim`
`\vdim` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vdim` `\exp`,
`\pexp`
`\bexp` 341 `_moremath_new_delim_op_cmds:nN {exp} \exp`
`\Bexp`
`\vexp` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vexp` `\gcd`,
`\pgcd`
`\bgcd` 342 `_moremath_new_delim_op_cmds:nN {gcd} \gcd`
`\Bgcd`
`\vgcd` (End of definition for `\parccos`. This function is documented on page 7.)
`\Vgcd` `\hom`,
`\phom`
`\bhom` 343 `_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 344 \_moremath_new_delim_op_cmds:nN {inf} \inf
\Binf
\vinf (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \injlim,
\binjlim 345 \_moremath_new_delim_op_cmds:nN {injlim} \injlim
\Binjlim
\vinjlim (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \ker,
\bker 346 \_moremath_new_delim_op_cmds:nN {ker} \ker
\Bker
\vker (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \lg,
\blg 347 \_moremath_new_delim_op_cmds:nN {lg} \lg
\Blg
\vlg (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \lim,
\blim 348 \_moremath_new_delim_op_cmds:nN {lim} \lim
\Blim
\vlim (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \liminf,
\bliminf 349 \_moremath_new_delim_op_cmds:nN {liminf} \liminf
\Bliminf
\vliminf (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \limsup,
\blimsup 350 \_moremath_new_delim_op_cmds:nN {limsup} \limsup
\Blimsup
\vlimsup (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \ln
\bln 351 \_moremath_new_delim_op_cmds:nN {ln} \ln
\Bln
\vln (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \log,
\blog 352 \_moremath_new_delim_op_cmds:nN {log} \log
\Blog
\vlog (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \max,
\bmax 353 \_moremath_new_delim_op_cmds:nN {max} \max
\Bmax
\vmax (End of definition for \parccos. This function is documented on page 7.)
\pinjlim \min,
\blmin 354 \_moremath_new_delim_op_cmds:nN {min} \min
\Blmin
\vmin (End of definition for \parccos. This function is documented on page 7.)
\pinjlim

```

```

\pPr \Pr,
\bPr 355 \_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 356 \_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 357 \_moremath_new_delim_op_cmds:nN {sec} \sec
\Bsec
\vsec (End of definition for \parccos. This function is documented on page 7.)
\Vsec \psin \sin,
\bsin 358 \_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 359 \_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 360 \_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 361 \_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 362 \_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 363 \_moremath_new_delim_op_cmds:nN {varinjlim} \varinjlim
\vvarinjlim 364 \_moremath_new_delim_op_cmds:nN {varprojlim} \varprojlim
\Vvarinjlim 365 \_moremath_new_delim_op_cmds:nN {varliminf} \varliminf
\pvarprojlim 366 \_moremath_new_delim_op_cmds:nN {varlimsup} \varlimsup
\bvarprojlim (End of definition for \parccos. This function is documented on page 7.)
\Bvarprojlim
\vvarprojlim 367 }{
\Vvarprojlim 368 \msg_info:nnnn {moremath} {load /disabling} {no-operators}
\pvarliminf 369 {
\bvarliminf 370 predefined-delimited-operator-macros
\Bvarliminf 371 }
\vvarliminf 372 } % End of the conditional
\Vvarliminf
\pvarlimsup
\bvarlimsup
\Bvarlimsup
\vvarlimsup
\Vvarlimsup

```

8 Vector Calculus Macros

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

8.1 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 $\langle tokens \rangle$

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

```

373 \cs_new_protected_nopar:Nn \__moremath_maybe_vcenter:n
374 {
375   \bool_if:NTF \l__moremath_vcenter_bool
376   {
377     \moremath_vcenter:n {#1}
378   }{
379     #1
380   }
381 }
```

(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.

```

382 \cs_new_protected:Nn \__moremath_gradient_operator_get:
383 {
384   \tl_if_empty:NTF \l__moremath_grad_op_tl
385   {
386     \__moremath_tex_mathop:n
387     {
```

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

```

388       \bool_if:NTF \l__moremath_nabla_arrow_bool
389       {
390         \vec
391         {
```

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

```

392           \__moremath_maybe_vcenter:n
393           {
394             \bool_if:NT \l__moremath_nabla_bold_bool
395             {
396               \boldsymbol
397             }
398             \l__moremath_nabla_tl
399           }
400         }
401       }{
```

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

```

402     \bool_if:NTF \l__moremath_nabla_bold_bool
403     {
404         \__moremath_maybe_vcenter:n
405         {
406             \boldsymbol
407             \l__moremath_nabla_tl
408         }
409     }{
410         \l__moremath_nabla_tl
411     }
412 }
413 }% \__moremath_tex_mathop:n
414 \nolimits
415 }{

```

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

```

416     \l__moremath_grad_op_tl
417 }
418 }

```

(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.

```

419 \cs_new_protected:Nn \__moremath_laplace_operator_get:
420 {
421     \tl_if_empty:NTF \l__moremath_laplacian_tl
422     {
423         \__moremath_tex_mathop:n
424         {
425             \bool_if:NTF \l__moremath_laplacian_delta_bool
426             {
427                 \Delta
428             }{
429                 \bool_if:NTF \l__moremath_laplacian_arrow_bool
430                 {
431                     \vec{
432                         \__moremath_maybe_vcenter:n
433                         {
434                             \bool_if:NT \l__moremath_nabla_bold_bool {\boldsymbol}
435                             \l__moremath_laplacian_symb_tl
436                         }
437                     }
438                 }{
439                     \bool_if:NTF \l__moremath_nabla_bold_bool
440                     {
441                         \__moremath_maybe_vcenter:n
442                         {
443                             \boldsymbol
444                             \l__moremath_laplacian_symb_tl
445                         }

```

```

446         }{
447         \l__moremath_laplacian_symb_tl
448         }
449     }
450 }
451 }\nolimits
452 \bool_if:NF \l__moremath_laplacian_delta_bool
453 {
454     \c_math_superscript_token
455     {
456         2
457     }
458 }
459 }{
460 \l__moremath_laplacian_tl
461 }
462 }

```

(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.

```

463 \cs_new_protected:Nn \__moremath_dalembert_operator_get:
464 {
465     \__moremath_tex_mathop:n
466     {
467         \l__moremath_dalembert_symb_tl
468     }% \__moremath_tex_mathop:n
469     \nolimits
470 }

```

(End of definition for `__moremath_dalembert_operator_get:`)

8.2 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.

```

471 \cs_new_protected_nopar:Nn \moremath_gradient_operator:n
472 {
473     \__moremath_gradient_operator_get:
474     \tl_if_empty:nF {#1} {\c_math_subscript_token {#1}}
475 }

```

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

```

476 \cs_new_protected_nopar:Nn \moremath_laplace_operator:n
477 {
478     \__moremath_laplace_operator_get:
479     \tl_if_empty:nF {#1} {\c_math_subscript_token {#1}}
480 }

```

(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.

```
481 \cs_new_protected_nopar:Nn \moremath_divergence_operator:n
482 {
```

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: $\nabla \cdot f(x)$

```
483 {
484   \moremath_gradient_operator:n {#1}
485 }
486 \cdot
487 }
488
489 \cs_new_protected_nopar:Nn \moremath_curl_operator:n
490 {
491   {
492     \moremath_gradient_operator:n {#1}
493   }
494   \times
495 }
```

(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.

```
496 \cs_new_protected:Nn \moremath_dalembert_operator:n
497 {
498   \__moremath_dalembert_operator_get:
499   \tl_if_empty:nF {#1}
500   {
501     \c_math_subscript_token {#1}
502   }
503 }
```

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

8.3 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.

```
504 \cs_new_protected_nopar:Nn \moremath_delim_nabla_op_noscale:NNnn
505 {
506   #1 {#3} #2{#4}
507 }
```

Then we create the version with automatic scaling.

```
508 \cs_new_protected_nopar:Nn \moremath_delim_nabla_op_autoscale:NNnn
509 {
510   #1 {#3} #2 * {#4}
511 }
```

(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.

```
512 \cs_new_protected_nopar:Nn \moremath_delim_nabla_op_manuscale:NNnnn
513 {
514   #1 {#4} #2 [#3] {#5}
515 }
```

`\moremath_delim_nabla_op_manuscale:NNVnn`

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

```
516 \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.

```
517 \cs_new_protected:Nn \__moremath_parse_kv_args:nN
518 {
```

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.

```
519 \prop_set_from_keyval:Nn \l_tmpa_prop {#1}
520
521 \prop_pop:NnNT \l_tmpa_prop {scale} #2 {}
522
523 \keys_set:ne {moremath} {\prop_to_keyval:N \l_tmpa_prop}
524 }
```

(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.

```
525 \msg_new:nnn { moremath } { vector-calc / scale-star-conflict }
526 {
527   Both~star~and~scaling~factor~given~to~'~#1'~.\\
528   Automatic~scaling~will~be~preferred,~the~size~command~'~#2'~will~be~
529   ignored~\msg_line_context:.
530 }
```

```
\__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.

```
531 \cs_new_protected:Nn \__moremath_new_delim_nabla_doc_cmd:NNN
532 {
533   \cs_if_free:NTF #1
534   {
535     \exp_args:NNe \NewDocumentCommand #1
536     {
537       s = {scale} o E { \char_generate:nn {'_'}{8} } { { } } m
538     }
539     { % command code
540       \group_begin:
541       % optional arguments given?
542       \tl_if_novalue:nF {##2}
543       {
544         \__moremath_parse_kv_args:nN {##2} \l_tmpa_tl
545       }
546       % star given?
547       \bool_if:nTF {##1}
548       {
```



```

549     % scale factor given?
550     \tl_if_empty:NF \l_tmpa_tl
551     {
552         \msg_warning:nnnV { moremath } { vector-calc / scale-star-conflict }
553         {#1} \l_tmpa_tl
554     }
555     \moremath_delim_nabla_op_autoscale:NNnn #2 #3 {##3} {##4}
556 }{ % \bool_if:nTF {##1} FALSE BRANCH
557     % scale factor given?
558     \tl_if_empty:NTF \l_tmpa_tl
559     {
560         \moremath_delim_nabla_op_noscale:NNnn #2 #3 {##3} {##4}
561     }{ % FALSE BRANCH
562         \moremath_delim_nabla_op_manuscale:NNVnn #2 #3 \l_tmpa_tl {##3} {##4}
563     }
564 } % \bool_if:nTF {##1}
565 \group_end:
566 }
567 }{ % \cs_if_free:NTF #1 FALSE BRANCH
568     \msg_warning:nnn { moremath } { vector-calc / already-defined-skip } {#1}
569 } % \cs_if_free:NTF #1
570 }
571 %
572 \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

```

573 \cs_new_protected:Nn \__moremath_new_nabla_doc_cmds:nN
574 {
575     \__moremath_new_delim_nabla_doc_cmd:cNN { p #1 } #2 \__moremath_inparent:w
576     \__moremath_new_delim_nabla_doc_cmd:cNN { b #1 } #2 \__moremath_inbrack:w
577     \__moremath_new_delim_nabla_doc_cmd:cNN { B #1 } #2 \__moremath_inbrace:w
578     \__moremath_new_delim_nabla_doc_cmd:cNN { v #1 } #2 \__moremath_in_vert:w
579     \__moremath_new_delim_nabla_doc_cmd:cNN { V #1 } #2 \__moremath_in_Vert:w
580 }

```

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

8.4 Document Level Commands

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

```

581 \bool_if:NTF \l__moremath_predef_vector_op_bool
582 {

```

8.4.1 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.

```

583 \msg_new:nnn { moremath } { vector-calc / already-defined-skip }
584 {
585   Control-sequence- $\#1$ '-is-already-defined.\
586   Skipping-definition-\msg_line_context:.
587 }

```

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

```

 $\text{\curl}$ 
 $\text{\laplacian}$ 
588 \cs_if_free:NTF \grad
589 {
590   \exp_args:NNe \NewDocumentCommand \grad { !o E{ \char_generate:nn {'_}{8} }{}} }
591   {
592     \group_begin:
593     \tl_if_novalue:nF {#1}
594     {
595       \keys_set:nn {moremath} {#1}
596     }
597     \moremath_gradient_operator:n {#2}
598     \group_end:
599   }
600 }{
601   \msg_warning:nnn {moremath} { vector-calc / already-defined-skip } {\grad}
602 }
603
604 \cs_if_free:NTF \divergence
605 {
606   \exp_args:NNe \NewDocumentCommand \divergence
607   { !o E{ \char_generate:nn {'_}{8} }{}} }
608   {
609     \group_begin:
610     \tl_if_novalue:nF {#1}
611     {
612       \keys_set:nn {moremath} {#1}
613     }
614     \moremath_divergence_operator:n {#2}
615     \group_end:
616   }
617 }{
618   \msg_warning:nnn {moremath} { vector-calc / already-defined-skip } {\divergence}
619 }
620
621 \cs_if_free:NTF \curl
622 {
623   \exp_args:NNe \NewDocumentCommand \curl
624   { !o E{ \char_generate:nn {'_}{8} }{}} }
625   {
626     \group_begin:

```

```

627 \tl_if_novalue:nF {#1}
628 {
629   \keys_set:nn {moremath} {#1}
630 }
631 \moremath_curl_operator:n {#2}
632 \group_end:
633 }
634 }{
635   \msg_warning:nnn {moremath} {vector-calc / already-defined-skip} {\curl}
636 }
637
638 \cs_if_free:NTF \laplacian
639 {
640   \exp_args:NNe \NewDocumentCommand \laplacian
641   { !o E{ \char_generate:nn {'_}{8} }{}} }
642   {
643     \group_begin:
644     \tl_if_novalue:nF {#1}
645     {
646       \keys_set:nn {moremath} {#1}
647     }
648     \moremath_laplace_operator:n {#2}
649     \group_end:
650   }
651 }{
652   \msg_warning:nnn {moremath} { vector-calc / already-defined-skip }
653   {\laplacian}
654 }

```

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`

```

655 \cs_if_free:NTF \quabla
656 {
657   \exp_args:NNe \NewDocumentCommand \quabla
658   { !o E{ \char_generate:nn {'_}{ 8 } }{}} }
659   {
660     \group_begin:
661     \tl_if_novalue:nF {#1}
662     {
663       \keys_set:nn { moremath } {#1}
664     }
665     \moremath_dalembert_operator:n {#2}
666     \group_end:
667   }
668 }{ % \cs_if_free:NTF \quabla FALSE BRANCH
669   \msg_warning:nnn { moremath } { vector-calc / already-defined-skip }
670   {\quabla}
671 }

```

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

8.4.2 Operators with Delimiters

`\pgrad` Now lets declare the delimited gradient operators. We provide five versions using parenthesis, brackets, braces, single `\vert`, and double `\Vert`.
`\bgrad`
`\Bgrad` 672 `_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` 673 `_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` Now to the curl macros.
`\pcurl`
`\bcurl` 674 `_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` Next we take care of the laplacian
`\plaplacian`
`\blaplacian` 675 `_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` Finally we define delimited commands for the d'Alembert operator.
`\pquabla`
`\bquabla` 676 `_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.

```
677 }{ % IF NOT \l__moremath_predef_vector_op_bool
678   \msg_info:nnnn {moremath} { load /disabling } {no-vector}
679   {
680     predefined~vector~calculus~macros
681   }
682 } % END \l__moremath_predef_vector_op_bool
```

9 Macros Producing Matrices and Vectors

9.1 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.

```
683 \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.

```
684 \cs_new_protected_nopar:Nn \__moremath_seq_to_column_vector:N
685 {
686   \seq_use:Nn #1 {\}
687 }
```

Then we get to the row vector.

```
688 \cs_new_protected_nopar:Nn \__moremath_seq_to_row_vector:N
689 {
690   \seq_use:Nn #1 {&}
691 }
```

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

`\moremath_column_vector:Vn`
`\moremath_row_vector:nn`
`\moremath_row_vector:Vn`

#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.

```
692 \cs_new_protected_nopar:Nn \moremath_column_vector:nn
693 {
694   \seq_clear:N \l__moremath_vector_entries_seq
695   \seq_set_from_clist:Nn \l__moremath_vector_entries_seq {#2}
696
697   \__moremath_matrix_star_begin:nV {#1} \l__moremath_matrix_align_tl
698   \__moremath_seq_to_column_vector:N \l__moremath_vector_entries_seq
699   \__moremath_matrix_star_end:n {#1}
700 }
701 %
702 \cs_new_protected_nopar:Nn \moremath_row_vector:nn
703 {
704   \seq_clear:N \l__moremath_vector_entries_seq
705   \seq_set_from_clist:Nn \l__moremath_vector_entries_seq {#2}
706
707   \__moremath_matrix_star_begin:nV {#1} \l__moremath_matrix_align_tl
708   \__moremath_seq_to_row_vector:N \l__moremath_vector_entries_seq
709   \__moremath_matrix_star_end:n {#1}
710 }
```

We also provide a `Vn` variant of those functions.

```
711 \cs_generate_variant:Nn \moremath_column_vector:nn { Vn }
712 \cs_generate_variant:Nn \moremath_row_vector:nn { Vn }
```

(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_column_smallvector:Vn`
`\moremath_row_smallvector:nn`
`\moremath_row_smallvector:Vn`

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.

```
713 \cs_new_protected_nopar:Nn \moremath_column_smallvector:nn
```

```

714 {
715   \moremath_column_vector:nn {#1 small} {#2}
716 }
717
718 \cs_new_protected_nopar:Nn \moremath_row_smallvector:nn
719 {
720   \moremath_row_vector:nn {#1 small} {#2}
721 }
722 % Variants
723 \cs_generate_variant:Nn \moremath_column_smallvector:nn { Vn }
724 \cs_generate_variant:Nn \moremath_row_smallvector:nn { Vn }

```

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

9.2 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`
`\l_moremath_mat_row_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

```

725 \seq_clear_new:N \l_moremath_mat_diag_entries_seq
726 \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`.)

9.2.1 (Anti-)diagonal matrices

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

`_moremath_constr_diagmat_row:n`
`_moremath_constr_antidiagmat_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.

```

727 \cs_new_protected:Nn \_moremath_constr_diagmat_row:n
728 {
729   \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.

```

730   \int_step_inline:nn {\seq_count:N \l_moremath_mat_diag_entries_seq}
731   {
732     \int_compare:nTF { #1 == ##1 }
733     {

```

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

```

734     \seq_put_right:Nx \l__moremath_mat_row_entries_seq
735     {
736       \seq_item:Nn \l__moremath_mat_diag_entries_seq { #1 }
737     }
738   }{ % false branch
739     \seq_put_right:NV \l__moremath_mat_row_entries_seq \l__moremath_matrix_fill_tl
740   } % \int_compare:nTF { #1 == ##1 }
741 }
742 \seq_use:Nn \l__moremath_mat_row_entries_seq { & } \\
743 }
744
745 % Anti-diagonal version
746 \cs_new_protected:Nn \__moremath_constr_antidiagmat_row:n
747 {
748   \seq_clear:N \l__moremath_mat_row_entries_seq
749   \int_step_inline:nn { \seq_count:N \l__moremath_mat_diag_entries_seq }
750   {
751     \int_compare:nTF { #1 == ##1 }
752     {
753       % as this is a anti diagonal matrix we put in the elements from the
754       % left so that the first entry is the right most entry
755       \seq_put_left:Nx \l__moremath_mat_row_entries_seq
756       {
757         \seq_item:Nn \l__moremath_mat_diag_entries_seq { #1 }
758       }
759     }{ % false branch
760       \seq_put_left:NV \l__moremath_mat_row_entries_seq \l__moremath_matrix_fill_tl
761     } % \int_compare:nTF { #1 == ##1 }
762   }
763   \seq_use:Nn \l__moremath_mat_row_entries_seq { & } \\
764 }

```

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

`\moremath_diagonal_matrix: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.
`\moremath_antidiagonal_matrix:nn` The functions take two arguments.
`\moremath_diagonal_smallmatrix:nn` #1 : The delimiter specifier.
`\moremath_antidiagonal_smallmatrix:nn` 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`.

```

765 \cs_new_protected:Nn \moremath_diagonal_matrix:nn
766 {
767   \seq_set_from_clist:Nn \l__moremath_mat_diag_entries_seq { #2 }
768   \__moremath_matrix_star_begin:nV { #1 } \l__moremath_matrix_align_tl
769   \int_step_function:nN { \seq_count:N \l__moremath_mat_diag_entries_seq }
770   \__moremath_constr_diagmat_row:n
771   \__moremath_matrix_star_end:n { #1 }
772 }
773 %
774 % Anti-diagonal matrix

```

```

775 \cs_new_protected:Nn \moremath_antidiagonal_matrix:nn
776 {
777   \seq_set_from_clist:Nn \l__moremath_mat_diag_entries_seq { #2 }
778   \__moremath_matrix_star_begin:nV { #1 } \l__moremath_matrix_align_tl
779     \int_step_function:nN { \seq_count:N \l__moremath_mat_diag_entries_seq }
780     \__moremath_constr_antidiagmat_row:n
781   \__moremath_matrix_star_end:n { #1 }
782 }
783
784 % Small versions
785 \cs_new_protected:Nn \moremath_diagonal_smallmatrix:nn
786 {
787   \moremath_diagonal_matrix:nn {#1 small} {#2}
788 }
789
790 \cs_new_protected:Nn \moremath_antidiagonal_smallmatrix:nn
791 {
792   \moremath_antidiagonal_matrix:nn {#1 small} {#2}
793 }

```

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

```

794 \cs_generate_variant:Nn \moremath_diagonal_matrix:nn { nV, Vn, VV }
795 \cs_generate_variant:Nn \moremath_antidiagonal_matrix:nn { nV, Vn, VV }
796 \cs_generate_variant:Nn \moremath_diagonal_smallmatrix:nn { nV, Vn, VV }
797 \cs_generate_variant:Nn \moremath_antidiagonal_smallmatrix:nn { nV, Vn, VV }

```

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

9.2.2 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.

The function `__moremath_generate_one_filled_clist:Nn` produces a $\langle clist \rangle$, consisting only of “1” as entries. This function takes two arguments:

- #1 : The *csname* of a $\langle clist var \rangle$ to store the $\langle clist \rangle$ into.
- #2 : An $\langle int \rangle$ to represent the number of entries to produce.

```

798 \cs_new_protected_nopar:Nn \__moremath_generate_one_filled_clist:Nn
799 {
800   \seq_clear:N \l_tmpa_seq
801   \int_step_inline:nn {#2}
802   {
803     \seq_put_right:NV \l_tmpa_seq \c_one_int
804   }
805   \clist_set_from_seq:NN #1 \l_tmpa_seq
806 }

```

We also define a variant accepting an integer variable.

```

807 \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.

```
808 \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.

```
809 \cs_new_protected_nopar:Nn \moremath_id_matrix:n
810 {
811   \clist_clear:N \l__moremath_id_entries_clist
812   \_moremath_generate_one_filled_clist:Nn \l__moremath_id_entries_clist {#1}
813   \moremath_diagonal_matrix:VV \l__moremath_matrix_delim_tl \l__moremath_id_entries_clist
814 }
815 \cs_new_protected_nopar:Nn \moremath_id_smallmatrix:n
816 {
817   \clist_clear:N \l__moremath_id_entries_clist
818   \_moremath_generate_one_filled_clist:Nn \l__moremath_id_entries_clist {#1}
819   \moremath_diagonal_smallmatrix:VV \l__moremath_matrix_delim_tl \l__moremath_id_entries_clist
820 }
```

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

```
821 \cs_generate_variant:Nn \moremath_id_matrix:n { V }
822 \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 26.)

9.3 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.

```
823 \msg_new:nnnn { moremath } { matrix / already-defined-doc-cmd-skip }
824 {
825   Control-sequence-’#1’-is-already-defined.\
826   Skipping-definition-\msg_line_context:.
827 }
828 {
829   The-control-sequence-’#1’-has-already\
830   been-defined-by-some-other-package.\
831   And-I-am-refusing-to-overwrite-the-existing-definition,\
832   therefore-I-am-skipping-the-definition-of-this-command.
833 }
```

9.3.1 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.

```
834 \bool_if:nTF \l_moremath_predef_crvector_bool
835 {
```

```

\cvector First we define the document level command for the bare column vector
\rvector
\smallcvector
\smallrvector
836 \cs_if_free:NTF \cvector
837 {
838   \NewDocumentCommand \cvector { o m }
839   {
840     \group_begin:
841     \tl_if_novalue:nF {#1}
842     {
843       \keys_set:nn { moremath / matrix } {#1}
844     }
845     \moremath_column_vector:Vn \l_moremath_matrix_delim_tl {#2}
846     \group_end:
847   }
848 }{
849   % issue a warning message if the csname is already taken.
850   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
851   {
852     \cvector
853   }
854 } % \cs_if_free:NTF \cvector
```

and the row vector.

```
855 \cs_if_free:NTF \rvector
856 {
857   \NewDocumentCommand \rvector { o m }
858   {
859     \group_begin:
860     \tl_if_novalue:nF {#1}
861     {
862       \keys_set:nn { moremath / matrix } {#1}
863     }
864     \moremath_row_vector:Vn \l_moremath_matrix_delim_tl {#2}
865     \group_end:
866   }
867 }{
868   % warn if csname is already taken
869   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip}
870   {
871     \rvector
872   }
873 } % \cs_if_free:NTF \rvector
```

Then we define the smaller inline versions of those commands.

```
874 \cs_if_free:NTF \smallcvector
875 {
876   \NewDocumentCommand \smallcvector { o m }
877   {
```

```

878 \group_begin:
879 \tl_if_novalue:nF {#1}
880 {
881   \keys_set:nn {moremath / matrix} {#1}
882 }
883 \moremath_column_smallvector:Vn \l__moremath_matrix_delim_tl {#2}
884 \group_end:
885 }
886 }{
887 % Issue a warning message if the csname is already taken
888 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
889 {
890   \smallcvector
891 }
892 } % \cs_if_free:NTF \smallcvector
893
894 \cs_if_free:NTF \smallrvector
895 {
896   \NewDocumentCommand \smallrvector { o m }
897   {
898     \group_begin:
899     \tl_if_novalue:nF {#1}
900     {
901       \keys_set:nn { moremath / matrix } {#1}
902     }
903     \moremath_row_smallvector:Vn \l__moremath_matrix_delim_tl {#2}
904     \group_end:
905   }
906 }{
907 % warn if csname is taken
908 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
909 {
910   \smallrvector
911 }
912 }

```

(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:Vn` like commands.
- #3** : The *⟨delimiter⟩* to use.
Usually one of `p, b, B, v, V`.

```

913 \cs_new_protected:Nn \__moremath_new_vector_shorth_doc_cmd:NNn
914 {
915   \cs_if_free:NTF #1
916   {

```

```

917 \NewDocumentCommand #1 { o m }
918 {
919   \group_begin:
920   % set the delimiter key pre-set for this function
921   \keys_set:nn {moremath / matrix } {delimiter = #3}
922   \tl_if_novalue:nF {##1}
923   {
924     \keys_set:nn {moremath / matrix } {##1}
925   }
926   #2 \l__moremath_matrix_delim_tl {##2}
927   \group_end:
928 }
929 }{
930 % warn if csname is taken
931 \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
932 {
933   #1
934 }
935 }
936 }

```

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

`\pcvector` 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.

```

937 % parenthesis
938 \__moremath_new_vector_shorth_doc_cmd:NNn \pcvector \moremath_column_vector:Vn {p}
939 % brackets
940 \__moremath_new_vector_shorth_doc_cmd:NNn \bcvector \moremath_column_vector:Vn {b}
941 % braces
942 \__moremath_new_vector_shorth_doc_cmd:NNn \Bcvector \moremath_column_vector:Vn {B}
943 % single vert
944 \__moremath_new_vector_shorth_doc_cmd:NNn \vcvector \moremath_column_vector:Vn {v}
945 % double vert
946 \__moremath_new_vector_shorth_doc_cmd:NNn \Vcvector \moremath_column_vector:Vn {V}

```

Now to the row vectors.

```

947 % parenthesis
948 \__moremath_new_vector_shorth_doc_cmd:NNn \prvector \moremath_row_vector:Vn {p}
949 % brackets
950 \__moremath_new_vector_shorth_doc_cmd:NNn \brvector \moremath_row_vector:Vn {b}
951 % braces
952 \__moremath_new_vector_shorth_doc_cmd:NNn \Brvector \moremath_row_vector:Vn {B}
953 % single vert
954 \__moremath_new_vector_shorth_doc_cmd:NNn \vrvector \moremath_row_vector:Vn {v}
955 % double vert
956 \__moremath_new_vector_shorth_doc_cmd:NNn \Vrvector \moremath_row_vector:Vn {V}

```

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

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

```

957 % column vectors
958 % parenthesis
959 \__moremath_new_vector_shorth_doc_cmd:NNn \psmallcvector \moremath_column_smallvector:Vn
\psmallrvector
\bsmallrvector
\Bsmallrvector
\vsmaillrvector
\Vsmallrvector

```

```

960 {p}
961 % brackets
962 \_moremath_new_vector_shorth_doc_cmd:NNn \bsmallcvector \moremath_column_smallvector:Vn
963 {b}
964 % braces
965 \_moremath_new_vector_shorth_doc_cmd:NNn \Bsmallcvector \moremath_column_smallvector:Vn
966 {B}
967 % single vert
968 \_moremath_new_vector_shorth_doc_cmd:NNn \vsmallcvector \moremath_column_smallvector:Vn
969 {v}
970 % double vert
971 \_moremath_new_vector_shorth_doc_cmd:NNn \Vsmallcvector \moremath_column_smallvector:Vn
972 {V}
973 %
974 % row vectors
975 % parenthesis
976 \_moremath_new_vector_shorth_doc_cmd:NNn \psmallrvector \moremath_row_smallvector:Vn {p}
977 % brackets
978 \_moremath_new_vector_shorth_doc_cmd:NNn \bsmallrvector \moremath_row_smallvector:Vn {b}
979 % braces
980 \_moremath_new_vector_shorth_doc_cmd:NNn \Bsmallrvector \moremath_row_smallvector:Vn {B}
981 % single vert
982 \_moremath_new_vector_shorth_doc_cmd:NNn \vsmallrvector \moremath_row_smallvector:Vn {v}
983 % double vert
984 \_moremath_new_vector_shorth_doc_cmd:NNn \Vsmallrvector \moremath_row_smallvector:Vn {V}
985
986
987 }{ % \bool_if:nTF \l_moremath_predef_crvector_bool FALSE PATH
988 \msg_info:nnnn {moremath} {load / disabling} {no-crvector}
989 {
990   commands~producing~row~and~column~vectors
991 }
992 } % \bool_if:nTF \l_moremath_predef_crvector_bool

```

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

9.3.2 (Anti-)diagonal Matrices

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

```

993 \bool_if:nTF \l_moremath_predef_matrix_bool
994 {
995
996   \diagmat
997   \antidiagmat
998   \smalldiagmat
999   \smallantidiagmat
1000
1001   \cs_if_free:NTF \diagmat
1002   {
1003     \NewDocumentCommand \diagmat { o m }
1004     {
1005       \group_begin:
1006       \tl_if_novalue:nF {#1}
1007       {
1008         \keys_set:nn { moremath / matrix } {#1}
1009       }
1010       \moremath_diagonal_matrix:Vn \l_moremath_matrix_delim_tl {#2}
1011     }
1012   }

```

```

1005     \group_end:
1006   }
1007 }{
1008   \msg_warning:nnn {moremath} {matrix / already-defined-doc-cmd-skip}
1009   {
1010     \diagmat
1011   }
1012 } % \cs_if_free:nTF \diagmat
1013
1014 \cs_if_free:NTF \antidiagmat
1015 {
1016   \NewDocumentCommand \antidiagmat { o m }
1017   {
1018     \group_begin:
1019     \tl_if_novalue:nF {#1}
1020     {
1021       \keys_set:nn { moremath / matrix } {#1}
1022     }
1023     \moremath_antidiagonal_matrix:Vn \l__moremath_matrix_delim_tl {#2}
1024     \group_end:
1025   }
1026 }{
1027   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1028   {
1029     \antidiagmat
1030   }
1031 } % \cs_if_free:nTF \antidiagmat
1032
1033 \cs_if_free:NTF \smalldiagmat
1034 {
1035   \NewDocumentCommand \smalldiagmat { o m }
1036   {
1037     \group_begin:
1038     \tl_if_novalue:nF {#1}
1039     {
1040       \keys_set:nn { moremath / matrix } {#1}
1041     }
1042     \moremath_diagonal_smallmatrix:Vn \l__moremath_matrix_delim_tl {#2}
1043     \group_end:
1044   }
1045 }{
1046   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1047   {
1048     \smalldiagmat
1049   }
1050 }
1051
1052 \cs_if_free:NTF \smallantidiagmat
1053 {
1054   \NewDocumentCommand \smallantidiagmat { o m }
1055   {
1056     \group_begin:
1057     \tl_if_novalue:nF {#1}
1058     {

```

```

1059     \keys_set:nn { moremath / matrix } {#1}
1060   }
1061   \moremath_antidiagonal_smallmatrix:Vn \l__moremath_matrix_delim_tl {#2}
1062   \group_end:
1063 }
1064 }{
1065   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1066   {
1067     \smallantidiagmat
1068   }
1069 }

```

(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

```

1070 \cs_new_protected:Nn \__moremath_new_matrix_shorth_doc_cmd:NNn
1071 {
1072   \cs_if_free:NTF #1
1073   {
1074     \NewDocumentCommand #1 { o m }
1075     {
1076       \group_begin:
1077       \tl_if_empty:nF {#3}
1078       {
1079         \keys_set:nn { moremath / matrix }
1080         {
1081           delimiter = #3
1082         }
1083       } % \tl_if_empty:nF {#3}
1084       \tl_if_novalue:nF {##1}
1085       {
1086         \keys_set:nn { moremath / matrix } {##1}
1087       }
1088       #2 \l__moremath_matrix_delim_tl {##2}
1089       \group_end:
1090     }
1091   }{
1092     \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1093     {
1094       #1
1095     }
1096   }
1097 }

```

(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 1098 \_moremath_new_matrix_shorth_doc_cmd:NNn \pdiagmat \moremath_diagonal_matrix:Vn {p}
\Bdiagmat 1099 \_moremath_new_matrix_shorth_doc_cmd:NNn \bdiagmat \moremath_diagonal_matrix:Vn {b}
\vdiagmat 1100 \_moremath_new_matrix_shorth_doc_cmd:NNn \Bdiagmat \moremath_diagonal_matrix:Vn {B}
\Vdiagmat 1101 \_moremath_new_matrix_shorth_doc_cmd:NNn \vdiagmat \moremath_diagonal_matrix:Vn {v}
1102 \_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 1103 \_moremath_new_matrix_shorth_doc_cmd:NNn \pantidiagmat
\Bantidiagmat 1104 \_moremath_antidiagonal_matrix:Vn {p}
\vantidiagmat 1105 \_moremath_new_matrix_shorth_doc_cmd:NNn \bantidiagmat
\Vantidiagmat 1106 \moremath_antidiagonal_matrix:Vn {b}
1107 \_moremath_new_matrix_shorth_doc_cmd:NNn \Bantidiagmat
1108 \moremath_antidiagonal_matrix:Vn {B}
1109 \_moremath_new_matrix_shorth_doc_cmd:NNn \vantidiagmat
1110 \moremath_antidiagonal_matrix:Vn {v}
1111 \_moremath_new_matrix_shorth_doc_cmd:NNn \Vantidiagmat
1112 \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 1113 \_moremath_new_matrix_shorth_doc_cmd:NNn \psmalldiagmat
\Bsmalldiagmat 1114 \moremath_diagonal_smallmatrix:Vn {p}
\vsmalldiagmat 1115 \_moremath_new_matrix_shorth_doc_cmd:NNn \bsmalldiagmat
\Vsmalldiagmat 1116 \moremath_diagonal_smallmatrix:Vn {b}
1117 \_moremath_new_matrix_shorth_doc_cmd:NNn \Bsmalldiagmat
1118 \moremath_diagonal_smallmatrix:Vn {B}
1119 \_moremath_new_matrix_shorth_doc_cmd:NNn \vsmalldiagmat
1120 \moremath_diagonal_smallmatrix:Vn {v}
1121 \_moremath_new_matrix_shorth_doc_cmd:NNn \Vsmalldiagmat
1122 \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 1123 \_moremath_new_matrix_shorth_doc_cmd:NNn \psmallantidiagmat
\Bsmallantidiagmat 1124 \moremath_antidiagonal_smallmatrix:Vn {p}
\vsmallantidiagmat 1125 \_moremath_new_matrix_shorth_doc_cmd:NNn \bsmallantidiagmat
\Vsmallantidiagmat 1126 \moremath_antidiagonal_smallmatrix:Vn {b}
1127 \_moremath_new_matrix_shorth_doc_cmd:NNn \Bsmallantidiagmat
1128 \moremath_antidiagonal_smallmatrix:Vn {B}
1129 \_moremath_new_matrix_shorth_doc_cmd:NNn \vsmallantidiagmat
1130 \moremath_antidiagonal_smallmatrix:Vn {v}
1131 \_moremath_new_matrix_shorth_doc_cmd:NNn \Vsmallantidiagmat
1132 \moremath_antidiagonal_smallmatrix:Vn {V}
```

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

9.3.3 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.

```

1133 \cs_if_free:NTF \idmat
1134 {
1135   \NewDocumentCommand \idmat { o m }
1136   {
1137     \group_begin:
1138     \tl_if_novalue:nF {#1}
1139     {
1140       \keys_set:nn { moremath / matrix } {#1}
1141     }
1142     \moremath_id_matrix:n {#2}
1143     \group_end:
1144   }
1145 }{ % \cs_if_free:NTF \idmat FALSE BRANCH
1146   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1147   {\idmat}
1148 }
```

Afterwards we continue with the inline math mode version.

```

1149 \cs_if_free:NTF \smallidmat
1150 {
1151   \NewDocumentCommand \smallidmat { o m }
1152   {
1153     \group_begin:
1154     \tl_if_novalue:nF {#1}
1155     {
1156       \keys_set:nn { moremath / matrix } {#1}
1157     }
1158     \moremath_id_smallmatrix:n {#2}
1159     \group_end:
1160   }
1161 }{ % \cs_if_free:NTF \smallidmat FALSE BRANCH
1162   \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1163   {\smallidmat}
1164 }
```

(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`.

```

1165 \cs_new_protected:Nn \__moremath_new_id_matrix_doc_cmd:NNn
1166 {
1167   \cs_if_free:NTF #1
1168   {
1169     \NewDocumentCommand #1 { o m }
1170     {
1171       \group_begin:
1172       \tl_if_empty:nF {#3}
1173       {
1174         \keys_set:nn { moremath / matrix } {#3}
1175       }
1176       \tl_if_novalue:nF {##1}
1177       {
1178         \keys_set:nn { moremath / matrix } {##1}
1179       }
1180       #2 {##2}
1181       \group_end:
1182     }
1183   }{ % \cs_if_free:NTF #1 FALSE BRANCH
1184     \msg_warning:nnn { moremath } { matrix / already-defined-doc-cmd-skip }
1185     {#1}
1186   }
1187 }

```

(End of definition for $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$.)

$\backslash\text{pidmat}$ We begin with the display math versions, starting with the version delimited by parenthesis,
 $\backslash\text{bidmat}$
 $\backslash\text{Bidmat}$ 1188 $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$ $\backslash\text{pidmat}$ $\backslash\text{moremath_id_matrix:n}$ { delimiter = p }
 $\backslash\text{vidmat}$ continue with the bracketed version,
 $\backslash\text{Vidmat}$ 1189 $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$ $\backslash\text{bidmat}$ $\backslash\text{moremath_id_matrix:n}$ { delimiter = b }
the version using braces,
1190 $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$ $\backslash\text{Bidmat}$ $\backslash\text{moremath_id_matrix:n}$ { delimiter = B }
single vertical lines,
1191 $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$ $\backslash\text{vidmat}$ $\backslash\text{moremath_id_matrix:n}$ { delimiter = v }
and finally double vertical lines.
1192 $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$ $\backslash\text{Vidmat}$ $\backslash\text{moremath_id_matrix:n}$ { 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,
 $\backslash\text{Bsmallidmat}$ 1193 $\backslash\text{__moremath_new_id_matrix_doc_cmd:NNn}$ $\backslash\text{psmallidmat}$ $\backslash\text{moremath_id_smallmatrix:n}$
 $\backslash\text{vsmallidmat}$ 1194 { delimiter = p }
 $\backslash\text{Vsmallidmat}$

then brackets,

```
1195 \_moremath_new_id_matrix_doc_cmd:NNn \bsmallidmat \moremath_id_smallmatrix:n
1196 { delimiter = b }
```

then braces,

```
1197 \_moremath_new_id_matrix_doc_cmd:NNn \Bsmallidmat \moremath_id_smallmatrix:n
1198 { delimiter = B }
```

followed by single vertical lines,

```
1199 \_moremath_new_id_matrix_doc_cmd:NNn \vsmallidmat \moremath_id_smallmatrix:n
1200 { delimiter = v }
```

and finally double vertical lines.

```
1201 \_moremath_new_id_matrix_doc_cmd:NNn \Vsmallidmat \moremath_id_smallmatrix:n
1202 { delimiter = V }
```

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

```
1203 }{ % \bool_if:nTF \l_moremath_predef_matrix_bool FALSE BRANCH
1204 \msg_info:nnnn {moremath} { load / disabling } { no-matrix }
1205 {
1206 (anti-)diagonal~matrix~commands
1207 }
1208 } % \bool_if:nTF \l_moremath_predef_matrix_bool
```

10 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.

```
1209 \msg_new:nnnn { moremath } { abs-shorth / csname-already-defined-skip }
1210 {
1211 Control~sequence~'#1'~is~already~defined.\
1212 Skipping~declaration~of~paired~delimiter~\msg_line_context:\
1213 Use~package~option~'no-abs-shorthands'~to~disable~the~paired\
1214 delimiter~shorthands.
1215 }{
1216 The~control~sequence~'#1'~has~already~been\
1217 defined~by~something~else.\
1218 I~am~refusing~to~overwrite~its~existing~definition~and~instead~avoid\
1219 declaring~a~paired~delimiter.\
1220 }
```

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

```
1221 \bool_if:NTF \l_moremath_predef_abs_bool
1222 {
```

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

```
\norm
1223 \cs_if_free:NTF \abs
1224 {
1225 \DeclarePairedDelimiter \abs {\lvert} {\rvert}
1226 }{
1227 % warn if the csname is taken
1228 \msg_warning:nnn { moremath } { abs-shorth / csname-already-defined-skip }
1229 {\abs}
1230 } % \cs_if_free:NTF \abs
```

```

1231
1232 \cs_if_free:NTF \norm
1233 {
1234   \DeclarePairedDelimiter \norm {\lVert} {\rVert}
1235 }{
1236   % warn if csname is already taken
1237   \msg_warning:nnn { moremath } { abs-shorth / csname-already-defined-skip }
1238   {\norm}
1239 } % \cs_if_free:NTF

```

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

```

1240 }{
1241   \msg_info:nnnn {moremath} {load / disabling} {no-abs-shorthands}
1242   {
1243     '\abs'~and~'\norm'~macros
1244   }
1245 } % End of the conditional
1246 </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).
- [Mit18] Frank Mittelbach. “A rollback concept for packages and classes.” In: *TUGboat* 2 (2018). URL: <https://www.latex-project.org/publications/2018-FMi-TUB-tb122mitt-version-rollback.pdf> (visited on 08/06/2024).

- [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
align: Added options for the matrix-based environments of mathtools.	31
dalembert-symb: Added options controlling appearance.	30
\DeclareDelimitedOperator: Added document level command.	38
\laplacian: Added \grad, \divergence, \curl and \laplacian commands.	51
\moremath_antidiagonal_smallmatrix:VV: Added variants.	56
\moremath_antidiagonal_smallmatrix:nn: Added functions producing (anti-)diagonal matrices. Also added versions for inline math.	56
\moremath_curl_operator:n: Added functions.	46
\moremath_delim_nabla_op_autoscale:NNnn: Added noscale and autoscale functions.	47
\moremath_delim_nabla_op_manuscale:NNVnn: Add variant.	47
\moremath_delim_nabla_op_manuscale:NNnnn: Add function.	47
\moremath_delim_op_autoscale:NNnnn: Added noscale and autoscale versions.	36
\moremath_delim_op_manuscale:NNVnnn: Function added.	37
\moremath_laplace_operator:n: Added functions.	45
\l_moremath_matrix_align_tl: Added internal variables.	31
\moremath_new_delim_op_command:cNN: Added function and variant.	38
\moremath_row_smallvector:Vn: Added functions producing inline math row- and column vectors.	54
\moremath_row_vector:Vn: Added functions producing row- and column vectors.	53
\moremath_setup:n: Added function for setting options.	31
\moremathsetup: Added document command for setting package options.	31
nopredef: Added package load-time options.	29
\norm: Added shorthands for absolute value and norm.	68
\smallantidiagmat: Added commands producing (anti-)diagonal matrices, including inline math versions.	63
\smallrvector: Added commands producing row and column vectors, including inline math versions.	59
\Vantidiagmat: Added commands for delimited antidiagonal matrices.	64
\Varccos: Added delimited document commands for all amsmath-defined operators.	39
\Vcurl: Added commands for delimited curl operators.	52
\Vdiagmat: Added commands for delimited diagonal matrices.	64
\Vdiv: Added commands for delimited divergence operators.	52
\Vgrad: Added commands for delimited gradient operators.	52
\Vlaplacian: Added commands for delimited Laplace operators.	52
\Vrvector: Added commands for row and column vectors with predefined delimiters.	60
\Vsmallantidiagmat: Added commands for delimited	

anti-diagonal matrices suitable for inline math.	64		
<code>\Vsmallldiagmat</code> : Added commands for delimited diagonal matrices suitable for inline math.	64		
<code>\Vsmallrvector</code> : Added commands for inline math row and column vectors with predefined delimiters.	61		
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.	32		
<code>dalembert-symb</code> : Added: New option <code>dalembert-symb</code>	30		
<code>\moremath_antidiagonal_smallmatrix:VV</code> : Added <code>nV</code> and <code>VV</code> variants.	56		
<code>\moremath_dalembert_operator:n</code> : Added new function <code>\moremath_dalembert_operator:n</code>	46		
<code>_moremath_dalembert_operator_get::</code> : Added function: <code>_moremath_dalembert_operator_get:</code>	45		
<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.	29		
<code>_moremath_generate_one_filled_clist:Nn</code> : Added function.	56		
<code>\l_moremath_id_entries_clist</code> : Added variable.	57		
<code>\moremath_id_smallmatrix:n</code> : Added functions <code>\moremath_id_matrix:n</code> and <code>\moremath_id_smallmatrix:n</code> and variants.	57		
<code>\l_moremath_predef_abs_bool</code> : Explicitly declared boolean variables used to store key-value options.	27		
<code>\quabla</code> : Added <code>\quabla</code> command.	51		
<code>\smallldmat</code> : Added <code>\idmat</code> and <code>\smallldmat</code> document level commands.	65		
<code>\Vidmat</code> : Added document level commands <code>\pidmat</code> , <code>\bidmat</code> , <code>\Bidmat</code> , <code>\vidmat</code> , and <code>\Vidmat</code>	66		
<code>\Vquabla</code> : Added commands for a delimited d'Alembert operator.	52		
		<code>\Vsmallldmat</code> : Added document level commands <code>\psmallldmat</code> , <code>\bsmallldmat</code> , <code>\vsmallldmat</code> and <code>\Vsmallldmat</code>	67
		v0.3.0 – 2024-07-08	
		General: Split the documentation into several parts.	1
		<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>	51
		<code>\moremath_vcenter:n</code> : Added new function.	34
		<code>\quabla</code> : Changed: Spaces between the <code><csname></code> and the optional argument are now disallowed.	51
		<code>vcenter</code> : Added new option.	30
		<code>\VCenterMath</code> : Added new document level command.	35
		v0.4.0 – 2024-07-15	
		General: Breaking Change : Rename the package and the <code><prefix></code> used for function names from <code>conmath</code> to <code>moremath</code>	1
		v0.5.0 – 2024-08-20	
		General: Add information for L ^A T _E X's package rollback mechanism.	27
		<code>_moremath_dalembert_operator_get::</code> : Now uses <code>_moremath_tex_mathop:n</code> instead of T _E X-primitives.	45
		<code>_moremath_gradient_operator_get::</code> : Now uses <code>_moremath_tex_mathop:n</code> instead of T _E X-primitives.	44
		<code>_moremath_laplace_operator_get::</code> : Now uses <code>_moremath_tex_mathop:n</code> instead of T _E X-primitives.	45
		<code>_moremath_matrix_star_end:V</code> : Added new functions and variants.	33
		<code>_moremath_new_vector_shorth_doc_cmd:NNn</code> : Require function with <code>Vn</code> signature to pass variable correctly.	60
		<code>\moremath_row_smallvector:Vn</code> : Added new <code>Vn</code> variants.	54
		<code>\moremath_row_vector:Vn</code> : Added new <code>Vn</code> variants.	53
		<code>_moremath_tex_mathop:n</code> : Added new function.	33
		<code>_moremath_tex_mathpalette:Nn</code> : Added new function.	32

<code>_moremath_tex_mathsurround:n</code> Added new function.	33	instead of \TeX -primitives.	34
<code>\moremath_vcenter:n</code> : Changed: Make use of <code>_moremath_tex_- mathpalette:Nn</code>	34	<code>\smallvector</code> : Use \mathbf{Vn} versions of functions when passing variables around.	59
<code>_moremath_vcenter:Nn</code> : Changed: Now uses <code>_moremath_tex_vcenter:n</code> and <code>_moremath_tex_mathsurround:n</code>		<code>\Rrvector</code> : Use functions with \mathbf{Vn} signature instead of <code>nn</code>	60
		<code>\Vsmallvector</code> : Use functions with \mathbf{Vn} signature instead of <code>nn</code>	61

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>	9, 14, 152, 206, 265, 270, 527, 585, 686, 742, 763, 825, 829, 830, 831, 1211, 1212, 1213, 1216, 1217, 1218, 1219		
A			
<code>\abs</code>	22, 1223, 1243		
<code>align</code> (option)	6, 127		
amsmath commands:			
<code>\arccos</code>	7, 39	<code>\Pr</code>	7, 39, 42
<code>\arcsin</code>	7, 39	<code>\projlim</code>	7, 39, 42
<code>\arctan</code>	7, 39	<code>\sec</code>	7, 39, 42
<code>\arg</code>	7, 39	<code>\sin</code>	7, 39, 42
<code>\cos</code>	7, 39, 40	<code>\sinh</code>	7, 39, 42
<code>\cosh</code>	7, 39, 40	<code>\sup</code>	7, 39, 42
<code>\cot</code>	7, 39, 40	<code>\tan</code>	7, 39, 42
<code>\coth</code>	7, 39, 40	<code>\tanh</code>	7, 39, 42
<code>\csc</code>	7, 39, 40	<code>\varinjlim</code>	7, 39, 42
<code>\DeclareMathOperator</code>	8, 23, 36, 37	<code>\varliminf</code>	7, 39, 42
<code>\deg</code>	7, 39, 40	<code>\varlimsup</code>	7, 39, 42
<code>\det</code>	7, 39, 40	<code>\varprojlim</code>	7, 39, 42
<code>\dim</code>	7, 39, 40	amssymb commands:	
<code>\exp</code>	7, 39, 40	<code>\square</code>	6, 32
<code>\gcd</code>	7, 39, 40	<code>\antidiagmat</code>	18, 995
<code>\hom</code>	7, 39, 40	<code>\arccos</code>	329
<code>\inf</code>	7, 39, 41	<code>\arcsin</code>	330
<code>\injlim</code>	7, 39, 41	<code>\arctan</code>	331
<code>\ker</code>	7, 39, 41	<code>\arg</code>	332
<code>\lg</code>	7, 39, 41	<code>arrowlaplace</code> (option)	6, 91
<code>\lim</code>	7, 39, 41	<code>arrownabla</code> (option)	5, 91
<code>\liminf</code>	7, 39, 41		
<code>\limsup</code>	7, 39, 41	B	
<code>\ln</code>	7, 39, 41	<code>\Bantidiagmat</code>	19, 1103
<code>\log</code>	7, 39, 41	<code>\bantidiagmat</code>	19, 1103
<code>\max</code>	7, 39, 41	<code>\Barccos</code>	7, 329
<code>\min</code>	7, 39, 41	<code>\barccos</code>	7, 329
		<code>\Barcsin</code>	330
		<code>\barcsin</code>	330
		<code>\Barctan</code>	331
		<code>\barctan</code>	331
		<code>\Barg</code>	332
		<code>\barg</code>	332
		<code>\Bcos</code>	333
		<code>\bcos</code>	333
		<code>\Bcosh</code>	334
		<code>\bcosh</code>	334

<code>\bcosh</code>	334	<code>\Blog</code>	352
<code>\bcot</code>	335	<code>\blog</code>	352
<code>\bcot</code>	335	<code>bm</code> (option)	5, 22
<code>\Bcoth</code>	336	<code>\Bmax</code>	353
<code>\bcoth</code>	336	<code>\bmax</code>	353
<code>\Bcsc</code>	337	<code>\Bmin</code>	354
<code>\bcsc</code>	337	<code>\bmin</code>	354
<code>\Bcurl</code>	13, 674	<code>boldnabla</code> (option)	5, 91
<code>\bcurl</code>	13, 674	<code>\boldsymbol</code>	5, 396, 406, 434, 443
<code>\Bcvector</code>	16, 937	bool commands:	
<code>\bcvector</code>	16, 937	<code>\bool_if:NTF</code>	155, 327, 375, 388, 394, 402, 425, 429, 434, 439, 452, 581, 1221
<code>\Bdeg</code>	338	<code>\bool_if:nTF</code>	284, 295, 547, 556, 564, 834, 987, 992, 993, 1203, 1208
<code>\bdeg</code>	338	<code>\bool_new:N</code>	17, 18, 19, 20, 21, 82, 83, 86, 87, 90
<code>\Bdet</code>	339	<code>\BPr</code>	355
<code>\bdet</code>	339	<code>\bPr</code>	355
<code>\Bdiagmat</code>	19, 1098	<code>\Bprojlim</code>	356
<code>\bdiagmat</code>	19, 1098	<code>\bprojlim</code>	356
<code>\Bdim</code>	340	<code>\Bquabla</code>	15, 676
<code>\bdim</code>	340	<code>\bquabla</code>	15, 676
<code>\Bdiv</code>	12, 673	<code>\Bvector</code>	16, 937
<code>\bdiv</code>	12, 673	<code>\bvector</code>	16, 937
<code>\begin</code>	179	<code>\Bsec</code>	357
<code>\Bexp</code>	341	<code>\bsec</code>	357
<code>\bexp</code>	341	<code>\Bsin</code>	358
<code>\Bgcd</code>	342	<code>\bsin</code>	358
<code>\bgcd</code>	342	<code>\Bsinh</code>	359
<code>\Bgrad</code>	10, 672	<code>\bsinh</code>	359
<code>\bgrad</code>	10, 672	<code>\Bsmallantidiagmat</code>	20, 1123
<code>\Bhom</code>	343	<code>\bsmallantidiagmat</code>	20, 1123
<code>\bhom</code>	343	<code>\Bsmallcvector</code>	17, 957
<code>\Bidmat</code>	21, 70, 1188	<code>\bsmallcvector</code>	17, 957
<code>\bidmat</code>	21, 70, 1188	<code>\Bsmallldiagmat</code>	20, 1113
<code>\Big</code>	7, 10, 23, 24, 37, 47	<code>\bsmallldiagmat</code>	20, 1113
<code>\big</code>	7, 10, 23, 24, 37, 47	<code>\Bsmallidmat</code>	21, 70, 1193
<code>\Bigg</code>	7, 10, 23, 37	<code>\bsmallidmat</code>	21, 70, 1193
<code>\bigg</code>	7, 10, 23, 47	<code>\Bsmallrvector</code>	17, 957
<code>\Binf</code>	344	<code>\bsmallrvector</code>	17, 957
<code>\binf</code>	344	<code>\Bsup</code>	360
<code>\Binjlim</code>	345	<code>\bsup</code>	360
<code>\binjlim</code>	345	<code>\Btan</code>	361
<code>\Bker</code>	346	<code>\btan</code>	361
<code>\bker</code>	346	<code>\Btanh</code>	362
<code>\Blaplacian</code>	14, 675	<code>\btanh</code>	362
<code>\blaplacian</code>	14, 675	<code>\Bvarinjlim</code>	363
<code>\Blg</code>	347	<code>\bvarinjlim</code>	363
<code>\blg</code>	347	<code>\Bvarliminf</code>	363
<code>\Blim</code>	348	<code>\bvarliminf</code>	363
<code>\blim</code>	348	<code>\Bvarlimsup</code>	363
<code>\Bliminf</code>	349	<code>\bvarlimsup</code>	363
<code>\bliminf</code>	349	<code>\Bvarprojlim</code>	363
<code>\Blimsup</code>	350	<code>\bvarprojlim</code>	363
<code>\blimsup</code>	350		
<code>\Bln</code>	351		
<code>\bln</code>	351		

C		<code>\divergence</code> 11 , 14 , 69 , 70 , 588
<code>\cdot</code> 486		
char commands:		E
<code>\char_generate:nn</code>		<code>\end</code> 184
. 279 , 537 , 590 , 607 , 624 , 641 , 658		<code>\exp</code> 341
clist commands:		exp commands:
<code>\clist_clear:N</code> 811 , 817		<code>\exp_args:NNe</code>
<code>\clist_new:N</code> 808	 278 , 535 , 590 , 606 , 623 , 640 , 657
<code>\clist_set_from_seq:NN</code> 805		F
<code>\cos</code> 333		fill (option) 6 , 127
<code>\cosh</code> 334		G
<code>\cot</code> 335		<code>\gcd</code> 342
<code>\coth</code> 336		<code>\grad</code> 9 , 11 , 12 , 14 , 69 , 70 , 588
cs commands:		grad-op (option) 6 , 91
<code>\cs_generate_variant:Nn</code> 181 , 186 ,		group commands:
262 , 312 , 516 , 572 , 711 , 712 , 723 ,		<code>\group_begin:</code>
724 , 794 , 795 , 796 , 797 , 807 , 821 , 822	 26 , 540 , 592 , 609 , 626 , 643 ,
<code>\cs_if_free:NTF</code> 216 ,		660 , 840 , 859 , 878 , 898 , 919 , 999 ,
621 , 638 , 655 , 668 , 836 , 854 , 855 ,		1018 , 1037 , 1056 , 1076 , 1137 , 1153 , 1171
873 , 874 , 892 , 894 , 915 , 995 , 1014 ,		<code>\group_end:</code>
1033 , 1052 , 1072 , 1133 , 1145 , 1149 ,	 26 , 565 , 598 , 615 , 632 , 649 ,
1161 , 1167 , 1183 , 1223 , 1230 , 1232 , 1239		666 , 846 , 865 , 884 , 904 , 927 , 1005 ,
<code>\cs_if_free:nTF</code> 306 , 1012 , 1031		1024 , 1043 , 1062 , 1089 , 1143 , 1159 , 1181
<code>\cs_new_nopar:Nn</code> 177 , 182		H
<code>\cs_new_protected:Nn</code> 137 ,		hbox commands:
145 , 161 , 165 , 235 , 274 , 319 , 382 ,		<code>\hbox:n</code> 22 , 26 , 195
419 , 463 , 496 , 517 , 531 , 573 , 727 ,		<code>\hom</code> 343
746 , 765 , 775 , 785 , 790 , 913 , 1070 , 1165		I
<code>\cs_new_protected_nopar:Nn</code>		<code>\idmat</code> 21 , 70 , 1133
. 169 , 173 ,		<code>\inf</code> 344
187 , 191 , 243 , 250 , 256 , 373 , 471 ,		<code>\injlim</code> 345
476 , 481 , 489 , 504 , 508 , 512 , 684 ,		int commands:
688 , 692 , 702 , 713 , 718 , 798 , 809 , 815		<code>\int_compare:nTF</code> 732 , 740 , 751 , 761
<code>\csc</code> 337		<code>\int_step_function:nN</code> 769 , 779
<code>\curl</code> 12 , 14 , 69 , 70 , 588		<code>\int_step_inline:nn</code> 730 , 749 , 801
<code>\CurrentOption</code> 78		<code>\c_one_int</code> 803
<code>\cvector</code> 16 , 17 , 836		K
D		<code>\ker</code> 346
dalembert-symb (option) 6 , 91		keys commands:
<code>\DeclareCurrentRelease</code> 4		<code>\keys_define:nn</code> 22 , 91 , 126 , 127
<code>\DeclareDelimitedOperator</code> 8 , 313		<code>\keys_set:nn</code> 139 , 523 ,
<code>\DeclareMathOperator</code> 24		595 , 612 , 629 , 646 , 663 , 843 , 862 ,
<code>\DeclarePairedDelimiter</code>		881 , 901 , 921 , 924 , 1002 , 1021 , 1040 ,
. 24 , 230 , 231 , 232 , 233 , 234 , 1225 , 1234		1059 , 1079 , 1086 , 1140 , 1156 , 1174 , 1178
<code>\deg</code> 338		L
delimiter (option) 6 , 127		laplacian (option) 6 , 91
<code>\Delta</code> 427		<code>\laplacian</code> 14 , 69 , 70 , 588
delta-laplace (option) 6 , 91		laplacian-symb (option) 6 , 91
<code>\det</code> 339		<code>\lbrace</code> 232
<code>\diag</code> 31		
<code>\diagmat</code> 18 , 19 , 995		
<code>\dim</code> 340		

<code>\lbrack</code>	231	<code>\moremath_diagonal_smallmatrix:nn</code>	...	25, 26, 57, 765, 785, 794, 796, 819, 1042, 1114, 1116, 1118, 1120, 1122
<code>\lg</code>	347	<code>\moremath_divergence_operator:n</code>	...	24, 481, 481, 614, 673
<code>\lim</code>	348	<code>\moremath_gradient_operator:n</code>	...	24, 46–48, 471, 471, 484, 492, 597, 672
<code>\liminf</code>	349	<code>\moremath_id_matrix:n</code>	...	26, 66, 70, 809, 809, 821, 821, 1142, 1188, 1189, 1190, 1191, 1192
<code>\limsup</code>	350	<code>\moremath_id_smallmatrix:n</code>	...	26, 66, 70, 809, 815, 821, 822, 1158, 1193, 1195, 1197, 1199, 1201
<code>\ln</code>	351	<code>\moremath_laplace_operator:n</code>	...	24, 471, 476, 648, 675
<code>\log</code>	352	<code>\moremath_new_delim_op_command:NNN</code>	...	24, 37, 274, 274, 312, 316, 321, 322, 323, 324, 325
<code>\lparen</code>	230	<code>\moremath_row_smallvector:nn</code>	...	25, 713, 718, 724, 903, 976, 978, 980, 982, 984
<code>\lVert</code>	234, 1234	<code>\moremath_row_vector:nn</code>	...	25, 53, 692, 702, 712, 720, 864, 948, 950, 952, 954, 956
<code>\lvert</code>	233, 1225	<code>\moremath_setup:n</code>	...	23, 137, 137, 143
		<code>\moremath_vcenter:n</code>	...	26, 34, 43, 187, 187, 222, 377
		moremath internal commands:		
		<code>_moremath_constr_antidiagmat_row:n</code>	...	54, 727, 746, 780
		<code>_moremath_constr_diagmat_row:n</code>	...	54, 727, 727, 770
		<code>_moremath_dalembert_operator_get:</code>	...	70, 463, 463, 498
		<code>\l_moremath_dalembert_symb_tl</code>	...	70, 81, 121, 467
		<code>_moremath_generate_one_filled_clist:Nn</code>	...	56, 798, 798, 807, 807, 812, 818
		<code>\l_moremath_grad_op_tl</code>	...	29, 81, 106, 384, 416
		<code>_moremath_gradient_operator_get:</code>	...	382, 382, 473
		<code>\l_moremath_id_entries_clist</code>	...	57, 808, 811, 812, 813, 817, 818, 819
		<code>_moremath_in_Vert:w</code>	...	230, 234, 325, 579
		<code>_moremath_in_vert:w</code>	...	230, 233, 324, 578
		<code>_moremath_inbrace:w</code>	...	230, 232, 323, 577
		<code>_moremath_inbrack:w</code>	...	230, 231, 322, 576
M				
<code>\mathpalette</code>	34, 167			
mathtools commands:				
<code>\DeclarePairedDelimiter</code>	...			8, 22, 23, 36, 37, 47, 48
<code>\max</code>	353			
<code>\min</code>	354			
mode commands:				
<code>\mode_if_math:TF</code>	220, 223			
moremath commands:				
<code>\moremath_antidiagonal_matrix:nn</code>	...			25, 765, 775, 792, 794, 795, 1023, 1104, 1106, 1108, 1110, 1112
<code>\moremath_antidiagonal_smallmatrix:nn</code>	...			25, 765, 790, 794, 797, 1061, 1124, 1126, 1128, 1130, 1132
<code>\moremath_column_smallvector:nn</code>	...			25, 713, 713, 723, 883, 959, 962, 965, 968, 971
<code>\moremath_column_vector:nn</code>	...			25, 53, 692, 692, 711, 715, 845, 938, 940, 942, 944, 946
<code>\moremath_curl_operator:n</code>	...			24, 481, 489, 631, 674
<code>\moremath_dalembert_operator:n</code>	...			24, 70, 496, 496, 665, 676
<code>\moremath_delim_nabla_op_autoscale:NNnn</code>	24, 47, 504, 508, 555			
<code>\moremath_delim_nabla_op_manuscale:NNnnn</code>	...			24, 512, 512, 516, 516, 562
<code>\moremath_delim_nabla_op_noscale:NNnn</code>	24, 47, 504, 504, 560			
<code>\moremath_delim_op_autoscale:NNnnn</code>	23, 24, 36, 243, 250, 287, 300			
<code>\moremath_delim_op_manuscale:NNnnn</code>	23, 36, 256, 256, 262, 302			
<code>\moremath_delim_op_noscale:NNnnn</code>	23, 24, 36, 243, 243, 290			
<code>\moremath_diagonal_matrix:nn</code>	...			25, 26, 57, 765, 765, 787, 794, 794, 813, 1004, 1098, 1099, 1100, 1101, 1102

`_moremath_inparent:w`
 [230](#), [230](#), [321](#), [575](#)
`_moremath_laplace_operator_-`
 get: [419](#), [419](#), [478](#)
`\l_moremath_laplacian_arrow_-`
 bool [29](#), [81](#), [115](#), [429](#)
`\l_moremath_laplacian_delta_-`
 bool [29](#), [81](#), [112](#), [425](#), [452](#)
`\l_moremath_laplacian_symb_tl` ..
 [29](#), [81](#), [109](#), [435](#), [444](#), [447](#)
`\l_moremath_laplacian_tl`
 [29](#), [81](#), [119](#), [421](#), [460](#)
`_moremath_load_time_setup:` ...
 [145](#), [145](#), [149](#)
`\l_moremath_mat_diag_entries_-`
 seq [54](#), [55](#), [725](#),
 [730](#), [736](#), [749](#), [757](#), [767](#), [769](#), [777](#), [779](#)
`\l_moremath_mat_row_entries_seq`
 [54](#), [725](#),
 [729](#), [734](#), [739](#), [742](#), [748](#), [755](#), [760](#), [763](#)
`\l_moremath_mat_row_seq` [54](#)
`\l_moremath_matrix_align_tl` ...
 [31](#), [55](#), [129](#), [697](#), [707](#), [768](#), [778](#)
`\l_moremath_matrix_delim_tl` ...
 . [31](#), [129](#), [813](#), [819](#), [845](#), [864](#), [883](#),
 [903](#), [926](#), [1004](#), [1023](#), [1042](#), [1061](#), [1088](#)
`\l_moremath_matrix_fill_tl`
 [31](#), [55](#), [129](#), [739](#), [760](#)
`_moremath_matrix_star_begin:nn`
 . [33](#), [177](#), [177](#), [181](#), [697](#), [707](#), [768](#), [778](#)
`_moremath_matrix_star_end:n` ...
 . [33](#), [177](#), [182](#), [186](#), [699](#), [709](#), [771](#), [781](#)
`_moremath_maybe_vcenter:n`
 [373](#), [373](#), [392](#), [404](#), [432](#), [441](#)
`\l_moremath_nabla_arrow_bold_-`
 bool [81](#)
`\l_moremath_nabla_arrow_bool` ...
 [29](#), [43](#), [81](#), [98](#), [388](#)
`\l_moremath_nabla_bold_bool` ...
 [29](#), [44](#), [102](#), [394](#), [402](#), [434](#), [439](#)
`\l_moremath_nabla_tl`
 [29](#), [81](#), [94](#), [110](#), [398](#), [407](#), [410](#)
`_moremath_new_delim_nabla_doc_-`
 cmd:NNN [49](#),
 [531](#), [531](#), [572](#), [575](#), [576](#), [577](#), [578](#), [579](#)
`_moremath_new_delim_op_cmds:nN`
 [39](#), [319](#), [319](#), [329](#), [330](#),
 [331](#), [332](#), [333](#), [334](#), [335](#), [336](#), [337](#),
 [338](#), [339](#), [340](#), [341](#), [342](#), [343](#), [344](#),
 [345](#), [346](#), [347](#), [348](#), [349](#), [350](#), [351](#),
 [352](#), [353](#), [354](#), [355](#), [356](#), [357](#), [358](#),
 [359](#), [360](#), [361](#), [362](#), [363](#), [364](#), [365](#), [366](#)
`_moremath_new_id_matrix_doc_-`
 cmd:NNn
 [1165](#), [1165](#), [1188](#), [1189](#), [1190](#),
 [1191](#), [1192](#), [1193](#), [1195](#), [1197](#), [1199](#), [1201](#)
`_moremath_new_matrix_shorth_-`
 doc_cmd:NNn [63](#), [1070](#), [1070](#), [1098](#),
 [1099](#), [1100](#), [1101](#), [1102](#), [1103](#), [1105](#),
 [1107](#), [1109](#), [1111](#), [1113](#), [1115](#), [1117](#),
 [1119](#), [1121](#), [1123](#), [1125](#), [1127](#), [1129](#), [1131](#)
`_moremath_new_nabla_doc_-`
 cmds:nN
 [573](#), [573](#), [672](#), [673](#), [674](#), [675](#), [676](#)
`_moremath_new_vector_shorth_-`
 doc_cmd [59](#)
`_moremath_new_vector_shorth_-`
 doc_cmd:NNn [913](#),
 [913](#), [938](#), [940](#), [942](#), [944](#), [946](#), [948](#),
 [950](#), [952](#), [954](#), [956](#), [959](#), [962](#), [965](#),
 [968](#), [971](#), [976](#), [978](#), [980](#), [982](#), [984](#)
`_moremath_operator:Nnn`
 [36](#), [235](#), [235](#), [245](#), [252](#), [258](#)
`_moremath_parse_kv_args:nN` ...
 [517](#), [517](#), [544](#)
`\l_moremath_predef_abs_bool` ...
 [27](#), [17](#), [38](#), [1221](#)
`\l_moremath_predef_crvector_-`
 bool [27](#), [17](#), [46](#), [834](#), [987](#), [992](#)
`\l_moremath_predef_matrix_bool` .
 [27](#), [65](#), [17](#), [42](#), [993](#), [1203](#), [1208](#)
`\l_moremath_predef_operators_-`
 bool [27](#), [17](#), [34](#), [327](#)
`\l_moremath_predef_vector_op_-`
 bool .. [27](#), [17](#), [30](#), [155](#), [581](#), [677](#), [682](#)
`_moremath_seq_to_column_-`
 vector:N [684](#), [684](#), [698](#)
`_moremath_seq_to_row_vector:N` .
 [684](#), [688](#), [708](#)
`_moremath_tex_mathop:n` [33](#),
 [70](#), [173](#), [173](#), [386](#), [413](#), [423](#), [465](#), [468](#)
`_moremath_tex_mathpalette:Nn` ..
 [32](#), [71](#), [165](#), [165](#), [189](#)
`_moremath_tex_mathsurround:n` ..
 [33](#), [71](#), [169](#), [169](#), [198](#)
`_moremath_tex_vcenter:n`
 [32](#), [71](#), [161](#), [161](#), [193](#)
`_moremath_vcenter:Nn`
 [34](#), [189](#), [191](#), [191](#)
`\l_moremath_vcenter_bool`
 [29](#), [43](#), [90](#), [123](#), [375](#)
`\l_moremath_vector_entries_seq` .
 . [52](#), [683](#), [694](#), [695](#), [698](#), [704](#), [705](#), [708](#)
`\moremathsetup` [5](#), [6](#), [141](#)
msg commands:
 `\msg_error:nnn` [224](#)
 `\msg_info:nn` [25](#), [157](#)
 `\msg_info:nnnn` [368](#), [678](#), [988](#), [1204](#), [1241](#)

<code>\msg_line_context</code>	10, 15, 153, 207, 211, 266, 272, 529, 586, 826, 1212
<code>\msg_new:nnn</code>	7, 12, 150, 204, 263, 268, 525, 583
<code>\msg_new:nnnn</code>	209, 823, 1209
<code>\msg_redirect_name:nnn</code>	315, 317
<code>\msg_warning:nnn</code>	228, 298, 307, 568, 601, 618, 635, 652, 669, 850, 869, 888, 908, 931, 1008, 1027, 1046, 1065, 1092, 1146, 1162, 1184, 1228, 1237
<code>\msg_warning:nnnn</code>	552
N	
<code>nabla</code> (option)	5, 91
<code>\nabla</code>	95
<code>\NeedsTeXFormat</code>	3
<code>\NewDelimitedOperator</code>	5
<code>\NewDocumentCommand</code>	141, 218, 278, 313, 535, 590, 606, 623, 640, 657, 838, 857, 876, 896, 917, 997, 1016, 1035, 1054, 1074, 1135, 1151, 1169
<code>no-abs-shorthands</code> (option)	5, 22, 22
<code>no-crvector</code> (option)	5, 15, 22
<code>no-matrix</code> (option)	5, 17, 22
<code>no-operators</code> (option)	5, 6, 22
<code>no-vector</code> (option)	5, 9, 22
<code>\nolimits</code>	414, 451, 469
<code>nopredef</code> (option)	5, 22
<code>\norm</code>	22, 1223, 1243
O	
<code>\operatorname</code>	36
options:	
<code>align</code>	6, 127
<code>arrowlaplace</code>	6, 91
<code>arrownabla</code>	5, 91
<code>bm</code>	5, 22
<code>boldnabla</code>	5, 91
<code>dalembert-symb</code>	6, 91
<code>delimiter</code>	6, 127
<code>delta-laplace</code>	6, 91
<code>fill</code>	6, 127
<code>grad-op</code>	6, 91
<code>laplacian</code>	6, 91
<code>laplacian-symb</code>	6, 91
<code>nabla</code>	5, 91
<code>no-abs-shorthands</code>	5, 22, 22
<code>no-crvector</code>	5, 15, 22
<code>no-matrix</code>	5, 17, 22
<code>no-operators</code>	5, 6, 22
<code>no-vector</code>	5, 9, 22
<code>nopredef</code>	5, 22
<code>vcenter</code>	6, 9, 123
P	
<code>\pantidiagmat</code>	19, 1103
<code>\parccos</code>	7, 329
<code>\parcsin</code>	330
<code>\parctan</code>	331
<code>\Parg</code>	332
<code>\PassOptionsToPackage</code>	78
<code>\pcos</code>	333
<code>\pcosh</code>	334
<code>\pcot</code>	335
<code>\pcoth</code>	336
<code>\pcsc</code>	337
<code>\pcurl</code>	13, 674
<code>\pcvector</code>	16, 937
<code>\pdeg</code>	338
<code>\pdet</code>	339
<code>\pdiagmat</code>	19, 1098
<code>\pdim</code>	340
<code>\pdiv</code>	12, 673
<code>\pexp</code>	341
<code>\pgcd</code>	342
<code>\pgrad</code>	10, 672
<code>\phom</code>	343
<code>\pidmat</code>	21, 70, 1188
<code>\pinf</code>	344
<code>\pinjlim</code>	345
<code>\pker</code>	346
<code>\plaplacian</code>	14, 675
<code>\plg</code>	347
<code>\plim</code>	348
<code>\pliminf</code>	349
<code>\plimsup</code>	350
<code>\pln</code>	351
<code>\plog</code>	352
<code>\pmax</code>	353
<code>\pmin</code>	354
<code>\pPr</code>	355
<code>\pprojlim</code>	356
<code>\pquabla</code>	15, 676
<code>\Pr</code>	355
<code>\ProcessKeyOptions</code>	147
<code>\projlim</code>	356
prop commands:	
<code>\prop_pop:NnNTF</code>	521
<code>\prop_set_from_keyval:Nn</code>	519
<code>\prop_to_keyval:N</code>	523
<code>\l_tmpa_prop</code>	519, 521, 523
<code>\ProvidesExplPackage</code>	5
<code>\prvector</code>	16, 937
<code>\psec</code>	357
<code>\psin</code>	358
<code>\psinh</code>	359
<code>\psmallantidiagmat</code>	20, 1123
<code>\psmallcvector</code>	17, 957

<code>\psmalldiagmat</code>	20, 1113	<code>\mathop</code>	33, 44, 45
<code>\psmallidmat</code>	21, 70, 1193	<code>\mathopen</code>	46
<code>\psmallrvector</code>	17, 957	<code>\mathpalette</code>	22, 26, 32, 34
<code>\psup</code>	360	<code>\mathsurround</code>	33, 34
<code>\ptan</code>	361	<code>\par</code>	34
<code>\ptanh</code>	362	<code>\vcenter</code>	22, 26, 32, 34, 43
<code>\pvarinjlim</code>	363	<code>\Vert</code>	52
<code>\pvarliminf</code>	363	<code>\vert</code>	52
<code>\pvarlimsup</code>	363	tex commands:	
<code>\pvarprojlim</code>	363	<code>\tex_mathop:D</code>	175
Q		<code>\tex_mathsurround:D</code>	171
<code>\quabla</code>	15, 51, 70, 655	<code>\tex_vcenter:D</code>	32, 163
R		<code>\times</code>	494
<code>\rbrace</code>	232	tl commands:	
<code>\rbrack</code>	231	<code>\tl_if_empty:NTF</code>	384, 421, 550, 558
<code>\RequirePackage</code>	26, 158, 160	<code>\tl_if_empty:nTF</code>	239, 241, 474, 479, 499, 1077, 1083, 1172
<code>\rparen</code>	230	<code>\tl_if_novalue:nTF</code>	281, 542, 593, 610, 627, 644, 661, 841, 860, 879, 899, 922, 1000, 1019, 1038, 1057, 1084, 1138, 1154, 1176
<code>\rvector</code>	16, 17, 836	<code>\tl_new:N</code>	81, 84, 85, 88, 89
<code>\rVert</code>	234, 1234	<code>\l_tmpa_tl</code>	544, 550, 553, 558, 562
<code>\rvert</code>	233, 1225	token commands:	
S		<code>\c_math_subscript_token</code>	241, 474, 479, 501
<code>\sec</code>	357	<code>\c_math_superscript_token</code>	454
seq commands:		V	
<code>\seq_clear:N</code>	694, 704, 729, 748, 800	<code>\Vantidiagmat</code>	19, 1103
<code>\seq_clear_new:N</code>	725, 726	<code>\vantidiagmat</code>	19, 1103
<code>\seq_count:N</code>	730, 749, 769, 779	<code>\Varccos</code>	7, 329
<code>\seq_item:Nn</code>	736, 757	<code>\varccos</code>	7, 329
<code>\seq_new:N</code>	683	<code>\Varcsin</code>	330
<code>\seq_put_left:Nn</code>	755, 760	<code>\varcsin</code>	330
<code>\seq_put_right:Nn</code>	734, 739, 803	<code>\Varctan</code>	331
<code>\seq_set_from_clist:Nn</code>	695, 705, 767, 777	<code>\varctan</code>	331
<code>\seq_use:Nn</code>	686, 690, 742, 763	<code>\Varg</code>	332
<code>\l_tmpa_seq</code>	800, 803, 805	<code>\varg</code>	332
<code>\shortcvector</code>	60	<code>\varinjlim</code>	363
<code>\shortrvector</code>	60	<code>\varliminf</code>	365
<code>\sin</code>	358	<code>\varlimsup</code>	366
<code>\sinh</code>	359	<code>\varprojlim</code>	364
<code>\smallantidiagmat</code>	19, 995	<code>vcenter (option)</code>	6, 9, 123
<code>\smallcvector</code>	17, 836	<code>\VCenterMath</code>	22, 23, 216
<code>\smalldiag</code>	31	<code>\Vcos</code>	333
<code>\smalldiagmat</code>	19, 995	<code>\vcos</code>	333
<code>\smallidmat</code>	21, 70, 1133	<code>\Vcosh</code>	334
<code>\smallrvector</code>	17, 836	<code>\vcosh</code>	334
<code>\square</code>	122	<code>\Vcot</code>	335
<code>\sup</code>	360	<code>\vcot</code>	335
T		<code>\Vcoth</code>	336
<code>\tan</code>	361	<code>\vcoth</code>	336
<code>\tanh</code>	362	<code>\Vcsc</code>	337
TeX and L ^A T _E X 2 _ε commands:			
<code>\cdot</code>	46		

<code>\vcsc</code>	337	<code>\Vlog</code>	352
<code>\Vcurl</code>	13, 674	<code>\vlog</code>	352
<code>\vcurl</code>	13, 674	<code>\Vmax</code>	353
<code>\Vcvector</code>	16, 937	<code>\vmax</code>	353
<code>\vcvector</code>	16, 937	<code>\Vmin</code>	354
<code>\Vdeg</code>	338	<code>\vmin</code>	354
<code>\vdeg</code>	338	<code>\VPr</code>	355
<code>\Vdet</code>	339	<code>\vPr</code>	355
<code>\vdet</code>	339	<code>\Vprojlim</code>	356
<code>\Vdiagmat</code>	19, 1098	<code>\vprojlim</code>	356
<code>\vdiagmat</code>	19, 1098	<code>\Vquabla</code>	15, 676
<code>\Vdim</code>	340	<code>\vquabla</code>	15, 676
<code>\vdim</code>	340	<code>\Vrvector</code>	16, 937
<code>\Vdiv</code>	12, 673	<code>\vrvector</code>	16, 937
<code>\vdiv</code>	12, 673	<code>\Vsec</code>	357
<code>\vec</code>	390, 431	<code>\vsec</code>	357
<code>\Vexp</code>	341	<code>\Vsin</code>	358
<code>\vexp</code>	341	<code>\vsin</code>	358
<code>\Vgcd</code>	342	<code>\Vsinh</code>	359
<code>\vgcd</code>	342	<code>\vsinh</code>	359
<code>\Vgrad</code>	10, 672	<code>\Vsmallantidiagmat</code>	20, 1123
<code>\vgrad</code>	10, 672	<code>\vsmallantidiagmat</code>	20, 1123
<code>\Vhom</code>	343	<code>\Vsmallcvector</code>	17, 957
<code>\vhom</code>	343	<code>\vsmallcvector</code>	17, 957
<code>\Vidmat</code>	21, 70, 1188	<code>\Vsmallldiagmat</code>	20, 1113
<code>\vidmat</code>	21, 70, 1188	<code>\vsmallldiagmat</code>	20, 1113
<code>\Vinf</code>	344	<code>\Vsmallldmat</code>	21, 70, 1193
<code>\vinf</code>	344	<code>\vsmallldmat</code>	21, 70, 1193
<code>\Vinjlim</code>	345	<code>\Vsmallrvector</code>	17, 957
<code>\vinjlim</code>	345	<code>\vsmallrvector</code>	17, 957
<code>\Vker</code>	346	<code>\Vsup</code>	360
<code>\vker</code>	346	<code>\vsup</code>	360
<code>\Vlaplacian</code>	14, 675	<code>\Vtan</code>	361
<code>\vlaplacian</code>	14, 675	<code>\vtan</code>	361
<code>\Vlg</code>	347	<code>\Vtanh</code>	362
<code>\vlg</code>	347	<code>\vtanh</code>	362
<code>\Vlim</code>	348	<code>\Vvarinjlim</code>	363
<code>\vlim</code>	348	<code>\vvarinjlim</code>	363
<code>\Vliminf</code>	349	<code>\Vvarliminf</code>	363
<code>\vliminf</code>	349	<code>\vvarliminf</code>	363
<code>\Vlimsup</code>	350	<code>\Vvarlimsup</code>	363
<code>\vlimsup</code>	350	<code>\vvarlimsup</code>	363
<code>\Vln</code>	351	<code>\Vvarprojlim</code>	363
<code>\vln</code>	351	<code>\vvarprojlim</code>	363