User's Guide for the `amsmath` Package (Version 2.0)

American Mathematical Society

1999/12/13

Contents

—1—

Introduction

The amsmath package is a LaTeX package that provides miscellaneous enhancements for improving the information structure and printed output of documents that contain mathematical formulas. Readers unfamiliar with LaTeX should refer to [7]. If you have an up-to-date version of LaTeX, the amsmath package is normally provided along with it. Upgrading when a newer version of the amsmath package is released can be done via http://www.ams.org/tex/amsmath.html or ftp://ftp.ams.org/pub/tex/.

This documentation describes the features of the amsmath package and discusses how they are intended to be used. It also covers some ancillary packages:

amsbsy	amstext
amscd	amsxtra
amsopn	

These all have something to do with the contents of math formulas. For information on extra math symbols and math fonts, see [1] and http://www.ams.org/tex/amsfonts.html. For documentation of the amsthm package or AMS document classes (amsart, amsbook, etc.) see [3] or [2] and http://www.ams.org/tex/author-info.html.

If you are a long-time LaTeX user and have lots of mathematics in what you write, then you may recognize solutions for some familiar problems in this list of amsmath features:

- A convenient way to define new 'operator name' commands analogous to `\sin` and `\lim`, including proper side spacing and automatic selection of the correct font style and size (even when used in sub- or superscripts).
- Multiple substitutes for the eqnarray environment to make various kinds of equation arrangements easier to write.
- Equation numbers automatically adjust up or down to avoid overprinting on the equation contents (unlike eqnarray).
- Spacing around equals signs matches the normal spacing in the equation environment (unlike eqnarray).
- A way to produce multiline subscripts as are often used with summation or product symbols.
- An easy way to substitute a variant equation number for a given equation instead of the automatically supplied number.
- An easy way to produce subordinate equation numbers of the form (1.3a) (1.3b) (1.3c) for selected groups of equations.

The amsmath package is distributed together with some small auxiliary packages:

amsmath Primary package, provides various features for displayed equations and other mathematical constructs.

`amstext` Provides a `\text` command for typesetting a fragment of text inside a display.

`amsopn` Provides `\DeclareMathOperator` for defining new 'operator names' like `\sin` and `\lim`.

`amsbsy` For backward compatibility this package continues to exist but use of the newer `bm` package that comes with LaTeX is recommended instead.

`amscd` Provides a `CD` environment for simple commutative diagrams (no support for diagonal arrows).

`amsxtra` Provides certain odds and ends such as `\fracwithdelims` and `\accentedsymbol`, for compatibility with documents created using version 1.1.

The `amsmath` package incorporates `amstext`, `amsopn`, and `amsbsy`. The features of `amscd` and `amsxtra`, however, are available only by invoking those packages separately.

—2—

Options for the `amsmath` package

The `amsmath` package has the following options:

`centertags` (default) For a split equation, place equation numbers vertically centered on the total height of the equation.

`tbtags` 'Top-or-bottom tags': For a split equation, place equation numbers level with the last (resp. first) line, if numbers are on the right (resp. left).

`sumlimits` (default) Place the subscripts and superscripts of summation symbols above and below, in displayed equations. This option also affects other symbols of the same type—$\prod$, $\coprod$, $\bigotimes$, $\bigoplus$, and so forth—but excluding integrals (see below).

`nosumlimits` Always place the subscripts and superscripts of summation-type symbols to the side, even in displayed equations.

`intlimits` Like `sumlimits`, but for integral symbols.

`nointlimits` (default) Opposite of `intlimits`.

`namelimits` (default) Like `sumlimits`, but for certain 'operator names' such as det, inf, lim, max, min, that traditionally have subscripts placed underneath when they occur in a displayed equation.

`nonamelimits` Opposite of `namelimits`.

To use one of these package options, put the option name in the optional argument of the `\usepackage` command—e.g., `\usepackage[intlimits]{amsmath}`.

The `amsmath` package also recognizes the following options which are normally selected (implicitly or explicitly) through the `\documentclass` command, and thus need not be repeated in the option list of the `\usepackage{amsmath}` statement.

`leqno` Place equation numbers on the left.

`reqno` Place equation numbers on the right.

`fleqn` Position equations at a fixed indent from the left margin rather than centered in the text column.

—3—

Displayed equations

3.1 Introduction

The `amsmath` package provides a number of additional displayed equation structures beyond the ones provided in basic LaTeX. The augmented set includes:

```
equation    equation*    align     align*
gather      gather*      flalign   flalign*
multline    multline*    alignat   alignat*
split
```

(Although the standard `eqnarray` environment remains available, it is better to use `align` or `equation`+`split` instead.)

Except for `split`, each environment has both starred and unstarred forms, where the unstarred forms have automatic numbering using LaTeX's `equation` counter. You can suppress the number on any particular line by putting `\notag` before the `\\`; you can also override it with a tag of your own using `\tag{`⟨*label*⟩`}`, where ⟨*label*⟩ means arbitrary text such as `$*$` or `ii` used to "number" the equation. There is also a `\tag*` command that causes the text you supply to be typeset literally, without adding parentheses around it. `\tag` and `\tag*` can also be used within the unnumbered versions of all the `amsmath` alignment structures. Some examples of the use of `\tag` may be found in the sample files `testmath.tex` and `subeqn.tex` provided with the `amsmath` package.

The `split` environment is a special subordinate form that is used only *inside* one of the others. It cannot be used inside `multline`, however.

In the structures that do alignment (`split`, `align` and variants), relation symbols have an `&` before them but not after—unlike `eqnarray`. Putting the `&` after the relation symbol will interfere with the normal spacing; it has to go before.

3.2 Single equations

The `equation` environment is for a single equation with an automatically generated number. The `equation*` environment is the same except for omitting the number.[1]

[1] Basic LaTeX doesn't provide an `equation*` environment, but rather a functionally equivalent environment named `displaymath`.

Table 3.1. Comparison of displayed equation environments (vertical lines indicating nominal margins)

```
\begin{equation*}
a=b
\end{equation*}
```

$$a = b$$

```
\begin{equation}
a=b
\end{equation}
```

$$a = b \tag{1}$$

```
\begin{equation}\label{xx}
\begin{split}
a& =b+c-d\\
 & \quad +e-f\\
 & =g+h\\
 & =i
\end{split}
\end{equation}
```

$$\begin{split} a &= b + c - d \\ &\quad + e - f \\ &= g + h \\ &= i \end{split} \tag{2}$$

```
\begin{multline}
a+b+c+d+e+f\\
+i+j+k+l+m+n
\end{multline}
```

$$\begin{multline} a + b + c + d + e + f \\ + i + j + k + l + m + n \tag{3} \end{multline}$$

```
\begin{gather}
a_1=b_1+c_1\\
a_2=b_2+c_2-d_2+e_2
\end{gather}
```

$$\begin{gather} a_1 = b_1 + c_1 \tag{4} \\ a_2 = b_2 + c_2 - d_2 + e_2 \tag{5} \end{gather}$$

```
\begin{align}
a_1& =b_1+c_1\\
a_2& =b_2+c_2-d_2+e_2
\end{align}
```

$$\begin{align} a_1 &= b_1 + c_1 \tag{6} \\ a_2 &= b_2 + c_2 - d_2 + e_2 \tag{7} \end{align}$$

```
\begin{align}
a_{11}& =b_{11}&
  a_{12}& =b_{12}\\
a_{21}& =b_{21}&
  a_{22}& =b_{22}+c_{22}
\end{align}
```

$$\begin{align} a_{11} &= b_{11} & a_{12} &= b_{12} \tag{8} \\ a_{21} &= b_{21} & a_{22} &= b_{22} + c_{22} \tag{9} \end{align}$$

```
\begin{flalign*}
a_{11}& =b_{11}&
  a_{12}& =b_{12}\\
a_{21}& =b_{21}&
  a_{22}& =b_{22}+c_{22}
\end{flalign*}
```

$$\begin{flalign*} a_{11} &= b_{11} & a_{12} &= b_{12} \\ a_{21} &= b_{21} & a_{22} &= b_{22} + c_{22} \end{flalign*}$$

3.3 Split equations without alignment

The `multline` environment is a variation of the `equation` environment used for equations that don't fit on a single line. The first line of a `multline` will be at the left margin and the last line at the right margin, except for an indention on both sides in the amount of `\multlinegap`. Any additional lines in between will be centered independently within the display width (unless the `fleqn` option is in effect).

Like `equation`, `multline` has only a single equation number; it is centered vertically on the total height unless the `tbtags` option is in effect. (Thus none of the individual lines need to be marked with `\notag`.)

It's possible to force one of the middle lines to the left or right with commands `\shoveleft`, `\shoveright`. These commands take the entire line as an argument, up to but not including the final `\\`; for example

$$\begin{multline}\framebox[.65\columnwidth]{A}\\ \framebox[.5\columnwidth]{B}\\ \shoveright{\framebox[.55\columnwidth]{C}}\\ \framebox[.65\columnwidth]{D} \tag{3.10}\end{multline}$$

```
\begin{multline}
\framebox[.65\columnwidth]{A}\\
\framebox[.5\columnwidth]{B}\\
\shoveright{\framebox[.55\columnwidth]{C}}\\
\framebox[.65\columnwidth]{D}
\end{multline}
```

The value of `\multlinegap` can be changed with the usual LaTeX commands `\setlength` or `\addtolength`.

3.4 Split equations with alignment

Like `multline`, the `split` environment is for *single* equations that are too long to fit on one line and hence must be split into multiple lines. Unlike `multline`, however, the `split` environment provides for alignment among the split lines, using `&` to mark alignment points. Unlike the other `amsmath` equation structures, the `split` environment provides no numbering, because it is intended to be used *only inside some other displayed equation structure*, usually an `equation`, `align`, or `gather` environment, which provides the numbering. For example:

$$\begin{split}H_c&=\frac{1}{2n} \sum^n_{l=0}(-1)^{l}(n-{l})^{p-2} \sum_{l _1+\dots+ l _p=l}\prod^p_{i=1} \binom{n_i}{l _i}\\ &\quad\cdot[(n-l)-(n_i-l _i)]^{n_i-l _i}\cdot \Bigl[(n-l)^2-\sum^p_{j=1}(n_i-l _i)^2\Bigr].\end{split}\tag{3.11}$$

```
\begin{equation}\label{e:barwq}\begin{split}
H_c&=\frac{1}{2n} \sum^n_{l=0}(-1)^{l}(n-{l})^{p-2}
\sum_{l _1+\dots+ l _p=l}\prod^p_{i=1} \binom{n_i}{l _i}\\
&\quad\cdot[(n-l )-(n_i-l _i)]^{n_i-l _i}\cdot
\Bigl[(n-l )^2-\sum^p_{j=1}(n_i-l _i)^2\Bigr].
\end{split}\end{equation}
```

The `split` structure should constitute the entire body of the enclosing structure, apart from commands like `\label` that produce no visible material.

3.5 Equation groups without alignment

The `gather` environment is used for a group of consecutive equations when there is no alignment desired among them; each one is centered separately within the text width (see Table 3.1). Equations inside `gather` are separated by a `\bslash` command. Any equation in a `gather` may consist of a `\begin{split}` ... `\end{split}` structure—for example:

```
\begin{gather}
  first equation\\
  \begin{split}
    second & equation\\
           & on two lines
  \end{split}
  \\
  third equation
\end{gather}
```

3.6 Equation groups with mutual alignment

The `align` environment is used for two or more equations when vertical alignment is desired; usually binary relations such as equal signs are aligned (see Table 3.1).

To have several equation columns side-by-side, use extra ampersands to separate the columns:

$$\begin{aligned} x&=y & X&=Y & a&=b+c \qquad (3.12)\\ x'&=y' & X'&=Y' & a'&=b \qquad (3.13)\\ x+x'&=y+y' & X+X'&=Y+Y' & a'b&=c'b \qquad (3.14) \end{aligned}$$

```
\begin{align}
x&=y       & X&=Y       & a&=b+c\\
x'&=y'     & X'&=Y'     & a'&=b\\
x+x'&=y+y' & X+X'&=Y+Y' & a'b&=c'b
\end{align}
```

Line-by-line annotations on an equation can be done by judicious application of

`\text` inside an `align` environment:

$$x = y_1 - y_2 + y_3 - y_5 + y_8 - \dots \qquad \text{by (3.21)} \tag{3.15}$$
$$= y' \circ y^* \qquad \text{by (4.1)} \tag{3.16}$$
$$= y(0)y' \qquad \text{by Axiom 1.} \tag{3.17}$$

```
\begin{align}
x& = y_1-y_2+y_3-y_5+y_8-\dots
                    && \text{by \eqref{eq:C}}\\
 & = y'\circ y^*    && \text{by \eqref{eq:D}}\\
 & = y(0) y'        && \text {by Axiom 1.}
\end{align}
```

A variant environment `alignat` allows the horizontal space between equations to be explicitly specified. This environment takes one argument, the number of "equation columns": count the maximum number of `&`s in any row, add 1 and divide by 2.

$$x = y_1 - y_2 + y_3 - y_5 + y_8 - \dots \quad \text{by (3.21)} \tag{3.18}$$
$$= y' \circ y^* \quad \text{by (4.1)} \tag{3.19}$$
$$= y(0)y' \quad \text{by Axiom 1.} \tag{3.20}$$

```
\begin{alignat}{2}
x& = y_1-y_2+y_3-y_5+y_8-\dots
                  &\quad& \text{by \eqref{eq:C}}\\
 & = y'\circ y^*  && \text{by \eqref{eq:D}}\\
 & = y(0) y'      && \text {by Axiom 1.}
\end{alignat}
```

3.7 Alignment building blocks

Like `equation`, the multi-equation environments `gather`, `align`, and `alignedat` are designed to produce a structure whose width is the full line width. This means, for example, that one cannot readily add parentheses around the entire structure. But variants `gathered`, `aligned`, and `alignedat` are provided whose total width is the actual width of the contents; thus they can be used as a component in a containing expression. E.g.,

$$\left.\begin{aligned} B'&=-\partial\times E,\\ E'&=\partial\times B - 4\pi j, \end{aligned} \right\} \qquad \text{Maxwell's equations}$$

```
\begin{equation*}
\left.\begin{aligned}
  B'&=-\partial\times E,\\
  E'&=\partial\times B - 4\pi j,
\end{aligned}
\right\}
\qquad \text{Maxwell's equations}
\end{equation*}
```

Like the `array` environment, these `-ed` variants also take an optional `[t]` or `[b]` argument to specify vertical positioning.

"Cases" constructions like the following are common in mathematics:

$$P_{r-j}=\begin{cases} 0 & \text{if $r-j$ is odd,}\\ r!\,(-1)^{(r-j)/2} & \text{if $r-j$ is even.} \end{cases} \tag{3.21}$$

and in the `amsmath` package there is a `cases` environment to make them easy to write:

```
P_{r-j}=\begin{cases}
    0&  \text{if $r-j$ is odd},\\
    r!\,(-1)^{(r-j)/2}&  \text{if $r-j$ is even}.
  \end{cases}
```

Notice the use of `\text` (cf. §6) and the nested math formulas.

3.8 Adjusting tag placement

Placing equation numbers can be a rather complex problem in multiline displays. The environments of the `amsmath` package try hard to avoid overprinting an equation number on the equation contents, if necessary moving the number down or up to a separate line. Difficulties in accurately calculating the profile of an equation can occasionally result in number movement that doesn't look right. There is a `\raisetag` command provided to adjust the vertical position of the current equation number, if it has been shifted away from its normal position. To move a particular number up by six points, write `\raisetag{6pt}`. This kind of adjustment is fine tuning like line breaks and page breaks, and should therefore be left undone until your document is nearly finalized, or you may end up redoing the fine tuning several times to keep up with changing document contents.

3.9 Vertical spacing and page breaks in multiline displays

You can use the `\\[`⟨*dimension*⟩`]` command to get extra vertical space between lines in all the `amsmath` displayed equation environments, as is usual in LaTeX. When the `amsmath` package is in use page breaks between equation lines are normally disallowed; the philosophy is that page breaks in such material should receive individual attention from the author. To get an individual page break inside a particular displayed equation, a `\displaybreak` command is provided. `\displaybreak` is best placed immediately before the `\\` where it is to take effect. Like LaTeX's `\pagebreak`, `\displaybreak` takes an optional argument between 0 and 4 denoting the desirability of the pagebreak. `\displaybreak[0]` means "it is permissible to break here" without encouraging a break; `\display-break` with no optional argument is the same as `\displaybreak[4]` and forces a break.

If you prefer a strategy of letting page breaks fall where they may, even in the middle of a multi-line equation, then you might put `\allowdisplaybreaks[1]` in the preamble of your document. An optional argument 1–4 can be used for finer control: `[1]` means allow page breaks, but avoid them as much as possible; values of 2,3,4 mean increasing permissiveness. When display breaks are enabled with `\allowdisplaybreaks`, the `\\*` command can be used to prohibit a pagebreak after a given line, as usual.

Note: Certain equation environments wrap their contents in an unbreakable box, with the consequence that neither `\displaybreak` nor `\allowdisplaybreaks` will have any effect on them. These include `split`, `aligned`, `gathered`, and `alignedat`.

3.10 Interrupting a display

The command `\intertext` is used for a short interjection of one or two lines of text in the middle of a multiple-line display structure (see also the `\text` command in §6). Its salient feature is preservation of the alignment, which would not happen if you simply ended the display and then started it up again afterwards. `\intertext` may only appear right after a `\\` or `\\*` command. Notice the position of the word "and" in this example.

$$A_1 = N_0(\lambda;\Omega') - \phi(\lambda;\Omega'), \tag{3.22}$$

$$A_2 = \phi(\lambda;\Omega') - \phi(\lambda;\Omega), \tag{3.23}$$

and

$$A_3 = \mathcal{N}(\lambda;\omega). \tag{3.24}$$

```
\begin{align}
A_1&=N_0(\lambda;\Omega')-\phi(\lambda;\Omega'),\\
A_2&=\phi(\lambda;\Omega')-\phi(\lambda;\Omega),\\
\intertext{and}
A_3&=\mathcal{N}(\lambda;\omega).
\end{align}
```

3.11 Equation numbering

3.11.1 Numbering hierarchy

In LaTeX if you wanted to have equations numbered within sections—that is, have equation numbers (1.1), (1.2), ..., (2.1), (2.2), ..., in sections 1, 2, and so forth—you could redefine `\theequation` as suggested in the LaTeX manual [7, §6.3, §C.8.4]:

```
\renewcommand{\theequation}{\thesection.\arabic{equation}}
```

This works pretty well, except that the equation counter won't be reset to zero at the beginning of a new section or chapter, unless you do it yourself using `\setcounter`. To make this a little more convenient, the `amsmath` package provides a command `\numberwithin`. To have equation numbering tied to section numbering, with automatic reset of the equation counter, write

```
\numberwithin{equation}{section}
```

As its name implies, the `\numberwithin` command can be applied to any counter, not just the `equation` counter.

3.11.2 Cross references to equation numbers

To make cross-references to equations easier, an `\eqref` command is provided. This automatically supplies the parentheses around the equation number. I.e., if `\ref{abc}` produces 3.2 then `\eqref{abc}` produces (3.2).

3.11.3 Subordinate numbering sequences

The `amsmath` package provides also a `subequations` environment to make it easy to number equations in a particular group with a subordinate numbering scheme. For example

```
\begin{subequations}
...
\end{subequations}
```

causes all numbered equations within that part of the document to be numbered (4.9a) (4.9b) (4.9c) . . . , if the preceding numbered equation was (4.8). A `\label` command immediately after `\begin{subequations}` will produce a `\ref` of the parent number 4.9, not 4.9a. The counters used by the subequations environment are `parentequation` and `equation` and `\addtocounter`, `\setcounter`, `\value`, etc., can be applied as usual to those counter names. To get anything other than lowercase letters for the subordinate numbers, use standard LaTeX methods for changing numbering style [7, §6.3, §C.8.4]. For example, redefining `\theequation` as follows will produce roman numerals.

```
\begin{subequations}
\renewcommand{\theequation}{\theparentequation \roman{equation}}
...
```

—4—

Miscellaneous mathematical features

4.1 Matrices

The `amsmath` package provides some environments for matrices beyond the basic `array` environment of LaTeX. The `pmatrix`, `bmatrix`, `Bmatrix`, `vmatrix` and `Vmatrix` have (respectively) $(\,)$, $[\,]$, $\{\,\}$, $|\,|$, and $\|\,\|$ delimiters built in. For naming consistency there is a `matrix` environment sans delimiters. This is not entirely redundant with the `array` environment; the matrix environments all use more economical horizontal spacing than the rather prodigal spacing of the `array` environment. Also, unlike the `array` environment, you don't have to give column specifications for any of the matrix environments; by default you can have up to 10 centered columns.[2] (If you need left or right alignment in a column or other special formats you must resort to `array`.)

[2]More precisely: The maximum number of columns in a matrix is determined by the counter `MaxMatrixCols` (normal value = 10), which you can change if necessary using LaTeX's `\setcounter` or `\addtocounter` commands.

To produce a small matrix suitable for use in text, there is a `smallmatrix` environment (e.g., $\bigl(\begin{smallmatrix} a&b\\ c&d \end{smallmatrix}\bigr)$) that comes closer to fitting within a single text line than a normal matrix. Delimiters must be provided; there are no `p`,`b`,`B`,`v`,`V` versions of `smallmatrix`. The above example was produced by

```
\bigl( \begin{smallmatrix}
  a&b\\ c&d
\end{smallmatrix} \bigr)
```

`\hdotsfor{`⟨*number*⟩`}` produces a row of dots in a matrix spanning the given number of columns. For example,

$$\begin{matrix} a&b&c&d\\ e&\hdotsfor{3} \end{matrix}$$

```
\begin{matrix} a&b&c&d\\
e&\hdotsfor{3} \end{matrix}
```

The spacing of the dots can be varied through use of a square-bracket option, for example, `\hdotsfor[1.5]{3}`. The number in square brackets will be used as a multiplier (i.e., the normal value is 1.0).

$$\begin{pmatrix} D_1t&-a_{12}t_2&\dots&-a_{1n}t_n\\ -a_{21}t_1&D_2t&\dots&-a_{2n}t_n\\ \hdotsfor[2]{4}\\ -a_{n1}t_1&-a_{n2}t_2&\dots&D_nt\end{pmatrix}, \tag{4.1}$$

```
\begin{pmatrix} D_1t&-a_{12}t_2&\dots&-a_{1n}t_n\\
-a_{21}t_1&D_2t&\dots&-a_{2n}t_n\\
\hdotsfor[2]{4}\\
-a_{n1}t_1&-a_{n2}t_2&\dots&D_nt\end{pmatrix}
```

4.2 Math spacing commands

The `amsmath` package slightly extends the set of math spacing commands, as shown below. Both the spelled-out and abbreviated forms of these commands are robust, and they can also be used outside of math.

Abbrev.	Spelled out	Example	Abbrev.	Spelled out	Example
	no space	$\Rightarrow\Leftarrow$		no space	$\Rightarrow\Leftarrow$
`\,`	`\thinspace`	$\Rightarrow\,\Leftarrow$	`\!`	`\negthinspace`	$\Rightarrow\!\Leftarrow$
`\:`	`\medspace`	$\Rightarrow\:\Leftarrow$		`\negmedspace`	$\Rightarrow\negmedspace\Leftarrow$
`\;`	`\thickspace`	$\Rightarrow\;\Leftarrow$		`\negthickspace`	$\Rightarrow\negthickspace\Leftarrow$
	`\quad`	$\Rightarrow\quad\Leftarrow$			
	`\qquad`	$\Rightarrow\qquad\Leftarrow$			

For the greatest possible control over math spacing, use `\mspace` and 'math units'. One math unit, or `mu`, is equal to 1/18 em. Thus to get a negative `\quad` you could write `\mspace{-18.0mu}`.

4.3 Dots

For preferred placement of ellipsis dots (raised or on-line) in various contexts there is no general consensus. It may therefore be considered a matter of taste. By using the semantically oriented commands

- `\dotsc` for "dots with commas"
- `\dotsb` for "dots with binary operators/relations"
- `\dotsm` for "multiplication dots"
- `\dotsi` for "dots with integrals"
- `\dotso` for "other dots" (none of the above)

instead of `\ldots` and `\cdots`, you make it possible for your document to be adapted to different conventions on the fly, in case (for example) you have to submit it to a publisher who insists on following house tradition in this respect. The default treatment for the various kinds follows American Mathematical Society conventions:

```
Then we have the series $A_1, A_2,
\dotsc$, the regional sum $A_1
+A_2 +\dotsb $, the orthogonal
product $A_1 A_2 \dotsm $, and
the infinite integral
\[\int_{A_1}\int_{A_2}\dotsi\].
```

Then we have the series $A_1, A_2, \dotsc$, the regional sum $A_1 + A_2 + \dotsb$, the orthogonal product $A_1 A_2 \dotsm$, and the infinite integral

$$\int_{A_1}\int_{A_2}\dotsi.$$

4.4 Nonbreaking dashes

A command `\nobreakdash` is provided to suppress the possibility of a linebreak after the following hyphen or dash. For example, if you write 'pages 1–9' as `pages 1\nobreakdash--9` then a linebreak will never occur between the dash and the 9. You can also use `\nobreakdash` to prevent undesirable hyphenations in combinations like `$p$-adic`. For frequent use, it's advisable to make abbreviations, e.g.,

```
\newcommand{\p}{$p$\nobreakdash}% for "\p-adic"
\newcommand{\Ndash}{\nobreakdash--}% for "pages 1\Ndash 9"
%    For "\n dimensional" ("n-dimensional"):
\newcommand{\n}[1]{$n$\nobreakdash-\hspace{0pt}}
```

The last example shows how to prohibit a linebreak after the hyphen but allow normal hyphenation in the following word. (It suffices to add a zero-width space after the hyphen.)

4.5 Accents in math

In ordinary LaTeX the placement of the second accent in doubled math accents is often poor. With the `amsmath` package you will get improved placement of the second accent: $\hat{\hat{A}}$ (`\hat{\hat{A}}`).

The commands `\dddot` and `\ddddot` are available to produce triple and quadruple dot accents in addition to the `\dot` and `\ddot` accents already available in LaTeX.

To get a superscripted hat or tilde character, load the `amsxtra` package and use `\sphat` or `\sptilde`. Usage is `A\sphat` (note the absence of the `^` character).

To place an arbitrary symbol in math accent position, or to get under accents, see the `accents` package by Javier Bezos.

4.6 Roots

In ordinary LaTeX the placement of root indices is sometimes not so good: $\sqrt[\beta]{k}$ (`\sqrt[\beta]{k}`). In the `amsmath` package `\leftroot` and `\uproot` allow you to adjust the position of the root:

```
\sqrt[\leftroot{-2}\uproot{2}\beta]{k}
```

will move the beta up and to the right: $\sqrt[\leftroot{-2}\uproot{2}\beta]{k}$. The negative argument used with `\leftroot` moves the β to the right. The units are a small amount that is a useful size for such adjustments.

4.7 Boxed formulas

The command `\boxed` puts a box around its argument, like `\fbox` except that the contents are in math mode:

$$\boxed{\eta \leq C(\delta(\eta) + \Lambda_M(0,\delta))} \tag{4.2}$$

```
\boxed{\eta \leq C(\delta(\eta) +\Lambda_M(0,\delta))}
```

4.8 Over and under arrows

Basic LaTeX provides `\overrightarrow` and `\overleftarrow` commands. Some additional over and under arrow commands are provided by the `amsmath` package to extend the set:

```
\overleftarrow         \underleftarrow
\overrightarrow        \underrightarrow
\overleftrightarrow    \underleftrightarrow
```

4.9 Extensible arrows

`\xleftarrow` and `\xrightarrow` produce arrows that extend automatically to accommodate unusually wide subscripts or superscripts. These commands take one optional argument (the subscript) and one mandatory argument (the superscript, possibly empty):

$$A \xleftarrow{n+\mu-1} B \xrightarrow[T]{n\pm i-1} C \tag{4.3}$$

```
\xleftarrow{n+\mu-1}\quad \xrightarrow[T]{n\pm i-1}
```

4.10 Affixing symbols to other symbols

LaTeX provides `\stackrel` for placing a superscript above a binary relation. In the `amsmath` package there are somewhat more general commands, `\overset` and `\underset`, that can be used to place one symbol above or below another symbol, whether it's a relation or something else. The input `\overset{*}{X}` will place a superscript-size $*$ above the X: $\overset{*}{X}$; `\underset` is the analog for adding a symbol underneath.

See also the description of `\sideset` in §7.2.

4.11 Fractions and related constructions

4.11.1 The \frac, \dfrac, and \tfrac commands

The `\frac` command, which is in the basic command set of LaTeX, takes two arguments—numerator and denominator—and typesets them in normal fraction form. The `amsmath` package provides also `\dfrac` and `\tfrac` as convenient abbreviations for `{\displaystyle\frac ... }` and `{\textstyle\frac ... }`.

$$\frac{1}{k}\log_2 c(f)\;\tfrac{1}{k}\log_2 c(f)\;\sqrt{\frac{1}{k}\log_2 c(f)}\;\sqrt{\dfrac{1}{k}\log_2 c(f)} \tag{4.4}$$

```
\begin{equation}
\frac{1}{k}\log_2 c(f)\;\tfrac{1}{k}\log_2 c(f)\;
\sqrt{\frac{1}{k}\log_2 c(f)}\;\sqrt{\dfrac{1}{k}\log_2 c(f)}
\end{equation}
```

4.11.2 The \binom, \dbinom, and \tbinom commands

For binomial expressions such as $\binom{n}{k}$ `amsmath` has `\binom`, `\dbinom` and `\tbinom`:

$$2^k-\binom{k}{1}2^{k-1}+\binom{k}{2}2^{k-2} \tag{4.5}$$

```
2^k-\binom{k}{1}2^{k-1}+\binom{k}{2}2^{k-2}
```

4.11.3 The \genfrac command

The capabilities of `\frac`, `\binom`, and their variants are subsumed by a generalized fraction command `\genfrac` with six arguments. The last two correspond to `\frac`'s numerator and denominator; the first two are optional delimiters (as seen in `\binom`); the third is a line thickness override (`\binom` uses this to set the fraction line thickness to 0—i.e., invisible); and the fourth argument is a mathstyle override: integer values 0–3 select respectively `\displaystyle`, `\textstyle`, `\scriptstyle`, and `\scriptscriptstyle`. If the third argument is left empty, the line thickness defaults to 'normal'.

> `\genfrac{`*left-delim*`}{`*right-delim*`}{`*thickness*`}{`*mathstyle*`}`
> `{`*numerator*`}{`*denominator*`}`

To illustrate, here is how `\frac`, `\tfrac`, and `\binom` might be defined.

```
\newcommand{\frac}[2]{\genfrac{}{}{}{}{#1}{#2}}
\newcommand{\tfrac}[2]{\genfrac{}{}{}{1}{#1}{#2}}
\newcommand{\binom}[2]{\genfrac{(}{)}{0pt}{}{#1}{#2}}
```

If you find yourself repeatedly using `\genfrac` throughout a document for a particular notation, you will do yourself a favor (and your publisher) if you define a meaningfully-named abbreviation for that notation, along the lines of `\frac` and `\binom`.

The primitive generalized fraction commands `\over`, `\overwithdelims`, `\atop`, `\atopwithdelims`, `\above`, `\abovewithdelims` produce warning messages if used with the `amsmath` package, for reasons discussed in `technote.tex`.

4.12 Continued fractions

The continued fraction

$$\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\dotsb}}} \tag{4.6}$$

can be obtained by typing

```
\cfrac{1}{\sqrt{2}+
 \cfrac{1}{\sqrt{2}+
  \cfrac{1}{\sqrt{2}+\dotsb
}}}
```

This produces better-looking results than straightforward use of `\frac`. Left or right placement of any of the numerators is accomplished by using `\cfrac[l]` or `\cfrac[r]` instead of `\cfrac`.

4.13 Smash options

The command `\smash` is used to typeset a subformula with an effective height and depth of zero, which is sometimes useful in adjusting the subformula's position with respect to adjacent symbols. With the `amsmath` package `\smash` has optional arguments `t` and `b`, because occasionally it is advantageous to be able to "smash" only the top or only the bottom of something while retaining the natural depth or height. For example, when adjacent radical symbols are unevenly sized or positioned because of differences in the height and depth of their contents, `\smash` can be employed to make them more consistent. Compare $\sqrt{x}+\sqrt{y}+\sqrt{z}$ and $\sqrt{x}+\sqrt{\smash[b]{y}}+\sqrt{z}$, where the latter was produced by `$\sqrt{x} + \sqrt{\smash[b]{y}} + \sqrt{z}$`.

4.14 Delimiters

4.14.1 Delimiter sizes

The automatic delimiter sizing done by `\left` and `\right` has two limitations: First, it is applied mechanically to produce delimiters large enough to encompass the largest contained item, and second, the range of sizes is not even approximately continuous but has fairly large quantum jumps. This means that a math fragment that is infinitesimally too large for a given delimiter size will get the next larger size, a jump of 3pt or so in normal-sized text. There are two or three situations where the delimiter size is commonly adjusted, using a set of commands that have 'big' in their names.

Delimiter size	text size	`\left` `\right`	`\bigl` `\bigr`	`\Bigl` `\Bigr`	`\biggl` `\biggr`	`\Biggl` `\Biggr`
Result	$(b)(\frac{c}{d})$	$(b)\left(\frac{c}{d}\right)$	$\bigl(b\bigr)\bigl(\frac{c}{d}\bigr)$	$\Bigl(b\Bigr)\Bigl(\frac{c}{d}\Bigr)$	$\biggl(b\biggr)\biggl(\frac{c}{d}\biggr)$	$\Biggl(b\Biggr)\Biggl(\frac{c}{d}\Biggr)$

The first kind of situation is a cumulative operator with limits above and below. With `\left` and `\right` the delimiters usually turn out larger than necessary, and using the `Big` or `bigg` sizes instead gives better results:

$$\left[\sum_i a_i\left\lvert\sum_j x_{ij}\right\rvert^p\right]^{1/p} \quad\text{versus}\quad \biggl[\sum_i a_i\Bigl\lvert\sum_j x_{ij}\Bigr\rvert^p\biggr]^{1/p}$$

```
\biggl[\sum_i a_i\Bigl\lvert\sum_j x_{ij}\Bigr\rvert^p\biggr]^{1/p}
```

The second kind of situation is clustered pairs of delimiters where `\left` and `\right` make them all the same size (because that is adequate to cover the encompassed material) but what you really want is to make some of the delimiters slightly larger to make the nesting easier to see.

$$\left((a_1 b_1) - (a_2 b_2)\right) \left((a_2 b_1) + (a_1 b_2)\right) \quad\text{versus}\quad \bigl((a_1 b_1) - (a_2 b_2)\bigr) \bigl((a_2 b_1) + (a_1 b_2)\bigr)$$

```
\left((a_1 b_1) - (a_2 b_2)\right)
\left((a_2 b_1) + (a_1 b_2)\right)
\quad\text{versus}\quad
\bigl((a_1 b_1) - (a_2 b_2)\bigr)
\bigl((a_2 b_1) + (a_1 b_2)\bigr)
```

The third kind of situation is a slightly oversize object in running text, such as $\left|\frac{b'}{d'}\right|$ where the delimiters produced by `\left` and `\right` cause too much line spreading. In that case `\bigl` and `\bigr` can be used to produce delimiters that are slightly larger than the base size but still able to fit within the normal line spacing: $\bigl|\frac{b'}{d'}\bigr|$.

In ordinary LaTeX `\big`, `\bigg`, `\Big`, and `\Bigg` delimiters aren't scaled properly over the full range of LaTeX font sizes. With the `amsmath` package they are.

4.14.2 Vertical bar notations

The `amsmath` package provides commands `\lvert`, `\rvert`, `\lVert`, `\rVert` (compare `\langle`, `\rangle`) to address the problem of overloading for the vert bar character `|`. This character is currently used in LaTeX documents to represent a wide variety of mathematical objects: the 'divides' relation in a number-theory expression like $p|q$, or the absolute-value operation $|z|$, or the 'such that' condition in set notation, or the 'evaluated at' notation $f_\zeta(t)\bigr|_{t=0}$. The multiplicity of uses in itself is not so bad; what is bad, however, is that fact that not all of the uses take the same typographical treatment, and that the complex discriminatory powers of a knowledgeable reader cannot be replicated in computer processing of mathematical documents. It is recommended therefore that there should be a one-to-one correspondence in any given document between the vert bar character `|` and a selected mathematical notation, and similarly for the double-bar command `\|`. This immediately rules out the use of `|` and `\|` for delimiters, because left and right delimiters are distinct usages that do not relate in the same way to adjacent symbols; recommended practice is therefore to define suitable commands in the document preamble for any paired-delimiter use of vert bar symbols:

```
\providecommand{\abs}[1]{\lvert#1\rvert}
\providecommand{\norm}[1]{\lVert#1\rVert}
```

whereupon the document would contain `\abs{z}` to produce $|z|$ and `\norm{v}` to produce $\|v\|$.

—5—

Operator names

5.1 Defining new operator names

Math functions such as log, sin, and lim are traditionally typeset in roman type to make them visually more distinct from one-letter math variables, which are set in math italic. The more common ones have predefined names, `\log`, `\sin`, `\lim`, and so forth, but new ones come up all the time in mathematical papers, so the `amsmath` package provides a general mechanism for defining new 'operator names'. To define a math function `\xxx` to work like `\sin`, you write

```
\DeclareMathOperator{\xxx}{xxx}
```

whereupon ensuing uses of `\xxx` will produce xxx in the proper font and automatically add proper spacing on either side when necessary, so that you get $A \operatorname{xxx} B$ instead of $AxxxB$. In the second argument of `\DeclareMathOperator` (the name text), a pseudo-text mode prevails: the hyphen character `-` will print as a text hyphen rather than a minus sign and an asterisk `*` will print as a raised text asterisk instead of a centered math star. (Compare $a\text{-}b^*c$ and $a - b * c$.) But otherwise the name text is printed in math mode, so that you can use, e.g., subscripts and superscripts there.

If the new operator should have subscripts and superscripts placed in 'limits' position above and below as with lim, sup, or max, use the `*` form of the `\DeclareMathOperator` command:

```
\DeclareMathOperator*{\Lim}{Lim}
```

See also the discussion of subscript placement in Section 7.3.

The following operator names are predefined:

`\arccos`	$\arccos$	`\deg`	$\deg$	`\lg`	$\lg$	`\projlim`	$\operatorname*{proj\,lim}$
`\arcsin`	$\arcsin$	`\det`	$\det$	`\lim`	$\lim$	`\sec`	$\sec$
`\arctan`	$\arctan$	`\dim`	$\dim$	`\liminf`	$\liminf$	`\sin`	$\sin$
`\arg`	$\arg$	`\exp`	$\exp$	`\limsup`	$\limsup$	`\sinh`	$\sinh$
`\cos`	$\cos$	`\gcd`	$\gcd$	`\ln`	$\ln$	`\sup`	$\sup$
`\cosh`	$\cosh$	`\hom`	$\hom$	`\log`	$\log$	`\tan`	$\tan$
`\cot`	$\cot$	`\inf`	$\inf$	`\max`	$\max$	`\tanh`	$\tanh$
`\coth`	$\coth$	`\injlim`	$\operatorname*{inj\,lim}$	`\min`	$\min$		
`\csc`	$\csc$	`\ker`	$\ker$	`\Pr`	$\Pr$		

`\varlimsup`	$\varlimsup$	`\varinjlim`	$\varinjlim$
`\varliminf`	$\varliminf$	`\varprojlim`	$\varprojlim$

There is also a command `\operatorname` such that using

```
\operatorname{abc}
```

in a math formula is equivalent to a use of `\abc` defined by `\DeclareMathOperator`. This may be occasionally useful for constructing more complex notation or other purposes. (Use the variant `\operatorname*` to get limits.)

5.2 `\mod` and its relatives

Commands `\mod`, `\bmod`, `\pmod`, `\pod` are provided to deal with the special spacing conventions of "mod" notation. `\bmod` and `\pmod` are available in LaTeX, but with the `amsmath` package the spacing of `\pmod` will adjust to a smaller value if it's used in a non-display-mode formula. `\mod` and `\pod` are variants of `\pmod` preferred by some authors; `\mod` omits the parentheses, whereas `\pod` omits the "mod" and retains the parentheses.

$$\gcd(n,m\bmod n);\quad x\equiv y\pmod b;\quad x\equiv y\mod c;\quad x\equiv y\pod d \tag{5.1}$$

```
\gcd(n,m\bmod n);\quad x\equiv y\pmod b;
\quad x\equiv y\mod c;\quad x\equiv y\pod d
```

—6—

The `\text` command

The main use of the command `\text` is for words or phrases in a display. It is very similar to the LaTeX command `\mbox` in its effects, but has a couple of advantages. If you want a word or phrase of text in a subscript, you can type `..._{\text{word or phrase}}`, which is slightly easier than the `\mbox` equivalent: `..._{\mbox{\scriptsize word or phrase}}`. The other advantage is the more descriptive name.

$$f_{[x_{i-1},x_i]} \text{ is monotonic,}\quad i = 1,\dots,c+1 \tag{6.1}$$

```
f_{[x_{i-1},x_i]} \text{ is monotonic,}
\quad i = 1,\dots,c+1
```

—7—

Integrals and sums

7.1 Multiline subscripts and superscripts

The `\substack` command can be used to produce a multiline subscript or superscript: for example

```
\sum_{\substack{
        0\le i\le m\\
        0<j<n}}
  P(i,j)
```

$$\sum_{\substack{0\le i\le m\\ 0<j<n}} P(i,j)$$

A slightly more generalized form is the `subarray` environment which allows you to specify that each line should be left-aligned instead of centered, as here:

```
\sum_{\begin{subarray}{l}
        i\in\Lambda\\ 0<j<n
      \end{subarray}}
 P(i,j)
```

$$\sum_{\begin{subarray}{l} i\in\Lambda\\ 0<j<n \end{subarray}} P(i,j)$$

7.2 The `\sideset` command

There's also a command called `\sideset`, for a rather special purpose: putting symbols at the subscript and superscript corners of a large operator symbol such as $\sum$ or $\prod$. *Note: this command is not designed to be applied to anything other than sum-class symbols.* The prime example is the case when you want to put a prime on a sum symbol. If there are no limits above or below the sum, you could just use `\nolimits`: here's `\sum\nolimits' E_n` in display mode:

$$\sum\nolimits' E_n \tag{7.1}$$

If, however, you want not only the prime but also something below or above the sum symbol, it's not so easy—indeed, without `\sideset`, it would be downright difficult. With `\sideset`, you can write

```
\sideset{}{'}
  \sum_{n<k,\;\text{$n$ odd}} nE_n
```

$$\sideset{}{'}\sum_{n<k,\;\text{n odd}} nE_n$$

The extra pair of empty braces is explained by the fact that `\sideset` has the capability of putting an extra symbol or symbols at each corner of a large operator; to put an asterisk at each corner of a product symbol, you would type

```
\sideset{_*^*}{_*^*}\prod
```

$$\sideset{_*^*}{_*^*}\prod$$

7.3 Placement of subscripts and limits

The default positioning for subscripts depends on the base symbol involved. The default for sum-class symbols is 'displaylimits' positioning: When a sum-class symbol appears in a displayed formula, subscript and superscript are placed in 'limits' position above and below, but in an inline formula, they are placed to the side, to avoid unsightly and wasteful spreading of the surrounding text lines. The default for integral-class symbols is to have sub- and superscripts always to the side, even in displayed formulas. (See the discussion of the `intlimits` and related options in Section 2.)

Operator names such as sin or lim may have either 'displaylimits' or 'limits' positioning depending on how they were defined. The standard operator names are defined according to normal mathematical usage.

The commands `\limits` and `\nolimits` can be used to override the normal behavior of a base symbol:

$$\sum\nolimits_X, \qquad \iint\limits_A, \qquad \varliminf\nolimits_{n\to\infty}$$

To define a command whose subscripts follow the same 'displaylimits' behavior as `\sum`, put `\displaylimits` at the tail end of the definition. When multiple instances of `\limits`, `\nolimits`, or `\displaylimits` occur consecutively, the last one takes precedence.

7.4 Multiple integral signs

`\iint`, `\iiint`, and `\iiiint` give multiple integral signs with the spacing between them nicely adjusted, in both text and display style. `\idotsint` is an extension of the same idea that gives two integral signs with dots between them.

$$\iint\limits_A f(x,y)\,dx\,dy \qquad \iiint\limits_A f(x,y,z)\,dx\,dy\,dz \tag{7.2}$$

$$\iiiint\limits_A f(w,x,y,z)\,dw\,dx\,dy\,dz \qquad \idotsint\limits_A f(x_1,\dots,x_k) \tag{7.3}$$

—8—

Commutative diagrams

Some commutative diagram commands like the ones in $\mathcal{AMS}$-TEX are available as a separate package, `amscd`. For complex commutative diagrams authors will need to turn to more comprehensive packages like `kuvio` or XY-pic, but for simple diagrams without diagonal arrows the `amscd` commands may be more convenient. Here is one example.

$$\begin{CD}
S^{{\mathcal{W}}_\Lambda}\otimes T @>j>> T\\
@VVV @VV{\operatorname{End} P}V\\
(S\otimes T)/I @= (Z\otimes T)/J
\end{CD}$$

```
\begin{CD}
S^{{\mathcal{W}}_\Lambda}\otimes T   @>j>>   T\\
@VVV                                   @VV{\End P}V\\
(S\otimes T)/I                  @=      (Z\otimes T)/J
\end{CD}
```

In the `CD` environment the commands `@>>>`, `@<<<`, `@VVV`, and `@AAA` give respectively right, left, down, and up arrows. For the horizontal arrows, material between the first and second `>` or `<` symbols will be typeset as a superscript, and material between the second and third will be typeset as a subscript. Similarly, material between the first and second or second and third `A`s or `V`s of vertical arrows will be typeset as left or right "sidescripts". The commands `@=` and `@|` give horizontal and vertical double lines. A "null arrow" command `@.` can be used instead of a visible arrow to fill out an array where needed.

—9—

Using math fonts

9.1 Introduction

For more comprehensive information on font use in LaTeX, see the LaTeX font guide (`fntguide.tex`) or *The LaTeX Companion* [4]. The basic set of math font commands in LaTeX includes `\mathbf`, `\mathrm`, `\mathcal`, `\mathsf`, `\mathtt`, `\mathit`. Additional math alphabet commands such as `\mathbb` for blackboard bold, `\mathfrak` for Fraktur, and `\mathscr` for Euler script are available through the packages `amsfonts` and `euscript` (distributed separately).

9.2 Recommended use of math font commands

If you find yourself employing math font commands frequently in your document, you might wish that they had shorter names, such as `\mb` instead of `\mathbf`. Of course, there is nothing to keep you from providing such abbreviations for yourself by suitable `\newcommand` statements. But for LaTeX to provide shorter names would actually be a disservice to authors, as that would obscure a much better alternative: defining custom command names derived from the names of the underlying mathematical objects, rather than from the names of the fonts used to distinguish the objects. For example, if you are using bold to indicate vectors, then you will be better served in the long run if you define a 'vector' command instead of a 'math-bold' command:

```
\newcommand{\vect}[1]{\mathbf{#1}}
```

you can write `\vect{a} + \vect{b}` to produce $\mathbf{a} + \mathbf{b}$. If you decide several months down the road that you want to use the bold font for some other purpose, and mark vectors by a small over-arrow instead, then you can put the change into effect merely by changing the definition of `\vect`; otherwise you would have to replace all occurrences of `\mathbf` throughout your document, perhaps even needing to inspect each one to see whether it is indeed an instance of a vector.

It can also be useful to assign distinct command names for different letters of a particular font:

```
\DeclareSymbolFont{AMSb}{U}{msb}{m}{n}% or use amsfonts package
\DeclareMathSymbol{\C}{\mathalpha}{AMSb}{"43}
\DeclareMathSymbol{\R}{\mathalpha}{AMSb}{"52}
```

These statements would define the commands `\C` and `\R` to produce blackboard-bold letters from the 'AMSb' math symbols font. If you refer often to the complex numbers or real numbers in your document, you might find this method more convenient than (let's say) defining a `\field` command and writing `\field{C}`, `\field{R}`. But for maximum flexibility and control, define such a `\field` command and then define `\C` and `\R` in terms of that command:

```
\usepackage{amsfonts}% to get the \mathbb alphabet
\newcommand{\field}[1]{\mathbb{#1}}
\newcommand{\C}{\field{C}}
\newcommand{\R}{\field{R}}
```

9.3 Bold math symbols

The `\mathbf` command is commonly used to obtain bold Latin letters in math, but for most other kinds of math symbols it has no effect, or its effects depend unreliably on the set of math fonts that are in use. For example, writing

```
\Delta \mathbf{\Delta}\mathbf{+}\delta \mathbf{\delta}
```

produces $\Delta\mathbf{\Delta}+\delta\delta$; the `\mathbf` has no effect on the plus sign or the small delta.

The `amsmath` package therefore provides two additional commands, `\boldsymbol` and `\pmb`, that can be applied to other kinds of math symbols. `\boldsymbol` can be used for a math symbol that remains unaffected by `\mathbf` if (and only if) your current math font set includes a bold version of that symbol. `\pmb` can be used as a last resort for any math symbols that do not have a true bold version provided by your set of math fonts; "pmb" stands for "poor man's bold" and the command works by typesetting multiple copies of the symbol with slight offsets. The quality of the output is inferior, especially for symbols that contain any hairline strokes. When the standard default set of LaTeX math fonts are in use (Computer Modern), the only symbols that are likely to require `\pmb` are large operator symbols like `\sum`, extended delimiter symbols, or the extra math symbols provided by the `amssymb` package [1].

The following formula shows some of the results that are possible:

```
A_\infty + \pi A_0
\sim \mathbf{A}_{\boldsymbol{\infty}} \boldsymbol{+}
```

```
  \boldsymbol{\pi} \mathbf{A}_{\boldsymbol{0}}
\sim\pmb{A}_{\pmb{\infty}} \pmb{+}\pmb{\pi} \pmb{A}_{\pmb{0}}
```

$$A_\infty + \pi A_0 \sim \mathbf{A}_{\boldsymbol{\infty}} \boldsymbol{+} \boldsymbol{\pi} \mathbf{A}_{\boldsymbol{0}} \sim \boldsymbol{A}_{\boldsymbol{\infty}} \boldsymbol{+} \boldsymbol{\pi} \boldsymbol{A}_{\boldsymbol{0}}$$

If you want to use only the `\boldsymbol` command without loading the whole `amsmath` package, the `bm` package is recommended (this is a standard LaTeX package, not an AMS package; you probably have it already if you have a 1997 or newer version of LaTeX).

9.4 Italic Greek letters

For italic versions of the capital Greek letters, the following commands are provided:

`\varGamma`	$\mathit{\Gamma}$	`\varSigma`	$\mathit{\Sigma}$
`\varDelta`	$\mathit{\Delta}$	`\varUpsilon`	$\mathit{\Upsilon}$
`\varTheta`	$\mathit{\Theta}$	`\varPhi`	$\mathit{\Phi}$
`\varLambda`	$\mathit{\Lambda}$	`\varPsi`	$\mathit{\Psi}$
`\varXi`	$\mathit{\Xi}$	`\varOmega`	$\mathit{\Omega}$
`\varPi`	$\mathit{\Pi}$		

—10—

Error messages and output problems

10.1 General remarks

This is a supplement to Chapter 8 of the LaTeX manual [7] (first edition: Chapter 6). For the reader's convenience, the set of error messages discussed here overlaps somewhat with the set in that chapter, but please be aware that we don't provide exhaustive coverage here. The error messages are arranged in alphabetical order, disregarding unimportant text such as `! LaTeX Error:` at the beginning, and nonalphabetical characters such as `\`. Where examples are given, we show also the help messages that appear on screen when you respond to an error message prompt by entering `h`.

There is also a section discussing some output errors, i.e., instances where the printed document has something wrong but there was no LaTeX error during typesetting.

10.2 Error messages

`\begin{split} won't work here.`

Example:

```
! Package amsmath Error: \begin{split} won't work here.
 ...

l.8 \begin{split}
```

```
? h
\Did you forget a preceding \begin{equation}?
If not, perhaps the 'aligned' environment is what you want.
?
```

Explanation: The `split` environment does not construct a stand-alone displayed equation; it needs to be used within some other environment such as `equation` or `gather`.

`Extra & on this line`
Example:

```
! Package amsmath Error: Extra & on this line.

See the amsmath package documentation for explanation.
Type  H <return>  for immediate help.
 ...

l.9 \end{alignat}

? h
\An extra & here is so disastrous that you should probably exit
 and fix things up.
?
```

Explanation: In an `alignat` structure the number of alignment points per line is dictated by the numeric argument given after `\begin{alignat}`. If you use more alignment points in a line it is assumed that you accidentally left out a newline command `\\` and the above error is issued.

`Improper argument for math accent`
Example:

```
! Package amsmath Error: Improper argument for math accent:
(amsmath)                Extra braces must be added to
(amsmath)                prevent wrong output.

See the amsmath package documentation for explanation.
Type  H <return>  for immediate help.
 ...

l.415 \tilde k_{\lambda_j} = P_{\tilde \mathcal
                                                  {M}}
?
```

Explanation: Non-simple arguments for any LaTeX command should be enclosed in braces. In this example extra braces are needed as follows:

```
... P_{\tilde{\mathcal{M}}}
```

`Font OMX/cmex/m/n/7=cmex7 not loadable ...`
Example:

```
! Font OMX/cmex/m/n/7=cmex7 not loadable: Metric (TFM) file not found.
<to be read again>
                   relax
l.8 $a
      b+b^2$
? h
I wasn't able to read the size data for this font,
so I will ignore the font specification.
[Wizards can fix TFM files using TFtoPL/PLtoTF.]
You might try inserting a different font spec;
e.g., type `I\font<same font id>=<substitute font name>'.
?
```

Explanation: Certain extra sizes of some Computer Modern fonts that were formerly available mainly through the AMSFonts distribution are considered part of standard LaTeX (as of June 1994): `cmex7–9`, `cmmib5–9`, and `cmbsy5–9`. If these extra sizes are missing on your system, you should try first to get them from the source where you obtained LaTeX. If that fails, you could try getting the fonts from CTAN (e.g., in the form of Metafont source files, directory `/tex-archive/fonts/latex/mf`, or in PostScript Type 1 format, directory `/tex-archive/fonts/cm/ps-type1/bakoma`).

If the font name begins with `cmex`, there is a special option `cmex10` for the `amsmath` package that provides a temporary workaround. I.e., change the `\usepackage` to

```
\usepackage[cmex10]{amsmath}
```

This will force the use of the 10-point size of the `cmex` font in all cases. Depending on the contents of your document this may be adequate.

`Math formula deleted: Insufficient extension fonts`
Example:

```
! Math formula deleted: Insufficient extension fonts.
l.8 $ab+b^2$

?
```

Explanation: This usually follows a previous error `Font ... not loadable`; see the discussion of that error (above) for solutions.

`Missing number, treated as zero`
Example:

```
! Missing number, treated as zero.
<to be read again>
                   a
l.100 \end{alignat}

? h
A number should have been here; I inserted `0'.
```

```
(If you can't figure out why I needed to see a number,
look up `weird error' in the index to The TeXbook.)

?
```

Explanation: There are many possibilities that can lead to this error. However, one possibility that is relevant for the `amsmath` package is that you forgot to give the number argument of an `alignat` environment, as in:

```
\begin{alignat}
 a&  =b&    c& =d\\
a'& =b'&   c'& =d'
\end{alignat}
```

where the first line should read instead

```
\begin{alignat}{2}
```

Another possibility is that you have a left bracket character `[` following a linebreak command `\\` in a multiline construction such as `array`, `tabular`, or `eqnarray`. This will be interpreted by LaTeX as the beginning of an 'additional vertical space' request [7, §C.1.6], even if it occurs on the following line and is intended to be part of the contents. For example

```
\begin{array}
a+b\\
[f,g]\\
m+n
\end{array}
```

To prevent the error message in such a case, you can add braces as discussed in the LaTeX manual [7, §C.1.1]:

```
\begin{array}
a+b\\
{[f,g]}\\
m+n
\end{array}
```

`Missing \right. inserted`
Example:

```
! Missing \right. inserted.
<inserted text>
                \right .
l.10 \end{multline}

? h
I've inserted something that you may have forgotten.
(See the <inserted text> above.)
With luck, this will get me unwedged. But if you
really didn't forget anything, try typing `2' now; then
my insertion and my current dilemma will both disappear.
```

Explanation: This error typically arises when you try to insert a linebreak inside a `\left`-`\right` pair of delimiters in a `multline` or `split` environment:

```
\begin{multline}
AAA\left(BBB\\
  CCC\right)
\end{multline}
```

There are two possible solutions: (1) instead of using `\left` and `\right`, use 'big' delimiters of fixed size (`\bigl \bigr \biggl \biggr` ...; see §4.14.1); or (2) use null delimiters to break up the `\left`-`\right` pair into parts for each line:

```
AAA\left(BBB\right.\\
  \left.CCC\right)
```

The latter solution may result in mismatched delimiter sizes; ensuring that they match requires using `\vphantom` in the line that has the smaller delimiter (or possibly `\smash` in the line that has the larger delimiter). In the argument of `\vphantom` put a copy of the tallest element that occurs in the other line, e.g.,

```
xxx \left(\int_t yyy\right.\\
  \left.\vphantom{\int_t} zzz ... \right)
```

`Paragraph ended before \xxx was complete`
Example:

```
Runaway argument?

! Paragraph ended before \multline was complete.
<to be read again>
                   \par
l.100

? h
I suspect you've forgotten a `}', causing me to apply this
control sequence to too much text. How can we recover?
My plan is to forget the whole thing and hope for the best.
?
```

Explanation: This might be produced by a misspelling in the `\end{multline}` command, e.g.,

```
\begin{multline}
...
\end{multiline}
```

or by using abbreviations for certain environments, such as `\bal` and `\eal` for `\begin{align}` and `\end{align}`:

```
\bal
...
\eal
```

For technical reasons that kind of abbreviation does not work with the more complex displayed equation environments of the `amsmath` package (`gather`, `align`, `split`, etc.; cf. `technote.tex`).

`Runaway argument?`

See the discussion for the error message `Paragraph ended before \xxx was complete`.

`Unknown option ‘xxx’ for package ‘yyy’`

Example:

```
! LaTeX Error: Unknown option ‘intlim’ for package ‘amsmath’.
...
? h
The option ‘intlim’ was not declared in package ‘amsmath’, perhaps you
misspelled its name. Try typing  <return>  to proceed.
?
```

Explanation: This means that you misspelled the option name, or the package simply does not have an option that you expected it to have. Consult the documentation for the given package.

`Old form ‘\pmatrix’ should be \begin{pmatrix}.`

Example:

```
! Package amsmath Error: Old form ‘\pmatrix’ should be
                         \begin{pmatrix}.

See the amsmath package documentation for explanation.
Type  H <return>  for immediate help.
 ...

\pmatrix ->\left (\matrix@check \pmatrix
                                        \env@matrix
l.16 \pmatrix
             {a&b\cr c&d\cr}
? h
‘\pmatrix{...}’ is old Plain-TeX syntax whose use is
ill-advised in LaTeX.
?
```

Explanation: When the `amsmath` package is used, the old forms of `\pmatrix`, `\matrix`, and `\cases` cannot be used any longer because of naming conflicts. Their syntax did not conform with standard LaTeX syntax in any case.

`Erroneous nesting of equation structures`

Example:

```
! Package amsmath Error: Erroneous nesting of equation structures;
(amsmath)                trying to recover with ‘aligned’.

See the amsmath package documentation for explanation.
```

```
Type  H <return>  for immediate help.
 ...

l.260 \end{alignat*}
                    \end{equation*}
```

Explanation: The structures `align`, `alignat`, etc., are designed for top-level use and for the most part cannot be nested inside some other displayed equation structure. The chief exception is that `align` and most of its variants can be used inside the `gather` environment.

10.3 Warning messages

`Foreign command \over [or \atop or \above]`
Example:

```
Package amsmath Warning: Foreign command \over; \frac or \genfrac
(amsmath)                should be used instead.
```

Explanation: The primitive generalized fraction commands of TeX—`\over`, `\atop`, `\above`—are deprecated when the `amsmath` package is used because their syntax is foreign to LaTeX and `amsmath` provides native LaTeX equivalents. See `technote.tex` for further information.

`Cannot use ‘split’ here`
Example:

```
Package amsmath Warning: Cannot use ‘split’ here;
(amsmath)                trying to recover with ‘aligned’
```

Explanation: The `split` environment is designed to serve as the entire body of an equation, or an entire line of an `align` or `gather` environment. There cannot be any printed material before or after it within the same enclosing structure:

```
\begin{equation}
\left\{ % <-- Not allowed
\begin{split}
...
\end{split}
\right. % <-- Not allowed
\end{equation}
```

10.4 Wrong output

10.4.1 Section numbers 0.1, 5.1, 8.1 instead of 1, 2, 3

This most likely means that you have the arguments for `\numberwithin` in reverse order:

```
\numberwithin{section}{equation}
```

That means ‘print the section number as *equation number.section number* and reset to 1 every time an equation occurs’ when what you probably wanted was the inverse

```
\numberwithin{equation}{section}
```

10.4.2 The `\numberwithin` command had no effect on equation numbers

Are you looking at the first section in your document? Check the section numbers elsewhere to see if the problem is the one described in §10.4.1.

—11—

Additional information

11.1 Converting existing documents

11.1.1 Converting from plain LaTeX

A LaTeX document will typically continue to work the same in most respects if `\usepackage{amsmath}` is added in the document preamble. By default, however, the `amsmath` package suppresses page breaks inside multiple-line displayed equation structures such as `eqnarray`, `align`, and `gather`. To continue allowing page breaks inside `eqnarray` after switching to `amsmath`, you will need to add the following line in your document preamble:

```
\allowdisplaybreaks[1]
```

To ensure normal spacing around relation symbols, you might also want to change `eqnarray` to `align`, `multline`, or `equation`/`split` as appropriate.

Most of the other differences in `amsmath` usage can be considered optional refinements, e.g., using

```
\DeclareMathOperator{\Hom}{Hom}
```

instead of `\newcommand{\Hom}{\mbox{Hom}}`.

11.1.2 Converting from $\mathcal{AMS}$-LaTeX 1.1

See `diffs-m.txt`.

11.2 Technical notes

The file `technote.tex` contains some remarks on miscellaneous technical questions that are less likely to be of general interest.

11.3 Getting help

Questions or comments regarding `amsmath` and related packages should be sent to:

American Mathematical Society
Technical Support
Electronic Products and Services
P. O. Box 6248
Providence, RI 02940

Phone: 800-321-4AMS (321-4267) or 401-455-4080
Internet: `tech-support@ams.org`

If you are reporting a problem you should include the following information to make proper investigation possible:

1. The source file where the problem occurred, preferably reduced to minimum size by removing any material that can be removed without affecting the observed problem.
2. A LaTeX log file showing the error message (if applicable) and the version numbers of the document class and option files being used.

11.4 Of possible interest

Information about obtaining AMSFonts or other TeX-related software from the AMS Internet archive `e-math.ams.org` can be obtained by sending a request through electronic mail to: `webmaster@ams.org`.

Information about obtaining the `amsmath` distribution on diskette from the AMS is available from:

American Mathematical Society
Customer Services
P. O. Box 6248
Providence, RI 02940

Phone: 800-321-4AMS (321-4267) or 401-455-4000
Internet: `cust-serv@ams.org`

The TeX Users Group is a nonprofit organization that publishes a journal (*TUGboat*), holds meetings, and serves as a clearing-house of general information about TeX and TeX-related software.

TeX Users Group
PO Box 2311
Portland, OR 97208-2311
USA

Phone: +1-503-223-9994
Email: `office@tug.org`

Membership in the TeX Users Group is a good way to support continued development of free TeX-related software. There are also many local TeX user groups in other countries; information about contacting a local user group can be gotten from the TeX Users Group.

There is a Usenet newsgroup called `comp.text.tex` that is a fairly good source of information about LaTeX and TeX in general. If you don't know about reading newsgroups, check with your local system administrator to see if newsgroup service is available at your site.

Bibliography

[1] *AMSFonts version* 2.2—*user's guide*, Amer. Math. Soc., Providence, RI, 1994; distributed with the AMSFonts package.

[2] *Instructions for preparation of papers and monographs—$\mathcal{A}\kern-.1667em\lower.5ex\hbox{$\mathcal{M}$}\kern-.125em\mathcal{S}$-LaTeX*, Amer. Math. Soc., Providence, RI, 1996, 1999.

[3] *Using the* `amsthm` *Package*, Amer. Math. Soc., Providence, RI, 1999.

[4] Michel Goossens, Frank Mittelbach, and Alexander Samarin, *The LaTeX companion*, Addison-Wesley, Reading, MA, 1994. [*Note: The 1994 edition is not a reliable guide for the* `amsmath` *package unless you refer to the errata for Chapter 8—file* `compan.err`*, distributed with LaTeX.*]

[5] G. Grätzer, *Math into LaTeX: An Introduction to LaTeX and AMS-LaTeX* `http://www.ams.org/cgi-bin/bookstore/bookpromo?fn=91&arg1=bookvideo&itmc=MLTEX`, Birkhäuser, Boston, 1995.

[6] Donald E. Knuth, *The TeXbook*, Addison-Wesley, Reading, MA, 1984.

[7] Leslie Lamport, *LaTeX: A document preparation system*, 2nd revised ed., Addison-Wesley, Reading, MA, 1994.

[8] Frank Mittelbach and Rainer Schöpf, *The new font family selection—user interface to standard LaTeX*, *TUGboat* **11**, no. 2 (June 1990), pp. 297–305.

[9] Michael Spivak, *The joy of TeX*, 2nd revised ed., Amer. Math. Soc., Providence, RI, 1990.

Index