

# User's manual for the halloweenmath package

G. Mezzetti

November 1, 2019

## Contents

<b>1</b>	<b>Package loading</b>	<b>2</b>
<b>2</b>	<b>Package usage</b>	<b>2</b>
2.1	Ordinary symbols . . . . .	2
2.2	Binary operators . . . . .	2
2.3	“Large” operators . . . . .	2
2.4	“Fraction-like” symbols . . . . .	3
2.5	“Arrow-like” symbols . . . . .	3
2.6	Extensible “arrow-like” symbols . . . . .	3
2.7	Extensible “over-/under-arrow-like” symbols . . . . .	5
2.8	Script-style versions of <code>amsmath</code> ’s over/under arrows . . . . .	5
<b>3</b>	<b>Examples of use</b>	<b>8</b>
3.1	Applying black magic . . . . .	8
3.2	Monoids . . . . .	8
3.3	Applications induced on power sets . . . . .	9
3.4	A comprehensive test . . . . .	10

## List of Tables

1	Ordinary symbols . . . . .	2
2	Binary operators . . . . .	2
3	“Large” operators . . . . .	3
4	“Fraction-like” symbols . . . . .	3
5	“Arrow-like” symbols . . . . .	3
6	Extensible “arrow-like” symbols . . . . .	4
7	Extensible “over-/under-arrow-like” symbols . . . . .	6
8	Over/under bats . . . . .	6
9	Extensible over/under arrows with reduced size . . . . .	7







 <code>\mathleftghost</code>	 <code>\mathghost</code>	 <code>\mathrightghost</code>
 <code>\mathleftbat</code>	 <code>\mathbat</code>	 <code>\mathrightbat</code>

Table 1: Ordinary symbols


 <code>\pumpkin</code>	 <code>\skull</code>
---	---

Table 2: Binary operators

## 1 Package loading

Load the `halloweenmath` package as any other  $\text{\LaTeX} 2_{\epsilon}$  package, that is, via the usual `\usepackage` declaration:

```
\usepackage{halloweenmath}
```

Note that the `halloweenmath` package requires the `amsmath` package, and loads it (without specifying any option) if it is not already loaded. If you want to pass options to `amsmath`, load it before `halloweenmath`.

The `halloweenmath` package defines no options by itself; nevertheless, it does honor the `[no]sumlimits` options from the `amsmath` package.

## 2 Package usage

The `halloweenmath` package defines a handful of commands, all of which are intended for use *in mathematical mode*, where they yield some kind of symbol that draws from the classic Halloween-related iconography (pumpkins, witches, ghosts, bats, and so on). Below, these symbols are grouped according to their mathematical “rôle” (ordinary symbols, binary operators, arrows...).

### 2.1 Ordinary symbols

Table 1 lists the ordinary symbols provided by the `halloweenmath` package.

### 2.2 Binary operators

Table 2 lists the binary operators available. Note that each binary operator has an associated “large” operator (see subsection 2.3).

### 2.3 “Large” operators

Table 3 lists the “large” operators. Each of them is depicted in two variants: the variant used for in-line math and the variant used for displayed formulas. In the table, besides the “large” operators called `\bigpumpkin`<sup>1</sup> and `\bigskull`, which are correlated to the binary operators `\pumpkin` and `\skull`, respectively, we find the commands `\mathwitch` and `\reversemathwitch`: note how these two last command have a `*`-form that adds a black cat on the broomstick.

All the “large” operators listed in table 3 honor the `[no]sumlimits` options from the `amsmath` package.

---

<sup>1</sup>As a homage to Linus van Pelt, `\greatpumpkin` is defined as synonym of `\bigpumpkin`.













		<code>\mathwitch</code>			<code>\reversemathwitch</code>
		<code>\mathwitch*</code>			<code>\reversemathwitch*</code>
		<code>\bigpumpkin<sup>1</sup></code>			<code>\bigskull</code>

Table 3: “Large” operators

		<code>\mathcloud</code>			<code>\reversemathcloud</code>
---	---	-------------------------	---	---	--------------------------------

Table 4: “Fraction-like” symbols

$\leftharpoonup$	<code>\leftbroom</code>	$\rightharpoonup$	<code>\rightbroom</code>
$\leftthreetimes$	<code>\hmleftpitchfork</code>	$\rightthreetimes$	<code>\hmrightpitchfork</code>

Table 5: “Arrow-like” symbols

## 2.4 “Fraction-like” symbols

There are also two commands, listed on table 4, that yield symbols that are somewhat similar to fractions, in that they grow in size when they are typeset in display style.<sup>2</sup> They are intended to denote an unspecified subformula that appears as a part of a larger one.

## 2.5 “Arrow-like” symbols

As we’ll see in subsection 2.6, the `halloweenmath` package provides a series of commands whose usage parallels that of “extensible arrows” like `\xrightarrow` or `\xleftarrow`; but the symbols that those commands yield when used with an empty argument turn out to be too short, and it is for this reason that the `halloweenmath` package also offers you the four commands you can see in table 5: they produce brooms, or pitchforks, having fixed length, which is approximately the same size of a `\longrightarrow` ( $\longrightarrow$ ). All of these symbols are treated as relations.

## 2.6 Extensible “arrow-like” symbols

You are probably already familiar with the “extensible arrows” like  $\xrightarrow{abc}$  and  $\xleftarrow{abc}$ ; for example, you probably know that the input

```
\[
  \bigoplus_{i=1}^n A_i \xrightarrow{f_1+\dots+f_n} B
\]
```

produces this result:

$$\bigoplus_{i=1}^n A_i \xrightarrow{f_1+\dots+f_n} B$$

<sup>2</sup>Another  $\mathrm{T\!E\!X}$ nicol aspect of these commands is that they yield an atom of type Inner.

$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{witch}}$	<code>\xleftwitchonbroom{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{witch}}$	<code>\xrightwitchonbroom{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{witch}} \text{cat}$	<code>\xleftwitchonbroom*{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{witch}} \text{cat}$	<code>\xrightwitchonbroom*{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{pitchfork}}$	<code>\xleftwitchonpitchfork{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{pitchfork}}$	<code>\xrightwitchonpitchfork{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{pitchfork}} \text{cat}$	<code>\xleftwitchonpitchfork*{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{pitchfork}} \text{cat}$	<code>\xrightwitchonpitchfork*{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{broom}}$	<code>\xleftbroom{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{broom}}$	<code>\xrightbroom{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{pitchfork}}$	<code>\xleftpitchfork{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{pitchfork}}$	<code>\xrightpitchfork{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{ghost}}$	<code>\xleftswishingghost{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{ghost}}$	<code>\xrightswishingghost{abc\dots z}</code>
$\overrightarrow{\text{abc}\dots z} \xleftarrow{\text{bat}}$	<code>\xleftflutteringbat{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z} \xrightarrow{\text{bat}}$	<code>\xrightflutteringbat{abc\dots z}</code>

Table 6: Extensible “arrow-like” symbols

The `halloweenmath` package features a whole assortment of extensible symbols of this kind, which are listed in table 6. For example, you could say

```
\[
  G \xrightswishingghost{h_{1}+\dots+h_{n}}
    \bigpumpkin_{t=1}^n S_t
\]
```

to get the following in print:

$$G \xrightswishingghost{\bigpumpkin_{t=1}^n S_t}$$

More generally, exactly as the commands `\leftarrow` and `\rightarrow`, on which they are modeled, all the commands listed in table 6 take one optional argument, in which you can specify a subscript, and one mandatory argument, where a—possibly empty—superscript must be indicated. For example,

```
\[
  A \xrightwitchonbroom*[abc\dots z]{f_{1}+\dots+f_{n}} B
    \xrightwitchonbroom*{f_{1}+\dots+f_{n}} C
    \xrightwitchonbroom*[abc\dots z]{} D
\]
```

results in

$$A \xrightarrow[\text{abc}\dots z]{f_1+\dots+f_n} B \xrightarrow{f_1+\dots+f_n} C \xrightarrow{\text{abc}\dots z} D$$

Note that, also in this family of symbols, the commands that involve a witch all provide a `*`-form that adds a cat on the broom (or pitchfork).

The commands listed above should not be confused with those presented in subsection 2.7.

## 2.7 Extensible “over-/under-arrow-like” symbols

The commands dealt with in subsection 2.6 typeset an extensible “arrow-like” symbol having some math above or below it. But the `amsmath` package also provides commands that act the other way around, that is, they put an arrow over, or under, some math, as in the case of

```
\overrightarrow{x_1+\dots+x_n}
```

that yields  $\overrightarrow{x_1 + \dots + x_n}$ . The `halloweenmath` package provides a whole bunch of commands like this, which are listed in table 7, and which all share the same syntax as the `\overrightarrow` command.

Although they are not extensible, and are thus more similar to math accents, we have chosen to include in this subsection also the commands listed in table 8. They typeset a subformula either surmounted by the bat produced by `\mathbat`, or with that symbol underneath. Their normal (*i.e.*, unstarred) form pretends that the bat has zero width (but some height), whereas the starred variant takes the actual width of the bat into account; for example, given the input

```
\begin{align*}
& \&x+y+z \&\& x+y+z \&\& \\
& \&x+\overbat{y}+z \&\& x+\overbat*{y}+z
\end{align*}
```

compare the spacing you get in the two columns of the output:

$$\begin{array}{cc} x + y + z & x + y + z \\ x + \overbat{y} + z & x + \overbat*{y} + z \end{array}$$

## 2.8 Script-style versions of `amsmath`’s over/under arrows

The commands listed in table 9 all produce an output similar to that of the corresponding `amsmath`’s command having the same name, but stripped of the `script` substring, with the only difference that the size of the arrow is smaller. More precisely, they use for the arrow the relative script size of the current size (that is, of the size in which their argument is typeset). For example, whilst `\overrightarrow{x+y+z}` yields  $\overrightarrow{x + y + z}$ , `\overscriptrightarrow{x+y+z}` results in  $x + y + \overscript{z}$  (do you see the difference?), which, in the author’s humble opinion, looks *much* better.

$\overleftarrow{\text{abc}\dots z}$	<code>\overleftwitchonbroom{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightwitchonbroom{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftwitchonbroom*{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightwitchonbroom*{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftwitchonpitchfork{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightwitchonpitchfork{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftwitchonpitchfork*{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightwitchonpitchfork*{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftbroom{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightbroom{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overscriptleftbroom{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overscriptrightbroom{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftpitchfork{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightpitchfork{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overscriptleftpitchfork{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overscriptrightpitchfork{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftswishingghost{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightswishingghost{abc\dots z}</code>
$\overleftarrow{\text{abc}\dots z}$	<code>\overleftflutteringbat{abc\dots z}</code>	$\overrightarrow{\text{abc}\dots z}$	<code>\overrightflutteringbat{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftwitchonbroom{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightwitchonbroom{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftwitchonbroom*{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightwitchonbroom*{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftwitchonpitchfork{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightwitchonpitchfork{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftwitchonpitchfork*{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightwitchonpitchfork*{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftbroom{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightbroom{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underscriptleftbroom{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underscriptrightbroom{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftpitchfork{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightpitchfork{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underscriptleftpitchfork{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underscriptrightpitchfork{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftswishingghost{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightswishingghost{abc\dots z}</code>
$\underleftarrow{\text{abc}\dots z}$	<code>\underleftflutteringbat{abc\dots z}</code>	$\underrightarrow{\text{abc}\dots z}$	<code>\underrightflutteringbat{abc\dots z}</code>

Table 7: Extensible “over-/under-arrow-like” symbols

$\overbat{xyz}$	<code>\overbat{xyz}</code>	$\underbat{xyz}$	<code>\underbat{xyz}</code>
-----------------	----------------------------	------------------	-----------------------------

Table 8: Over/under bats

$\overleftarrow{abc\dots z}$	<code>\overscriptleftarrow{abc\dots z}</code>	$\underline{abc\dots z}$	<code>\underscriptleftarrow{abc\dots z}</code>
$\overrightarrow{abc\dots z}$	<code>\overscriptrightarrow{abc\dots z}</code>	$\underline{abc\dots z}$	<code>\underscriptrightarrow{abc\dots z}</code>
$\overleftrightarrow{abc\dots z}$	<code>\overscriptleftrightarrow{abc\dots z}</code>	$\underline{abc\dots z}$	<code>\underscriptleftrightarrow{abc\dots z}</code>

Table 9: Extensible over/under arrows with reduced size

### 3 Examples of use

This section illustrates the use of the commands provided by the `halloweenmath` package: by reading the source code for this document, you can see how the output presented below can be obtained.

#### 3.1 Applying black magic

The  $\mathfrak{W}$  symbol was invented with the intent to provide a notation for the operation of applying black magic to a formula. Its applications range from simple reductions sometimes made by certain undergraduate freshmen, as in

$$\mathfrak{W} 2 \sin \frac{x}{2} = \sin x$$

to key steps that permit to simplify greatly the proof of an otherwise totally impenetrable theorem, for example

$$\mathfrak{W} \left( \sup \{ p \in \mathbb{N} \mid p \text{ and } p+2 \text{ are both prime} \} \right) = \infty$$

Another way of denoting the same operation is to place the broom and the witch *over* the relevant subformula:

$$\overbrace{\sup \{ p \in \mathbb{N} \mid p \text{ and } p+2 \text{ are both prime} \}}^{\mathfrak{W}} = \infty$$

Different types of magic, that you might want to apply to a given formula, can be distinguished by adding a black cat on the broom: for example, a student could claim that

$$\mathfrak{W}^{\text{cat}} 2x \sin x = 2 \sin x^2$$

whereas, for another student,

$$\mathfrak{W}^{\text{cat}} 2x \sin x = \sin 3x$$

#### 3.2 Monoids

Let  $X$  be a non-empty set, and suppose there exists a map

$$X \times X \longrightarrow X, \quad (x, y) \longmapsto P(x, y) = x \mathfrak{O} y \quad (1)$$

Suppose furthermore that this map satisfies the **associative property**

$$\forall x \in X, \forall y \in X, \forall z \in X \quad x \mathfrak{O} (y \mathfrak{O} z) = (x \mathfrak{O} y) \mathfrak{O} z \quad (2)$$

Then, the pair  $(X, \mathfrak{O})$  is called a **semigroup**, and  $\mathfrak{O}$  denotes its **operation**. If, in addition, there exists in  $X$  an element  $\mathfrak{I}$  with the property that

$$\forall x \in X \quad \mathfrak{I} \mathfrak{O} x = x = x \mathfrak{O} \mathfrak{I} \quad (3)$$



the triple  $(X, \odot, \mathbb{1})$  is called a **monoid**, and the element  $\mathbb{1}$  is called its **unit**. It is immediate to prove that the unit of a monoid is unique: indeed, if  $\mathbb{1}'$  is another element of  $X$  having the property (3), then

$$\mathbb{1}' = \mathbb{1}' \odot \mathbb{1} = \mathbb{1}$$

(the first equality holds because  $\mathbb{1}' \in X$  and  $\mathbb{1}$  satisfies (3), and the second because  $\mathbb{1} \in X$  and  $\mathbb{1}'$  satisfies (3)).

Let  $(X, \odot, \mathbb{1})$  be a monoid. Since its operation  $\odot$  is associative, we may set, for  $x, y, z \in X$ ,

$$x \odot y \odot z =_{\text{def}} (x \odot y) \odot z = x \odot (y \odot z)$$

More generally, since the order in which the operations are performed doesn't matter, given  $n$  elements  $x_1, \dots, x_n \in X$ , with  $n \in \mathbb{N}$ , the result of

$$\bigodot_{i=1}^n x_i = x_1 \odot \dots \odot x_n$$

is unambiguously defined (it being  $\mathbb{1}$  if  $n = 0$ ).

A monoid  $(X, \odot, \mathbb{1})$  is said to be **commutative** if

$$\forall x \in X, \forall y \in X \quad x \odot y = y \odot x \quad (4)$$

In this case, even the order of the *operands* becomes irrelevant, so that, for any finite (possibly empty) set  $F$ , the notation  $\bigodot_{i \in F} x_i$  also acquires a meaning.

### 3.3 Applications induced on power sets

If  $X$  is a set, we'll denote by  $\wp(X)$  the set of all subsets of  $X$ , that is

$$\wp(X) = \{ S : S \subseteq X \}$$

Let  $f: A \rightarrow B$  a function. Starting from  $f$ , we can define two other functions  $f^{\rightsquigarrow}: \wp(A) \rightarrow \wp(B)$  and  $f^{\smile}: \wp(B) \rightarrow \wp(A)$  in the following way:

$$\text{for } X \subseteq A, \quad f^{\rightsquigarrow}(X) = \{ f(x) : x \in X \} \quad (5)$$

$$\text{for } Y \subseteq B, \quad f^{\smile}(Y) = \{ x \in A : f(x) \in Y \} \quad (6)$$

In the case of functions with long names, or with long descriptions, we'll also use a notation like  $\overline{f_1 + \dots + f_n}$  to mean the same thing as  $(f_1 + \dots + f_n)^{\rightsquigarrow}$ .

For example,

$$\begin{aligned} \overline{\sin}^{\rightsquigarrow}(\mathbb{R}) &= [-1, 1] \\ \overline{\sin}^{\smile}([0, \pi]) &= [0, 1] \\ \overline{\arcsin}^{\smile}\left([0, \frac{\pi}{2}]\right) &= [0, 1] \\ \overline{\sin + \cos}^{\rightsquigarrow}(\mathbb{R}) &= [-\sqrt{2}, \sqrt{2}] \\ \overline{\log}^{\rightsquigarrow}([-\infty, 0]) &= ]0, 1] \end{aligned}$$

### 3.4 A comprehensive test

A comparison between the “standard” and the “script” extensible over/under arrows:

$$\begin{array}{l} \overrightarrow{f_1 + \cdots + f_n} \neq \overrightarrow{f_1 + \cdots + f_n} \\ \overleftarrow{f_1 + \cdots + f_n} \neq \overleftarrow{f_1 + \cdots + f_n} \\ \overleftrightarrow{f_1 + \cdots + f_n} \neq \overleftrightarrow{f_1 + \cdots + f_n} \\ \underline{\overrightarrow{f_1 + \cdots + f_n}} \neq \underline{\overrightarrow{f_1 + \cdots + f_n}} \\ \overleftarrow{\underline{f_1 + \cdots + f_n}} \neq \overleftarrow{\underline{f_1 + \cdots + f_n}} \\ \overleftrightarrow{\underline{f_1 + \cdots + f_n}} \neq \overleftrightarrow{\underline{f_1 + \cdots + f_n}} \end{array}$$

A reduction my students are likely to make:

$$\text{☞} \frac{\sin x}{s} = x \text{ in}$$

The same reduction as an in-line formula:  $\text{☞} \frac{\sin x}{s} = x \text{ in.}$

Now with limits:

$$\text{☞}_{i=1}^n \frac{i\text{-th magic term}}{2^i\text{-th wizardry}}$$

And repeated in-line:  $\text{☞}_{i=1}^n x_i y_i.$

The bold math version is honored:

$$\text{☞} \left\langle \begin{array}{c} \text{something terribly} \\ \text{complicated} \end{array} \right\rangle = 0$$

Compare it with normal math:

$$\text{☞} \left\langle \begin{array}{c} \text{something terribly} \\ \text{complicated} \end{array} \right\rangle = 0$$

In-line math comparison:  $\text{☞} f(x)$  versus  $\text{☞} f(x).$

There is also a left-facing witch:

$$\text{☞} \frac{\sin x}{s} = x \text{ in}$$

And here is the in-line version:  $\text{☞} \frac{\sin x}{s} = x \text{ in.}$

Test for \dots:

$$\text{☞}_{i_1=1}^{n_1} \cdots \text{☞}_{i_p=1}^{n_p} \frac{i_1\text{-th magic factor}}{2^{i_1}\text{-th wizardry}} \odot \cdots \odot \frac{i_p\text{-th magic factor}}{2^{i_p}\text{-th wizardry}}$$

And repeated in-line:  $\text{☞} \cdots \text{☞}_{i=1}^n x_i y_i.$

Now the pumpkins. First the **bold** math version::

$$\bigoplus_{h=1}^m \bigoplus_{k=1}^n P_{h,k}$$

Then the **normal** one:

$$\bigoplus_{h=1}^m \bigoplus_{k=1}^n P_{h,k}$$

In-line math comparison:  $\bigoplus_{i=1}^n P_i \neq \bigoplus_{i=1}^n P_i$  versus  $\bigoplus_{i=1}^n P_i \neq \bigoplus_{i=1}^n P_i$ .

Close test:  $\bigoplus \bigoplus$ . And against the pumpkins:  $\bigoplus \bigoplus \bigoplus \bigoplus$ .

In-line, but with `\limits`:  $\bigoplus_{h=1}^m \bigoplus_{k=1}^n P_{h,k}$ .

Binary:  $x \oplus y \neq x \oplus y$ . And in display:

$$a \oplus \frac{x \oplus y}{x \oplus y} \otimes b$$

Close test:  $\bigoplus \bigoplus$ . And with the pumpkins too:  $\bigoplus \bigoplus \bigoplus \bigoplus$ .

In general,

$$\bigoplus_{i=1}^n P_i = P_1 \oplus \cdots \oplus P_n$$

**The same in bold:**

$$\bigoplus_{i=1}^n P_i = P_1 \oplus \cdots \oplus P_n$$

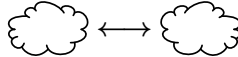
Other styles:  $\frac{x \oplus y}{2}$ , exponent  $Z^\oplus$ , subscript  $W_{x \oplus y}$ , double script  $2^{t_{x \oplus y}}$ .

Clouds. A hypothetical identity:  $\frac{\sin^2 x + \cos^2 x}{\cos^2 x} = \text{cloud}$ . Now the same identity set in display:

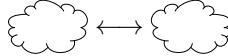
$$\frac{\sin^2 x + \cos^2 x}{\cos^2 x} = \text{cloud}$$

Now in smaller size:  $\frac{\sin x + \cos x}{\text{cloud}} = 1$ .

Specular clouds, **bold**...



...and in **normal** math.



In-line math comparison:  $\text{cloud} \leftrightarrow \text{cloud}$  versus  $\text{cloud} \leftrightarrow \text{cloud}$ . Abutting:  $\text{cloud}\text{cloud}$ .

Ghosts:  $\text{ghost}\text{ghost}\text{ghost}\text{ghost}\text{ghost}$ . Now with letters:  $H\text{ghost}H\text{ghost}h\text{ghost}ab\text{ghost}f\text{ghost}wxy\text{ghost}$ , and also  $2\text{ghost}^3 + 5\text{ghost}^2 - 3\text{ghost}_i = 12\text{ghost}_j^4$ . Then, what about  $x^{2\text{ghost}}$  and  $z_{\text{ghost}+1} = z_{\text{ghost}}^2 + z_{\text{ghost}}$ ?

In subscripts:

$$F_{\text{ghost}+2} = F_{\text{ghost}+1} + F_{\text{ghost}}$$

$$F_{\text{ghost}+2} = F_{\text{ghost}+1} + F_{\text{ghost}}$$

Another test:  $\text{ghost}|\text{ghost}|\text{ghost}|\text{ghost}|\text{ghost}|\text{ghost}|\text{ghost}$ . We should also try this:  $\text{ghost}\text{ghost}\text{ghost}$ .

Let us now compare ghosts set in normal math  $\text{ghost}\text{ghost}\text{ghost}$  with (a few words to push the bold ghosts to the right) **ghosts like these  $\text{ghost}\text{ghost}\text{ghost}$ , which are set in bold math.**

Extensible arrows:

$$\begin{array}{c}
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D
\end{array}$$

And  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$  versus  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$ ; or  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$  versus  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$ .

Now repeat in bold:

$$\begin{array}{c}
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D
\end{array}$$

And  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$  versus  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$ ; or  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$  versus  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$ .

Hovering ghosts:  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$ . I wonder whether there is enough space left for the swishing ghost; let's try again:  $(x_1 + \dots + x_n)y = 0$ ! Yes, it looks like there is enough room, although, of course, we cannot help the line spacing going awry. Also try  $\xrightarrow{\text{ghost}}$ .

$$\begin{array}{c}
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D \\
\begin{array}{c} x_1 + \dots + x_n \\ \text{a} \end{array} \xrightarrow{\text{ghost}} A \xrightarrow{\text{ghost}} B \xrightarrow{x+z} C \xrightarrow{\text{ghost}} D
\end{array}$$

Another hovering ghost:  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$ . Lorem ipsum dolor sit amet consectetur adipiscing elit. Ulla rutrum, vel sivi sit anismus oret, rubi sitiunt silvae. Let's see how it looks like when the ghost hovers on a taller formula, as in  $\xrightarrow{\text{ghost}} H_1 \oplus \dots \oplus H_k$ . Mmm, it's suboptimal, to say the least.<sup>3</sup>

Under "arrow-like" symbols:  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$  and  $\xrightarrow{\text{ghost}} x + y + z$ . There are  $\xrightarrow{\text{ghost}} x_1 + \dots + x_n = 0$  and  $\xrightarrow{\text{ghost}} x + y + z$  as well.

<sup>3</sup>We'd better try  $\xrightarrow{\text{ghost}} y_1 + \dots + y_n$ , too; well, this one looks good!

Compare  $A \xrightarrow[x_1+\dots+x_n]{\quad} B$  with (add a few words to push it to the next line)  
**its bold version**  $A \xrightarrow[x_1+\dots+x_n]{\quad} B$ .

Bats:  $\text{Bats}$ . We are interested in seeing whether a bat affixed to a letter as an exponent causes the lines of a paragraph to be further apart than usual. Therefore, we now try  $f^{\text{bat}}$ , also **in bold**  $f^{\text{bat}}$ , then we type a few more words (just enough to obtain another typeset line or two) in order to see what happens. We need to look at the transcript file, to check the outcome of the following tracing commands.

Asymmetric bats:  $\text{Asymmetric bats}$ , and also  $\text{Asymmetric bats}$ . Exponents: this is **normal** math  $x^{\text{bat}} \odot y^{\text{bat}}$ , while **this is bold math**  $x^{\text{bat}} \odot y^{\text{bat}}$ . Do you note the difference? Let's try subscripts, too:  $f_{\text{bat}} \odot g_{\text{bat}}$  versus **bold**  $f_{\text{bat}} \odot g_{\text{bat}}$ . Now, keep on repeating some silly text, just in order to fill up the paragraph with a sufficient number of lines. Now, keep on repeating some silly text, just in order to fill up the paragraph with a sufficient number of lines. Now, keep on repeating some silly text, just in order to fill up the paragraph with a sufficient number of lines. That's enough!

Hovering bats:  $x_1 + \dots + x_n = 0$ . I wonder whether there is enough space left for the swishing bat; let's try again:  $(x_1 + \dots + x_n)y = 0$ ! Yes, it looks like there is enough room (with the usual remark about line spacing). Also try  $\text{bat}$ .

$$A \xrightarrow[x_1+\dots+x_n]{\quad} B \xrightarrow[x_1+\dots+x_n]{\quad} C \xrightarrow[x_1+\dots+x_n]{\quad} D$$

$$A \xrightarrow[x_1+\dots+x_n]{\quad} B \xrightarrow[x_1+\dots+x_n]{\quad} C \xrightarrow[x_1+\dots+x_n]{\quad} D$$

Another hovering bat:  $x_1 + \dots + x_n = 0$ .

Under "arrow-like" bats:  $x_1 + \dots + x_n = 0$  and  $x + y + z$ .

Compare  $A \xrightarrow[x_1+\dots+x_n]{\quad} B$  with (add a few words to push it to the next line)  
**its bold version**  $A \xrightarrow[x_1+\dots+x_n]{\quad} B$ .

Test for checking the placement of the formulas that go over or under the fluttering bat:

$$A \xrightarrow[\text{a long subscript}]{\text{a long superscript}} B \xrightarrow[\text{a long subscript}]{\quad} C \xrightarrow[\text{a long subscript}]{\quad} D \xrightarrow[\text{a long subscript}]{\quad} E$$

$$A \xrightarrow[\text{a long subscript}]{\text{a long superscript}} B \xrightarrow[\text{a long subscript}]{\quad} C \xrightarrow[\text{a long subscript}]{\quad} D \xrightarrow[\text{a long subscript}]{\quad} E$$

I'd say it's now OK...

Extensible arrows with pitchfork:

$$A \xrightarrow[x_1+\dots+x_n]{\quad} B \xrightarrow[x_1+\dots+x_n]{\quad} C \xrightarrow[x_1+\dots+x_n]{\quad} D$$

$$A \xrightarrow[x_1+\dots+x_n]{\quad} B \xrightarrow[x_1+\dots+x_n]{\quad} C \xrightarrow[x_1+\dots+x_n]{\quad} D$$

$$A \xrightarrow[x_1+\dots+x_n]{\quad} B \xrightarrow[x_1+\dots+x_n]{\quad} C \xrightarrow[x_1+\dots+x_n]{\quad} D$$

$$A \xrightarrow[x_1+\dots+x_n]{\quad} B \xrightarrow[x_1+\dots+x_n]{\quad} C \xrightarrow[x_1+\dots+x_n]{\quad} D$$

And  $\frac{\exists}{x_1 + \dots + x_n} = 0$  versus  $\frac{\exists}{x_1 + \dots + x_n} = 0$ ; or  $\frac{\exists}{x_1 + \dots + x_n} \in 0$  versus  $\frac{\exists}{x_1 + \dots + x_n} \in 0$ . There are  $x_1 + \dots + x_n = 0$  and  $x + y + z$  as well.

Now again, but all in boldface:

$$\begin{array}{l}
 A \frac{x_1 + \dots + x_n}{a} \exists B \frac{x + z}{a} \exists C \frac{x}{a} \exists D \\
 A \frac{x_1 + \dots + x_n}{a} \exists B \frac{x + z}{a} \exists C \frac{x}{a} \exists D \\
 A \frac{x_1 + \dots + x_n}{a} \in B \frac{x + z}{a} \in C \frac{x}{a} \in D \\
 A \frac{x_1 + \dots + x_n}{a} \in B \frac{x + z}{a} \in C \frac{x}{a} \in D
 \end{array}$$

And  $\frac{\exists}{x_1 + \dots + x_n} = 0$  versus  $\frac{\exists}{x_1 + \dots + x_n} = 0$ ; or  $\frac{\exists}{x_1 + \dots + x_n} \in 0$  versus  $\frac{\exists}{x_1 + \dots + x_n} \in 0$ . There are  $x_1 + \dots + x_n = 0$  and  $x + y + z$  as well.

The big table of the rest:

$A \frac{x_1 + \dots + x_n}{a} \exists B$	$\frac{x_1 + \dots + x_n}{a} \exists = 0$	$\frac{x_1 + \dots + x_n}{a} \exists = 0$
	$\frac{x_1 + \dots + x_n}{a} \exists = 0$	$\frac{x_1 + \dots + x_n}{a} \exists = 0$
$A \frac{x_1 + \dots + x_n}{a} \in B$	$\frac{x_1 + \dots + x_n}{a} \in = 0$	$\frac{x_1 + \dots + x_n}{a} \in = 0$
	$\frac{x_1 + \dots + x_n}{a} \in = 0$	$\frac{x_1 + \dots + x_n}{a} \in = 0$
$A \frac{x_1 + \dots + x_n}{a} \exists B$	$\frac{x_1 + \dots + x_n}{a} \exists = 0$	$\frac{x_1 + \dots + x_n}{a} \exists = 0$
	$\frac{x_1 + \dots + x_n}{a} \exists = 0$	$\frac{x_1 + \dots + x_n}{a} \exists = 0$
$A \frac{x_1 + \dots + x_n}{a} \in B$	$\frac{x_1 + \dots + x_n}{a} \in = 0$	$\frac{x_1 + \dots + x_n}{a} \in = 0$
	$\frac{x_1 + \dots + x_n}{a} \in = 0$	$\frac{x_1 + \dots + x_n}{a} \in = 0$

Now in bold... No, please, seriously, just the examples for the minimal size: in normal math we show  $A \geq B$  and  $C \in D$  and  $\geq$  and  $\in$ , which we now repeat in bold math  $A \geq B$  and  $C \in D$  and  $\geq$  and  $\in$ . Mmmh, the minimal size seems way too narrow: is it the same for the standard arrows? Let's see:

$A \rightarrow B$	$\rightarrow$	$\rightarrow$
$A \leftarrow B$	$\leftarrow$	$\leftarrow$
$A \geq B$	$\geq$	$\geq$
$A \leq B$	$\leq$	$\leq$

Well, almost so, but the arrow tip is much more "discrete"...

To cope with this problem, `\rightbroom` and siblings have been introduced: for example,  $X \succsim Y$ .

A comparative table follows:

$A \succsim B$	$C \ni D$
$A \prec B$	$C \ni D$
$A \rightarrow B$	$C \Rightarrow D$
$A \leftarrow B$	$C \Leftarrow D$
$A \succsim B$	$C \ni D$
$A \prec B$	$C \ni D$

Finally,  $\tilde{y} + x + z = 0$  versus  $\tilde{y} + x + z = 0$ , and also note that  $\tilde{x}_2 \neq \tilde{x}_2$ . Oh, wait, we have to check **the bold version**  $\tilde{x}_2 \neq \tilde{x}_2$  too!

We’ve now gotten to skulls.

$$A \text{ ——— } B \odot C$$

Skulls are similar to pumpkins, and thus to `\oplus`:

$$\begin{aligned} H_1 \odot \cdots \odot H_n \\ H_1 \oplus \cdots \oplus H_n \\ H_1 \odot \cdots \odot H_n \end{aligned}$$

As you can see, though, the dimensions differ slightly:  $\odot \oplus \odot$ . Subscript:  $A_{x \odot y}$ . Now the “large” operator version:

$$\begin{aligned} \bigodot_{i=1}^n H_i &= H_1 \odot \cdots \odot H_n \\ \bigoplus_{i=1}^n H_i &= H_1 \oplus \cdots \oplus H_n \\ \bigodot_{i=1}^n H_i &= H_1 \odot \cdots \odot H_n \end{aligned}$$

In-line:  $\bigodot_{i=1}^n H_i = H_1 \odot \cdots \odot H_n$ . Example of close comparison:  $\bigoplus \bigodot \bigodot X$ . **Now repeat in bold:**  $\bigodot_{i=1}^n H_i = H_1 \odot \cdots \odot H_n$ .

Skulls look much gloomier than pumpkins: compare  $P \odot U \odot M = P$  with  $S \odot K \odot U = L \odot L$ . Why did I ever outline such a grim and dreary picture? The “large operator” variant, then, is truly dreadful! How could anybody write a formula like  $\bigodot_i \bigodot_j A_i \otimes B_j$ ? How much cheerier is  $\bigodot_i \bigodot_j A_i \otimes B_j$ ? And look at the displayed version:

$$\bigodot_{i=1}^m \bigodot_{j=1}^n A_i \otimes B_j \neq \bigodot_{i=1}^m \bigodot_{j=1}^n A_i \otimes B_j$$

Comparison between math versions:  $x \otimes y$  is normal math, **whereas  $x \otimes y$  is bold**. Similarly,  $\bigotimes_{i=1}^n K_i = L$  is normal, **but  $\bigotimes_{i=1}^n K_i = L$  is bold**. And now the displays: normal

$$\bigotimes_{i=1}^m \bigotimes_{j=1}^n A_i \otimes B_j \neq \bigotimes_{i=1}^m \bigotimes_{j=1}^n A_i \otimes B_j$$

versus **bold**

$$\bigotimes_{i=1}^m \bigotimes_{j=1}^n A_i \otimes B_j \neq \bigotimes_{i=1}^m \bigotimes_{j=1}^n A_i \otimes B_j$$

**math.** Back to the normal font.