Fun Template 2

Senan Sekhon

February 25, 2023

Contents

Conventions

$\mathbb{F}$ denotes either $\mathbb{R}$ or $\mathbb{C}$.
$\mathbb{N}$ denotes the set $\{1, 2, 3, \ldots\}$ of natural numbers (excluding 0).
Inner products are taken to be linear in the first argument and conjugate linear in the second.
The Einstein summation convention is used for tensors unless otherwise specified.

1 Main structure

We start by defining several theorem-type environments (e.g. definition, proposition, theorem, lemma, corollary) using the `\newtcbtheorem` macro from the `tcolorbox` package. In a real mathematics document, these will all have the same format (perhaps differing only in color). The purpose of having different formats here is to help you choose your favorite one (or adapt one or more of them to your liking) and apply it to all of the theorem-type environments.

For best practices and other general advice on writing in LaTeX, see [1, 2].

1.1 Theorem environments

Definition 1.1

A function $f : I \to \mathbb{R}$ is **continuous** at $a \in I$ if $\lim_{x\to a} f(x) = f(a)$.

Definition 1.2. The Derivative

The **derivative** of a function $f : I \to \mathbb{R}$ at $a \in I$ is given by:

$$f'(a) = \lim_{x\to a} \frac{f(x) - f(a)}{x - a}$$

Note that Definition 1.2 has a name (The Derivative), while Definition 1.1 does not.

You know those awesome commutative diagrams?

$$\begin{tikzcd} A \arrow[r, "p"] \arrow[d, "q"'] & B \arrow[d, "r"] \\ C \arrow[r, "s"'] & D \end{tikzcd}$$

The derivative has *nothing* to do with them!

Proposition 1.3. Differentiable ⇒ continuous

If f is differentiable at a, then f is continuous at a.

Proof. Exercise (but only because this is a template). 😎

The converse of Proposition 1.3 is not true in general.

Examples.

1. $f(x) = |x|$ (not differentiable at 0)
2. $f(x) = \begin{cases} \sin(x) & x \geq 0 \\ 0 & x < 0 \end{cases}$ (not differentiable at 0)

Theorem 1.4. Two statements

The following statements are true:

1. First statement
2. Second statement

Proof.

1. Proof of (1).
2. Proof of (2). Note that the QED symbol is on this line, thanks to `\qedhere`. 😎

Corollary 1.5 (Another statement)**.** Third statement.

Proof. This follows by combining (1) and (2) of Theorem 1.4. ← Try clicking on these 😎

Remark. This corollary is also obvious.

Corollary 1.6. Unlike Corollary 1.5, this corollary has no name.

Lemma 1.7 $(a + b)^2 = a^2 + 2ab + b^2$

Proof. Expand the left side. Note that there is no QED symbol in this proof, thanks to `\let\qed\relax`.

Remarks.

1. It's also kind of obvious.
2. No extra points for guessing what $(a-b)^2$ is.

Example. $(2+4)^2 = 2^2 + 2 \cdot 2 \cdot 4 + 4^2 = 4 + 16 + 16 = 36$

Proposition 1.8

$$(\cos(x) + i\sin(x))^2 = \cos(x)^2 - \sin(x)^2 + 2i\sin(x)\cos(x) \tag{1}$$

Proof. Exercise (use Lemma 1.7). 😎

Note that referencing proposition 1.8 is not the same as referencing (1).

Lemma 1.9. Euler's identity

$$e^{ix} = \cos(x) + i\sin(x)$$

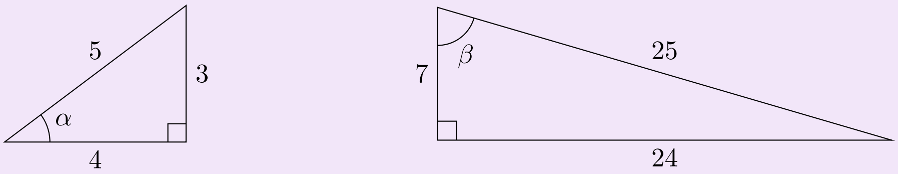
Theorem 1.10

$$\cos(2x) = \cos(x)^2 - \sin(x)^2 \qquad \sin(2x) = \sin(x)\cos(x) \tag{2}$$

Proof. By Lemma 1.9, the left side of (1) is simply $\left(e^{ix}\right)^2 = e^{2ix}$. Now take the real and imaginary parts of the right side. 😎

This proof assumes x and y are real numbers. What would you do if they were complex numbers?

Corollary 1.11. Suppose α and β are the angles shown below:



Then $\beta = 2\alpha$.

Proof. Exercise (use Theorem 1.10). 😎

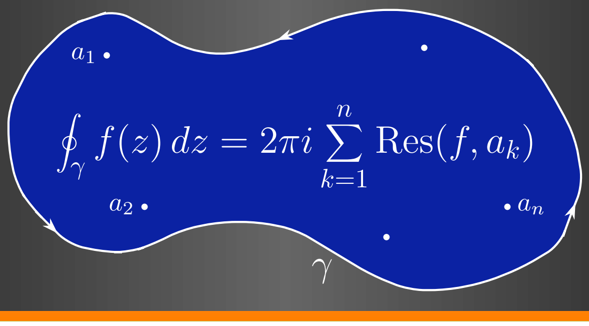
You can generate other Pythagorean triples this way[1].

1.2 Other options

There's no need to stick to the $\boxed{\textit{status quo}}$ with LaTeX formatting. You can go ⚡**WILD**⚡ !

Theorem 1.12. Residue Theorem

Suppose Ω is a simply connected open subset of $\mathbb{C}$, γ is a positively oriented simple closed curve in Ω and $f : \Omega \to \mathbb{C}$ is holomorphic on Ω except at the points $a_1, a_2, \ldots, a_n$ inside γ. Then:



$$\oint_\gamma f(z)\,dz = 2\pi i \sum_{k=1}^{n} \operatorname{Res}(f, a_k)$$

ENGINEERS LOVE THIS!

A lot of nice integrals can be computed using the $\underbrace{\text{residue theorem}}_{\text{Click on this link}}$, see [3, §5.2].

[1] In fact, you can generate *all* Pythagorean triples this way. Can you figure out how?

2 Macros

2.1 The talign environment

The talign and talign* environments work like the align and align* environments, except they render equations in inline size. They have the same numbering and labeling options as align and align*.

```
\begin{align}
    \sum_{n=1}^2 \frac{1}{3n}=\int_0^1 x\,dx
\end{align}
```

$$\sum_{n=1}^{2} \frac{1}{3n} = \int_0^1 x\,dx \tag{3}$$

```
\begin{align*}
    \sum_{n=1}^2 \frac{1}{3n}=\int_0^1 x\,dx
\end{align*}
```

$$\sum_{n=1}^{2} \frac{1}{3n} = \int_0^1 x\,dx$$

```
\begin{talign}
    \sum_{n=1}^2 \frac{1}{3n}=\int_0^1 x\,dx
\end{talign}
```

$$\textstyle\sum_{n=1}^{2} \frac{1}{3n} = \int_0^1 x\,dx \tag{4}$$

```
\begin{talign*}
    \sum_{n=1}^2 \frac{1}{3n}=\int_0^1 x\,dx
\end{talign*}
```

$$\textstyle\sum_{n=1}^{2} \frac{1}{3n} = \int_0^1 x\,dx$$

2.2 Vector and Matrix macros

The \cvc, \rvc and \mat commands can be used to typeset column vectors, row vectors and $m \times n$ matrices efficiently. The vector/matrix delimiters can also be changed, using the optional argument p, b, B, v or V (in accordance with pmatrix, bmatrix, Bmatrix, vmatrix or Vmatrix respectively from amsmath). The default is parentheses ( ). Of course, these commands can only be used in math mode.

```
$$\cvc{2,3,1}$$
```

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

```
$$\cvc[]{2,3,1}$$
```

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

```
$$\rvc{5,7,6}$$
```

$$\begin{pmatrix} 5 & 7 & 6 \end{pmatrix}$$

```
$$\rvc[p]{5,7,6}$$
```

$$\begin{pmatrix} 5 & 7 & 6 \end{pmatrix}$$

```
$$\rvc{8,,4}$$
```

$$\begin{pmatrix} 8 & & 4 \end{pmatrix}$$

```
$$\rvc[b]{8,,4}$$
```

$$\begin{bmatrix} 8 & & 4 \end{bmatrix}$$

```
$$\rvc{a,b_1,c^2,\&}$$
```

$$\begin{pmatrix} a & b_1 & c^2 & \& \end{pmatrix}$$

```
$$\rvc[B]{a,b_1,c^2,\&}$$
```

$$\begin{Bmatrix} a & b_1 & c^2 & \& \end{Bmatrix}$$

```
$$\mat{4,8}{0,9}{6,3}$$
```

$$\begin{pmatrix} 4 & 8 \\ 0 & 9 \\ 6 & 3 \end{pmatrix}$$

```
$$\mat[v]{4,8}{0,9}{6,3}$$
```

$$\begin{vmatrix} 4 & 8 \\ 0 & 9 \\ 6 & 3 \end{vmatrix}$$

```
$$\mat{1,7}{8,9,2,5}$$
```

$$\begin{pmatrix} 1 & 7 & & \\ 8 & 9 & 2 & 5 \end{pmatrix}$$

```
$$\mat[V]{1,7}{8,9,2,5}$$
```

$$\begin{Vmatrix} 1 & 7 & & \\ 8 & 9 & 2 & 5 \end{Vmatrix}$$

From now on, every subsection will start on a new page (see the code for how to do this).

2.3 Box macros

Here we provide three customizable `tcolorbox` environments that are perhaps less suited to theorem-type environments (but of course, they can still be used for these) and more suited to “lower-level” environments, e.g. discussions or examples one would like to explain in more detail.

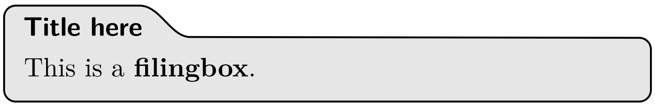
The `filingbox` environment

The `filingbox` environment is a `tcolorbox` environment that creates boxes in the shape of filing dividers. It has two optional arguments (the first delimited by parentheses, the second by brackets) and one mandatory argument.

- The first optional argument specifies the position of the tab. It can be `ul` (up left), `uc` (up center), `ur` (up right), `dl` (down left), `dc` (down center) and `dr` (down right). The default is `ul`.
- The second optional argument specifies the color of the box. The background of the box is automatically set to this color with 10% opacity. The default is black.
- The mandatory argument specifies the title.

Additional `tcolorbox` options can be specified after the title.

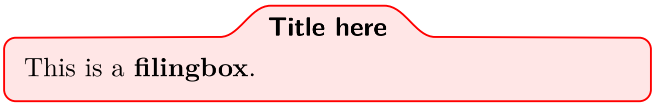
```
\begin{filingbox}{Title here}
This is a \textbf{filingbox}.
\end{filingbox}
```



Title here

This is a **filingbox**.

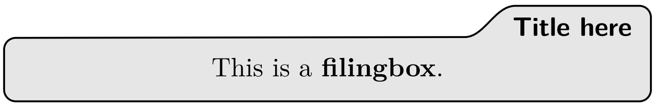
```
\begin{filingbox}(uc)[red]{Title here}
This is a \textbf{filingbox}.
\end{filingbox}
```



Title here

This is a **filingbox**.

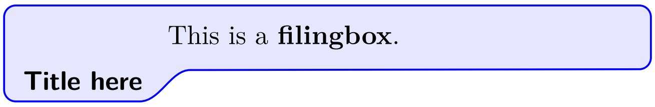
```
\begin{filingbox}(ur){Title here}[halign=center]
This is a \textbf{filingbox}.
\end{filingbox}
```



Title here

This is a **filingbox**.

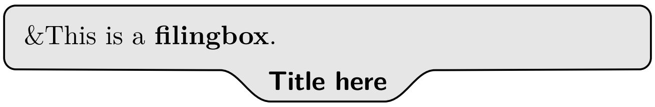
```
\begin{filingbox}(dl)[blue]{Title here}[left=20mm]
This is a \textbf{filingbox}.
\end{filingbox}
```



This is a **filingbox**.

Title here

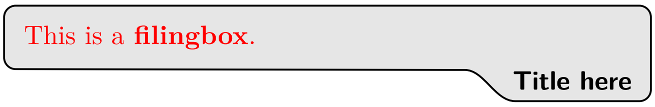
```
\begin{filingbox}(dc){Title here}[before upper=\&]
This is a \textbf{filingbox}.
\end{filingbox}
```



&This is a **filingbox**.

Title here

```
\begin{filingbox}(dr){Title here}[colupper=red]
This is a \textbf{filingbox}.
\end{filingbox}
```



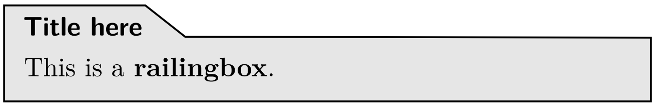
This is a **filingbox**.

Title here

The `railingbox` environment

The `railingbox` environment is a `tcolorbox` environment that creates boxes in the shape of railing bars. It is similar to the `filingbox` environment, except that it produces a box with sharp corners. It has the same argument structure as `filingbox`.

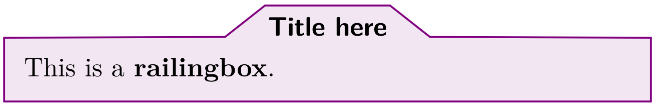
```
\begin{railingbox}{Title here}
This is a \textbf{railingbox}.
\end{railingbox}
```



Title here

This is a **railingbox**.

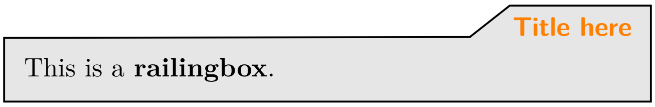
```
\begin{railingbox}(uc)[violet]{Title here}
This is a \textbf{railingbox}.
\end{railingbox}
```



Title here

This is a **railingbox**.

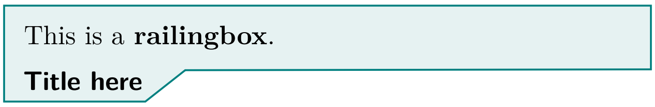
```
\begin{railingbox}(ur){Title here}[coltitle=orange]
This is a \textbf{railingbox}.
\end{railingbox}
```



Title here

This is a **railingbox**.

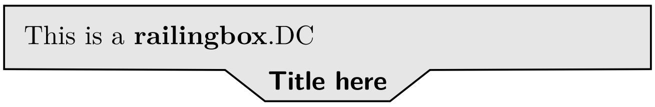
```
\begin{railingbox}(dl)[teal]{Title here}
This is a \textbf{railingbox}.
\end{railingbox}
```



This is a **railingbox**.

Title here

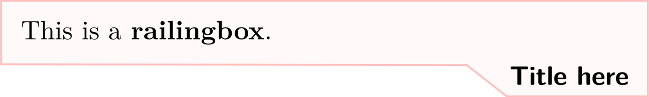
```
\begin{railingbox}(dc){Title here}[after upper=DC]
This is a \textbf{railingbox}.
\end{railingbox}
```



This is a **railingbox**.DC

Title here

```
\begin{railingbox}(dr)[pink]{Title here}
This is a \textbf{railingbox}.
\end{railingbox}
```



This is a **railingbox**.

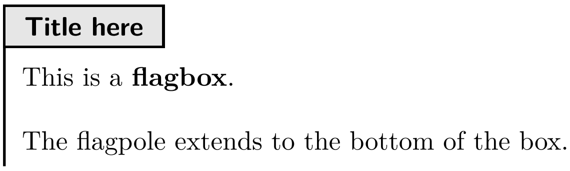
Title here

The `flagbox` environment

The `flagbox` environment is a `tcolorbox` environment that creates boxes in the shape of flags. It has the same argument structure as `filingbox`, except that the position of the tab can only be `ul`, `ur`, `dl` or `dr` (the default is still `ul`).

```
\begin{flagbox}{Title here}
This is a \textbf{flagbox}.\\

The flagpole extends to the bottom of the box.
\end{flagbox}
```



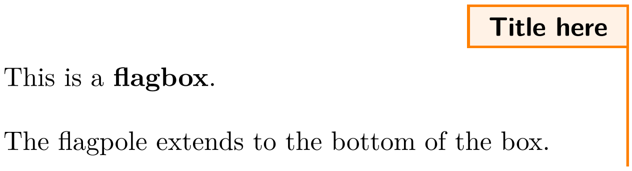
Title here

This is a **flagbox**.

The flagpole extends to the bottom of the box.

```
\begin{flagbox}(ur)[orange]{Title here}
This is a \textbf{flagbox}.\\

The flagpole extends to the bottom of the box.
\end{flagbox}
```



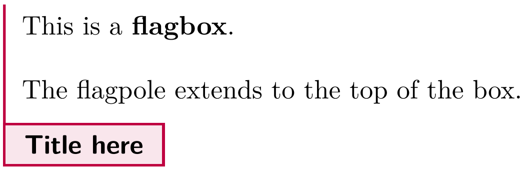
Title here

This is a **flagbox**.

The flagpole extends to the bottom of the box.

```
\begin{flagbox}(dl)[purple]{Title here}
This is a \textbf{flagbox}.\\

The flagpole extends to the top of the box.
\end{flagbox}
```



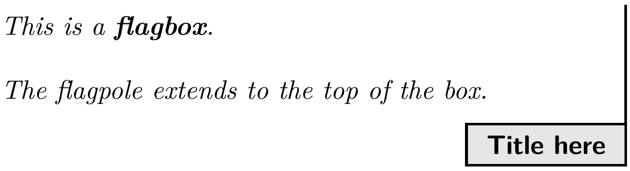
This is a **flagbox**.

The flagpole extends to the top of the box.

Title here

```
\begin{flagbox}(dr){Title here}[fontupper=\itshape]
This is a \textbf{flagbox}.\\

The flagpole extends to the top of the box.
\end{flagbox}
```



*This is a **flagbox**.*

The flagpole extends to the top of the box.

Title here

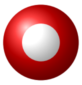
2.4 Ball macros

The command `\solidball` produces a customizable solid pool ball. It has four arguments, two optional and two mandatory.

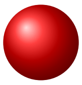
- The first (optional) argument specifies the text in the center of the ball. The default is blank (no text). The text format can also be manually specified, e.g. with `\color{white} abc`.
- The second (mandatory) argument specifies the color of the ball.
- The third (optional) argument specifies the color of the "background" of the ball, i.e. the area behind the text (though it will be shown even if there is no text). The default is `white`.
- The fourth (mandatory) argument specifies the size of the ball. If this is `1`, the diameter of the ball will be 1 cm. This argument can be specified as an integer (e.g. `2`), a floating-point number (e.g. `0.75`) or a mathematical expression (e.g. `2*3/8`).

There is also a starred version `\solidball*`, where the background is not drawn. In this version, the third argument is redundant.

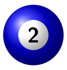
```
\solidball{red}{1}
```



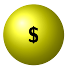
```
\solidball*{red}{1}
```



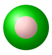
```
\solidball[2]{blue}{0.8}
```



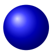
```
\solidball*[\$]{yellow}{4/5}
```



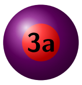
```
\solidball{green}[pink]{0.6}
```



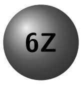
```
\solidball*{blue}[green]{(9-3)/10}
```



```
\solidball[3a]{violet}[red]{1}
```

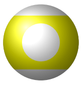


```
\solidball*[6Z]{gray}[blabla]{1}
```

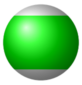


The command `\stripedball` produces a customizable striped pool ball. It has the same argument structure as `\solidball`, as well as a starred version that omits the background (but the stripe is still drawn, so the third argument is no longer redundant).

```
\stripedball{yellow}{1}
```



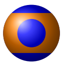
```
\stripedball*{green}{1}
```



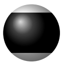
```
\stripedball[2]{violet}{0.5}
```

```
\stripedball*[V]{white}[brown]{1/2}
```

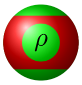
```
\stripedball{orange}[blue]{.75}
```



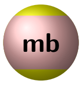
```
\stripedball*{black}{sqrt(9/16)}
```



```
\stripedball[$\rho$]{red}[green]{1}
```

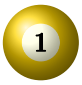


```
\stripedball*[mb]{pink}[yellow]{1}
```

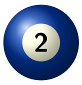


The command `\poolball` is based on `\solidball` and `\stripedball` and produces the standard (Aramith Tournament) pool balls. It has one optional argument and two mandatory arguments.

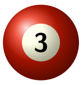
- The optional argument specifies the text in the center of the ball. If this is not specified, the value of the first mandatory argument will be used.
- The first mandatory argument specifies the number on the ball (the color and whether it is solid or striped are automatically determined from this). If this is `0` (or any multiple of 16), it will produce the cue ball.
- The second mandatory argument specifies the size of the ball (identically to `\solidball` and `\stripedball`).



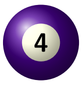
`\poolball{17}{1}`



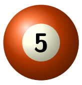
`\poolball{2}{1}`



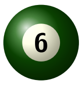
`\poolball{3}{1}`



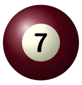
`\poolball{4}{1}`



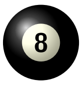
`\poolball{5}{1}`



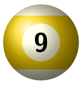
`\poolball{6}{1}`



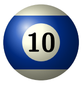
`\poolball{7}{1}`



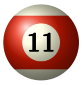
`\poolball{8}{1}`



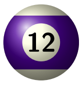
`\poolball{9}{1}`



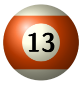
`\poolball{10}{1}`



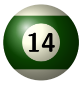
`\poolball{11}{1}`



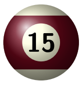
`\poolball{12}{1}`



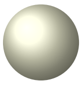
`\poolball{13}{1}`



`\poolball{14}{1}`



`\poolball{15}{1}`



`\poolball{0}{1}`

The first argument is only defined modulo 16, so `\poolball{19}{1}` and `\poolball{-13}{1}` produce the same result as `\poolball{3}{1}`.

2.5 Smart Operator Macros

The `\Sum` command is a “smart” version of the `\sum` command (provided by LaTeX) that incorporates styles and upper and lower limits into its arguments. It is based on the options provided by `amsmath`.

The `\Sum` command takes one optional argument and one set of mandatory arguments, separated by commas.

- If no mandatory argument is present, the sum is over n and runs from $n = 1$ to ∞.
- If one mandatory argument #1 is present, the sum is over n and runs from $n =$ #1 to ∞.
- If two mandatory arguments #1, #2 are present, the sum is over n and runs from $n =$ #1 to #2.
- If three mandatory arguments #1, #2, #3 are present, the sum is over #1 and runs from #1 = #2 to #3.
- If four mandatory arguments #1, #2, #3, #4 are present, the sum is over #1 *and* #2 and runs from #1, #2 = #3 to #4.
- If five mandatory arguments #1, #2, #3, #4, #5 are present, the sum is over #1, #2 *and* #3 runs from #1, #2, #3 = #4 to #5.

⚠ The mandatory argument cannot be omitted! Simply writing `\Sum` will lead to errors.

Inline mode

`$\Sum{}$`	$\sum_{n=1}^{\infty}$	`$\Sum{4}$`	$\sum_{n=4}^{\infty}$
`$\Sum{3,8}$`	$\sum_{n=3}^{8}$	`$\Sum{m,6,25}$`	$\sum_{m=6}^{25}$
`$\Sum{i,j,\Omega,\tau}$`	$\sum_{i,j=\Omega}^{\tau}$	`$\Sum{\&,\$,\{,?,+}$`	$\sum_{\&,\$,\{=?}^{+}$

Display mode

`$$\Sum{}$$`	$\displaystyle\sum_{n=1}^{\infty}$	`$$\Sum{4}$$`	$\displaystyle\sum_{n=4}^{\infty}$
`$$\Sum{3,8}$$`	$\displaystyle\sum_{n=3}^{8}$	`$$\Sum{m,6,25}$$`	$\displaystyle\sum_{m=6}^{25}$
`$$\Sum{i,j,\Omega,\tau}$$`	$\displaystyle\sum_{i,j=\Omega}^{\tau}$	`$$\Sum{\&,\$,\{,?,+}$$`	$\displaystyle\sum_{\&,\$,\{=?}^{+}$

The optional argument of the `\Sum` command specifies the style of the sum. The standard `\sum` command can be customized in the following ways:

- By adding `\limits` (to show the limits above and below the sum) or `\nolimits` (to show the limits beside the sum). These are added *after* the command `\sum`.
- It can also be customized by adding `\textstyle` (to show the sum in inline style) or `\displaystyle` (to show the sum in display style). These are added *before* the command `\sum`.

In the smart `\Sum` command, these options are incorporated into the optional argument.

There are eight options for the optional argument of `\Sum`:

`l`	Limits above and below	`il`	Inline style with limits above and below
`n`	Limits beside	`in`	Inline style with limits beside
`i`	Inline style	`dl`	Display style with limits above and below
`d`	Display style	`dn`	Display style with limits beside

The optional argument overrides any pre-defined options, i.e. `\textstyle\Sum[d]{8}` is equivalent to `\Sum[d]{8}`.

If the optional argument is empty, it will be treated as absent, i.e. `\Sum[]{8}` is equivalent to `\Sum{8}`.

`$$\Sum[l]{}$$`	$\displaystyle\sum\limits_{n=1}^{\infty}$	`$$\Sum[il]{}$$`	$\textstyle\sum\limits_{n=1}^{\infty}$
`$$\Sum[n]{}$$`	$\displaystyle\sum\nolimits_{n=1}^{\infty}$	`$$\Sum[in]{}$$`	$\textstyle\sum\nolimits_{n=1}^{\infty}$
`$$\Sum[i]{}$$`	$\textstyle\sum\nolimits_{n=1}^{\infty}$	`$$\Sum[dl]{}$$`	$\displaystyle\sum\limits_{n=1}^{\infty}$
`$$\Sum[d]{}$$`	$\displaystyle\sum\limits_{n=1}^{\infty}$	`$$\Sum[dn]{}$$`	$\displaystyle\sum\nolimits_{n=1}^{\infty}$

All of the standard large operators have a “smart” version, as shown below:

Standard operator	`\sum`	`\prod`	`\coprod`	`\bigcap`	`\bigcup`	`\bigsqcup`
Smart operator	`\Sum`	`\Prod`	`\Coprod`	`\Capp`	`\Cupp`	`\Kupp`
Inline style	$\sum_{n=1}^{\infty}$	$\prod_{n=1}^{\infty}$	$\coprod_{n=1}^{\infty}$	$\bigcap_{n=1}^{\infty}$	$\bigcup_{n=1}^{\infty}$	$\bigsqcup_{n=1}^{\infty}$
Display style	$\displaystyle\sum_{n=1}^{\infty}$	$\displaystyle\prod_{n=1}^{\infty}$	$\displaystyle\coprod_{n=1}^{\infty}$	$\displaystyle\bigcap_{n=1}^{\infty}$	$\displaystyle\bigcup_{n=1}^{\infty}$	$\displaystyle\bigsqcup_{n=1}^{\infty}$
Standard operator	`\bigodot`	`\bigoplus`	`\bigotimes`	`\biguplus`	`\bigwedge`	`\bigvee`
Smart operator	`\Odot`	`\Oplus`	`\Otimes`	`\Uplus`	`\Wedge`	`\Vee`
Inline style	$\bigodot_{n=1}^{\infty}$	$\bigoplus_{n=1}^{\infty}$	$\bigotimes_{n=1}^{\infty}$	$\biguplus_{n=1}^{\infty}$	$\bigwedge_{n=1}^{\infty}$	$\bigvee_{n=1}^{\infty}$
Display style	$\displaystyle\bigodot_{n=1}^{\infty}$	$\displaystyle\bigoplus_{n=1}^{\infty}$	$\displaystyle\bigotimes_{n=1}^{\infty}$	$\displaystyle\biguplus_{n=1}^{\infty}$	$\displaystyle\bigwedge_{n=1}^{\infty}$	$\displaystyle\bigvee_{n=1}^{\infty}$

However, if you want something more elaborate, such as $\displaystyle\sum_{n\in\mathbb{Z}\setminus\{0\}}$, it is easier to type it out normally.

To create your own smart operator, add the following lines (for example) to the preamble:

```
\DeclareMathOperator*{\TheOp}{Sym}
\CreateSmartLargeOperator{TheOp}{SmartOp}
```

Then the command `\SmartOp{2}` will produce $\operatorname{Sym}_{n=2}^{\infty}$ in inline mode and $\displaystyle\operatorname*{Sym}_{n=2}^{\infty}$ in display mode.

The first line defines `\TheOp` as the operator denoted by Sym. The second line defines `\SmartOp` as the smart version of this operator. Note that neither argument in the second line has a backslash, this is important. And of course, the operator only renders in math mode.

2.6 Miscellaneous

Some commands have been defined to convenience typing in LaTeX. For example, the symbol $\mathbb{R}$ for real numbers is normally typed as `\mathbb{R}`, which is rather cumbersome.

$\mathbb{A}$	$\mathcal{A}$	$\overline{A}$	$\mathbf{A}$	$\widetilde{A}$
`\A`	`\As`	`\Abar`	`\Ab`	`\Atil`

The above commands have been defined similarly for all letters of the alphabet, with exceptions for pre-defined commands (e.g. `\S`, which renders as §).

The physics package defines bracket-like commands (such as `\abs` and `\norm`) that automatically resize themselves according to their content. Here, we have defined the commands `\floor` and `\ceil` with the same features.

`$\floor{4.1}$`	$\left\lfloor 4.1 \right\rfloor$	`$\ceil{7.6}$`	$\left\lceil 7.6 \right\rceil$
`$\floor{x^2}$`	$\left\lfloor x^2 \right\rfloor$	`$\ceil{x^5}$`	$\left\lceil x^5 \right\rceil$
`$\floor{\dfrac{9}{5}}$`	$\left\lfloor \dfrac{9}{5} \right\rfloor$	`$\ceil{\dfrac{3}{7}}$`	$\left\lceil \dfrac{3}{7} \right\rceil$
`$\floor*{\dfrac{x^4}{y^7}}$`	$\lfloor \dfrac{x^4}{y^7} \rfloor$	`$\ceil*{\dfrac{x^5}{y^3}}$`	$\lceil \dfrac{x^5}{y^3} \rceil$

The starred versions `\floor*` and `\ceil*` do not resize the brackets.

The physics package also defines the identical commands `\ip` and `\braket` for inner products. Here, we have redefined `\ip` to take only one argument (which may contain commas), while `\braket` is left as it is. Both commands have automatic resizing, which is suppressed with their starred versions.

`$\ip{x,y}$`	$\langle x, y \rangle$	`$\braket{x}{y}$`	$\langle x \vert y \rangle$

The amsmath package provides various macros for trigonometric functions, such as `\sin` for sin. The physics package overwrites these macros and provides additional ones, all with automatic resizing of brackets (unless the package is loaded with the option `notrig`). Here we have extended this list of macros to the following (all with automatic resizing):

sin cos tan cot sec csc
asin acos atan acot asec acsc
arcsin arccos arctan arccot arcsec arccsc
sinh cosh tanh coth sech csch
asinh acosh atanh acoth asech acsch
arsinh arcosh artanh arcoth arsech arcsch
arcsinh arccosh arctanh arccoth arcsech arccsch
sen tg ctg senh tgh ctgh
Re Im arg Arg im ker

LaTeX provides a command `\iff` which renders as $\iff$ (identical to `\Longleftrightarrow`). The amsmath package also provides two commands, `\implies` which renders as $\implies$ (identical to `\Longrightarrow`), and `\impliedby` which renders as $\impliedby$ (identical to `\Longleftarrow`).

Here, these three commands are redefined to produce shorter sets of arrows, $\Leftrightarrow$, $\Rightarrow$ and $\Leftarrow$, identical to `\Leftrightarrow`, `\Rightarrow` and `\Leftarrow` respectively. The original (long) arrows can be obtained using the starred versions `\iff*`, `\implies*` and `\impliedby*`.

Unlike the original commands, these can also be used in text mode, however the spacing needs to be accounted for depending on the text/math on either side of the arrows.

`a\iff b\implies c\impliedby d`	a⇔b⇒c⇐d
`a\iff* b\implies* c\impliedby* d`	a⟺ b⟹ c⟸ d
`$a\iff b\implies c\impliedby d$`	$a \Leftrightarrow b \Rightarrow c \Leftarrow d$
`$a\iff* b\implies* c\impliedby* d$`	$a \iff b \implies c \impliedby d$

References

[1] Evan Chen. *LaTeX FAQs*. URL: https://web.evanchen.cc/faq-latex.html.

[2] Adam Gleave. *Writing Beautifully in LaTeX*. URL: https://www.gleave.me/post/latex-design-patterns/.

[3] Joseph L. Taylor. *Complex Variables*. AMS, 2011. ISBN: 978-0-8218-6901-7.

If you're struggling to find the LaTeX command for a symbol, go to `detexify.kirelabs.org`.

If you're looking for ideas/inspiration for your LaTeX documents, check out these websites:

- `castel.dev`
- `dec41.user.srcf.net`
- `youtube.com/@DrTrefor`
- `coffeeintotheorems.com`
- `physicsread.com`
- `latextemplates.com`
- `ralphs16.github.io`
- `github.com/vhbelvadi`

As well as `tex.stackexchange.com` and `stackoverflow.com`, where you can ask and answer questions.

And of course, the Overleaf gallery: `overleaf.com/latex/templates`.

Have fun! 😎