$\mathcal{AMS}$-LaTeX Version 1.2
User's Guide

American Mathematical Society

01 November 1996
Version 1.02

Contents

What is '$\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX', and why would anyone want to use it?

What is '$\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX'?

The name $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX is used for convenience to describe a set of loosely related files that are distributed together by the American Mathematical Society. Basically they may be described as miscellaneous enhancements to LaTeX for *superior information structure of mathematical documents* and *superior printed output*. Because $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX is an extension for LaTeX, which in turn is a 'macro package' for the TeX typesetting program, it follows that in order to use any of the pieces of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX you need to have TeX and LaTeX installed first.

LaTeX by itself does a rather good job of typesetting mathematics, compared to non-TeX-based software; it doesn't add much, however, to the basic set of mathematical capabilities that it adopted from the Plain TeX macro package.

At the same time that LaTeX was being developed by Leslie Lamport (roughly 1982–1986), the American Mathematical Society was throwing its resources into the development of a different macro package known as $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TeX, written by Michael Spivak. By 1987 or so it became evident that $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TeX and LaTeX had complementary feature sets: $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TeX focused on the typesetting of math formulas and on fine-tuning typically done by publishers, and was relatively weak in other areas (for example no automatic numbering or cross-reference facilities); LaTeX focused on document structure and logical markup of text, and had a comparatively limited set of features for dealing with math formula contents. This situation led to dissatisfaction among both $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TeX and LaTeX users who saw desirable features tantalizingly out of reach in the other macro package. So the American Mathematical Society looked into the question of producing some sort of combination of the two macro packages that would better serve mathematicians in their writing tasks. The decision that was eventually taken was to graft the mathematical capabilities of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TeX onto the base stock of LaTeX through an extension package: $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX. Most of the programming work was done by Frank Mittelbach and Rainer Schöpf in 1989–1990 and version 1.0 of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX was released in mid-1990.

Why would a LaTeX user want to bother with $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX?

If you are just starting out as a LaTeX user, you'll probably have to take our word for this (or the word of friends and colleagues), but:

If your writing contains a significant proportion of mathematics, and you care about the quality of the printed results, then sooner or later you'll find shortcomings in standard LaTeX and want to remedy them. Chances are that at least the first few of the shortcomings you encounter will be ones that are already addressed by an $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX package. If you want to have maximum mathematical typesetting power ready at hand, rather than stop to cast about for a solution whenever you run into some unusual demand in your writing, then $\mathcal{AMS}$-LaTeX will go a long way toward meeting your needs.

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 $\mathcal{AMS}$-LaTeX 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.
- A `\boldsymbol` command for printing bold versions of individual symbols, including things like ∞ and lowercase Greek letters.
- An `amsthm` package that provides a useful `proof` environment and some enhancements to the `\newtheorem` command: support for multiple theorem styles in a single document and for unnumbered theorem types.

—1—

How to use $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX

1.1 Using an AMS package in a LaTeX document

A 'package' in LaTeX terminology is an extension written in such a form that it can be used via the `\usepackage` command. Many of the principal features of $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX are provided in separate packages so that they can be used individually on demand. The `amsmath` package is perhaps the single most noteworthy package, as it subsumes the `amstext`, `amsbsy`, and `amsopn` packages, and provides a number of other enhancements for mathematical typesetting. The current list of packages is:

`amsmath` Defines extra environments for multiline displayed equations, as well as a number of other enhancements for math.

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

`amsbsy` Defines `\boldsymbol` and `\pmb` 'poor man's bold' commands.

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

`amsthm` Provides a `proof` environment and extensions for the `\newtheorem` command.

`amsintx` Provides more descriptive command syntax for integrals and sums.

`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.

`upref` Makes `\ref` print cross-reference numbers always in an upright/roman font regardless of context.

1.2 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.

For symmetry there should perhaps be a `centereqn` option as well, to balance with `fleqn`, but as things currently stand there doesn't seem to be a genuine need for it.

—2—

Displayed equations (`amsmath` package)

2.1 Introduction

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

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

(Although the standard `eqnarray` environment remains available, `align` or `split` are recommended instead.)

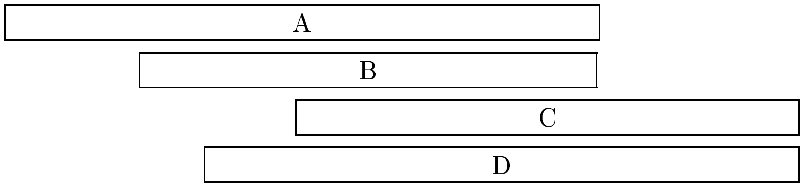
Except for `split`, each environment has both starred and unstarred forms, where the unstarred forms have automatic numbering using LaTeX's `equation` counter. You can suppress the number on any particular line by putting `\notag` before the `\\`; you can also override it with a tag of your own using `\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 AMS-LaTeX sample files `testmath.tex` and `subeqn.tex`.

2.2 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]

2.3 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`. Intermediate lines will be centered independently within the display width. However, it's possible to force a line 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

(2.10)



```
\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 using LaTeX's `\setlength` and `\addtolength` commands. If the `multline` contains more than two lines, any

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

Table 2.1. Comparison of displayed equation environments (vertical lines indicating nominal margins)

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

$$a = b$$

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

$$(1) \qquad a = b$$

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

$$(2) \qquad \begin{aligned} a &= b+c-d \\ &\quad +e-f \\ &= g+h \\ &= i \end{aligned}$$

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

$$(3) \qquad \begin{aligned} &a+b+c+d+e+f \\ &\qquad\qquad +i+j+k+l+m+n \end{aligned}$$

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

$$\begin{gathered} (4) \qquad a_1 = b_1 + c_1 \\ (5) \qquad a_2 = b_2 + c_2 - d_2 + e_2 \end{gathered}$$

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

$$\begin{aligned} (6) \qquad a_1 &= b_1 + c_1 \\ (7) \qquad a_2 &= b_2 + c_2 - d_2 + e_2 \end{aligned}$$

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

$$\begin{aligned} (8) \qquad a_{11} &= b_{11} & a_{12} &= b_{12} \\ (9) \qquad a_{21} &= b_{21} & a_{22} &= b_{22} + c_{22} \end{aligned}$$

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

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

lines other than the first and last will be centered individually between the margins (except when the `fleqn` option is in effect).

2.4 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, as usual. In addition, 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:

$$\begin{split}
H_c&=\frac{1}{2n} \sum^n_{l=0}(-1)^{l}(n-{l})^{p-2}
\sum_{l _1+\dots+ l _p=l}\prod^p_{i=1} \binom{n_i}{l _i}\\
&\quad\cdot[(n-l)-(n_i-l _i)]^{n_i-l _i}\cdot
\Bigl[(n-l)^2-\sum^p_{j=1}(n_i-l _i)^2\Bigr].
\end{split} \tag{2.11}$$

```
\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}
```

2.5 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 2.1).

2.6 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 2.1).

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

$$x=y \qquad X=Y \qquad a=b+c \tag{2.12}$$

$$x'=y' \qquad X'=Y' \qquad a'=b \tag{2.13}$$

$$x+x'=y+y' \qquad X+X'=Y+Y' \qquad a'b=c'b \tag{2.14}$$

```
\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:

$$
\begin{align}
x &= y_1 - y_2 + y_3 - y_5 + y_8 - \dots && \text{by (2.21)} \tag{2.15}\\
&= y' \circ y^* && \text{by (3.1)} \tag{2.16}\\
&= y(0)y' && \text{by Axiom 1.} \tag{2.17}
\end{align}
$$

```
\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 space between equation columns to be explicitly specified. Here the number of equation columns must also be specified (where the number of 'columns' is calculated as $(1 + \&_{\mathrm{max}})/2$ with $\&_{\mathrm{max}}$ = maximum number of `&` markers on any line).

$$
\begin{alignat}{2}
x &= y_1 - y_2 + y_3 - y_5 + y_8 - \dots &\quad& \text{by (2.21)} \tag{2.18}\\
&= y' \circ y^* && \text{by (3.1)} \tag{2.19}\\
&= y(0)y' && \text{by Axiom 1.} \tag{2.20}
\end{alignat}
$$

```
\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}
```

2.7 Alignment building blocks

Some other equation alignment environments, such as `aligned` and `gathered`, construct self-contained units that can be used inside of other expressions, or set side-by-side. These environments take an optional argument to specify their vertical positioning with respect to the material on either side. The default is 'middle' placement with the vertical midpoint of the total unit falling on the math axis[2]. For example:

$$
\begin{aligned}
\alpha &= \alpha\alpha\\
\beta &= \beta\beta\beta\beta\beta\\
\gamma &= \gamma
\end{aligned}
\qquad\text{versus}\qquad
\begin{aligned}
\delta &= \delta\delta\\
\eta &= \eta\eta\eta\eta\eta\eta\\
\varphi &= \varphi
\end{aligned}
$$

[2]The height of the cross-bar in the + symbol.

```
\begin{equation*}
\begin{aligned}
\alpha&=\alpha\alpha\\
\beta&=\beta\beta\beta\beta\beta\\
\gamma&=\gamma
\end{aligned}
\qquad\text{versus}\qquad
\begin{aligned}[t]
\delta&=\delta\delta\\
\eta&=\eta\eta\eta\eta\eta\eta\\
\varphi&=\varphi
\end{aligned}
\end{equation*}
```

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

$$P_{r-j}=\begin{cases} 0 & \text{if $r-j$ is odd,}\\ r!\,(-1)^{(r-j)/2} & \text{if $r-j$ is even.} \end{cases} \tag{2.21}$$

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 and the embedded math.

2.8 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. Even so, difficulties in accurately calculating the profile of an equation can occasionally result in a number placement that doesn't look right. So there is a \raisetag command provided to adjust the vertical position of the current equation number. 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.)

2.9 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. Unlike eqnarray, the amsmath environments don't allow page breaks between lines, unless \displaybreak or \allowdisplaybreaks is used. The philosophy is that page breaks in such situations should receive individual attention from the author. `\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; `\displaybreak` 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` 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.

2.10 Textual interjections within a display

The command `\intertext` is used for a short interjection of one or two lines of text in the middle of a display alignment. 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.

$$A_1 = N_0(\lambda;\Omega') - \phi(\lambda;\Omega'), \tag{2.22}$$

$$A_2 = \phi(\lambda;\Omega') - \phi(\lambda;\Omega), \tag{2.23}$$

and

$$A_3 = \mathcal{N}(\lambda;\omega). \tag{2.24}$$

```
\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}
```

2.11 Equation numbering

2.11.1 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 [5, §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, the command would be

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

2.11.2 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, and adds an italic correction if necessary. To refer to an equation that was labeled with the label `e:baset`, the usage would be `\eqref{e:baset}`.

2.11.3 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 [5, §6.3, §C.8.4]. For example, redefining `\theequation` as follows will produce roman numerals.

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

—3—

Miscellaneous mathematics features (`amsmath` package)

3.1 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.[1] (If you need left or right alignment in a column or other special formats you must resort to `array`.)

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`,`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).

$$\begin{pmatrix} D_1t&-a_{12}t_2&\dots&-a_{1n}t_n\\
-a_{21}t_1&D_2t&\dots&-a_{2n}t_n\\
\hdotsfor[2]{4}\\
-a_{n1}t_1&-a_{n2}t_2&\dots&D_nt\end{pmatrix}, \tag{3.1}$$

```
\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}
```

3.2 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
`\,`	`\thinspace`	$\lrcorner\,\llcorner$	`\!`	`\negthinspace`	$\lrcorner\!\llcorner$
`\:`	`\medspace`	$\lrcorner\:\llcorner$		`\negmedspace`	$\lrcorner\negmedspace\llcorner$
`\;`	`\thickspace`	$\lrcorner\;\llcorner$		`\negthickspace`	$\lrcorner\negthickspace\llcorner$
	`\quad`	$\lrcorner\quad\llcorner$			
	`\qquad`	$\lrcorner\qquad\llcorner$			

[1]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.

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}`.

3.3 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 fill out the set:

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

3.4 Dots

When the `amsmath` package is used, ellipsis dots should normally be typed as `\dots`. Placement (on the baseline or centered) is determined by whatever follows the `\dots`. If the next thing is a plus sign or other binary symbol, the dots will be centered; if it's any other kind of symbol, they will be on the baseline.

If the dots fall at the end of a math formula, the next thing is something like `\end` or `\)` or `$`, which does not give any information about how to place the dots. Then you must help by using `\dotsc` for "dots with commas," or `\dotsb` for "dots with binary operators/relations," or `\dotsm` for "multiplication dots," or `\dotsi` for "dots with integrals." For example, the input

```
Then we have the series $A_1,A_2,\dotsc$,
the regional sum $A_1+A_2+\dotsb$,
the orthogonal product $A_1A_2\dotsm$,
and the infinite integral
\[\int_{A_1}\int_{A_2}\dotsi\].
```

will produce low dots in the first instance and centered dots in the others, with the spacing on either side of the dots nicely adjusted:

> Then we have the series $A_1, A_2, \dots$, the regional sum $A_1 + A_2 + \cdots$, the orthogonal product $A_1 A_2 \cdots$, and the infinite integral
>
> $$\int_{A_1}\int_{A_2}\cdots.$$

Specifying dots this way, in terms of their meaning rather than in terms of their visual placement, is in keeping with the general philosophy of LaTeX and makes documents more easily adaptable to different conventions.

3.5 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.)

3.6 Accents in math

The following accent commands automatically give good positioning of double accents:

```
\Hat    \Check  \Tilde  \Acute  \Grave  \Dot  \Ddot
\Breve  \Bar    \Vec
```

With the usual non-capitalized math accent commands, the second accent will sometimes be askew; for example: $\hat{\hat{A}}$ (`\hat{\hat{A}}`). With the `amsmath` package, if you type `\Hat{\Hat{A}}` (using the capitalized form for both accents) the second accent will be better positioned: $\Hat{\Hat{A}}$.

This double accent operation is complicated and tends to slow down the processing of a document. If your document contains many double accents, you may wish to use the `amsxtra` package, which provides an `\accentedsymbol` command. `\accentedsymbol` is a sort of hybrid of `\newcommand` and `\savebox`; you use it in the preamble of your document to store the result of the double accent command in a 'box' for quick retrieval.

```
\accentedsymbol{\Ahathat}{\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.

3.7 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.

3.8 Boxed formulas

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

$$\boxed{\eta \leq C(\delta(\eta) + \Lambda_M(0,\delta))} \tag{3.2}$$

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

3.9 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):

$$A \xleftarrow{n+\mu-1} B \xrightarrow[T]{n\pm i-1} C \tag{3.3}$$

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

3.10 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 §7.3.

3.11 Fractions and related constructions

3.11.1 Disallowing primitive TeX fraction commands

The six generalized fraction commands `\over`, `\overwithdelims`, `\atop`, `\atopwithdelims`, `\above`, `\abovewithdelims` are expressly forbidden by the `amsmath` package, as their syntax is decidedly out of place in LaTeX; use of the forms `\frac`, `\binom`, `\genfrac`, and variants is required.[2]

3.11.2 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

[2]Not only is the unusual syntax of the primitive TeX fraction commands rather out of place in LaTeX, but furthermore that syntax seems to be solely responsible for one of the most significant flaws in TeX's mathematical typesetting capabilities: the fact that the current mathstyle at any given point in a math formula cannot be determined until the end of the formula, because of the possibility that a following generalized fraction command will change the mathstyle of the *preceding* material. As the side effects are a bit technical in nature, they are discussed in `technote.tex` rather than here.

form. The `amsmath` package provides also `\dfrac` and `\tfrac` as convenient abbreviations for `{\displaystyle\frac ... }` and `{\textstyle\frac ... }`.

$$\frac{1}{k}\log_2 c(f)\;\tfrac{1}{k}\log_2 c(f)\;\sqrt{\frac{1}{k}\log_2 c(f)}\;\sqrt{\dfrac{1}{k}\log_2 c(f)} \tag{3.4}$$

```
\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}
```

3.11.3 The `\binom`, `\dbinom`, and `\tbinom` commands

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

$$2^k-\binom{k}{1}2^{k-1}+\binom{k}{2}2^{k-2} \tag{3.5}$$

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

3.11.4 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`.

3.12 Continued fractions

The continued fraction

$$\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cdots}}} \tag{3.6}$$

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`.

3.13 Smash options

The command `\smash` is used to typeset a subformula and give it 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 `$\sqrt{x} + \sqrt{\smash[b]{y}} + \sqrt{z}$`.

3.14 Delimiters

3.14.1 Delimiter sizes

A subject that escapes mention in the LaTeX book is how to control the size of large delimiters if the automatic sizing done by `\left` and `\right` produces unsatisfactory results. The automatic sizing 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	$(b)(\frac{c}{d})$	$\left(b\right)\left(\frac{c}{d}\right)$	$\bigl(b\bigr)\bigl(\frac{c}{d}\bigr)$	$\Bigl(b\Bigr)\Bigl(\frac{c}{d}\Bigr)$	$\biggl(b\biggr)\biggl(\frac{c}{d}\biggr)$	$\Biggl(b\Biggr)\Biggl(\frac{c}{d}\Biggr)$

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:

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

```
\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.

3.14.2 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, at least not without a significant cost in processing speed, and even then not without falling somewhat short of human readers' abilities. 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, as in the notations for absolute value or norm, 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:

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

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

—4—

Operator names (`amsopn`, `amsmath` packages)

4.1 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 `amsopn` package provides a general mechanism for defining new 'operator names'. As the `amsopn` package is loaded internally by the `amsmath` package, the following features are available there also. 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 $A\mathrm{xxx}B$. 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}
```

A few special operator names are predefined by the `amsopn` package: `\varinjlim`, `\varprojlim`, `\varliminf`, and `\varlimsup`:

`\varlimsup`	$\varlimsup_{n\to\infty} \mathcal{Q}(u_n, u_n - u) \le 0$
`\varliminf`	$\varliminf_{n\to\infty} \lvert a_{n+1}\rvert \,/\, \lvert a_n\rvert = 0$
`\varinjlim`	$\varinjlim (m_i^\lambda)^* \le 0$
`\varprojlim`	$\varprojlim_{p\in S(A)} A_p \le 0$

4.2 `\mod` and its relatives

Commands `\mod`, `\bmod`, `\pmod`, `\pod` are provided by the `amsopn` package to deal with the special spacing conventions of "mod" notation. `\bmod` and `\pmod` are available in LaTeX, but with the `amsopn` 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.

$$\gcd(n,m\bmod n);\quad x\equiv y\pmod b;\quad x\equiv y\mod c;\quad x\equiv y\pod d \tag{4.1}$$

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

—5—

The `\text` command (`amstext`, `amsmath` packages)

The `\text` command is defined by the `amsmath` package through a subordinate package `amstext` (which can also be used independently if desired). 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.

$$f_{[x_{i-1},x_i]} \text{ is monotonic,}\quad i = 1,\dots,c+1 \tag{5.1}$$

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

—6—

The `\boldsymbol` command (`amsbsy`, `amsmath` packages)

The `\boldsymbol` and `\pmb` commands are defined by the `amsbsy` package (also loaded by `amsmath`). The `\boldsymbol` command is used to obtain bold numbers and other nonalphabetic symbols, as well as bold Greek letters, which cannot be made bold via the `\mathbf` command.[1] It can also be used to obtain bold math italic letters; compare the results of `M`, `\mathbf{M}` and `\boldsymbol{M}`: $M\mathbf{M}\boldsymbol{M}$.

[1] Actually, depending on which font set you use, `\mathbf` may—inconsistently—work for cap Greek letters but not for lowercase.

The availability of bold symbols varies on different systems depending on whether or not suitable fonts are installed. The `\boldsymbol` command should usually work fine for the common math symbols at 10pt size or larger, but if you find that it is not having the desired effect for a particular symbol, you could either (a) verify that the necessary fonts are available and properly installed; or (b) use `\pmb`: "poor man's bold", which works by printing multiple copies of the same symbol with slight offsets.

$$A_\infty + \pi A_0 \sim \mathbf{A}_{\boldsymbol{\infty}} \boldsymbol{+} \boldsymbol{\pi} \mathbf{A}_{\boldsymbol{0}} \sim \boldsymbol{A}_{\boldsymbol{\infty}} \boldsymbol{+} \boldsymbol{\pi} \boldsymbol{A}_{\boldsymbol{0}} \tag{6.1}$$

```
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}}
```

—**7**—

Integrals and sums (amsmath, amsintx packages)

7.1 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.

7.2 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)
```

produces a two-line subscript underneath the sum:

$$\sum_{\substack{0\le i\le m\\ 0<j<n}} P(i,j) \tag{7.1}$$

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:

$$\sum_{\begin{subarray}{l} i\in\Lambda\\ 0<j<n \end{subarray}} P(i,j) \tag{7.2}$$

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

7.3 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 $\prod$. 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:

$$\sum\nolimits' E_n \tag{7.3}$$

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
```

to get

$$\sideset{}{'}\sum_{n<k,\;\text{n odd}} nE_n \tag{7.4}$$

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
```

producing

$$\sideset{_*^*}{_*^*}\prod \tag{7.5}$$

7.4 The `amsintx` package

The `amsintx` package is an experimental package that provides variants of the `\int` and `\sum` commands to better mark the boundaries of the quantity being summed or integrated. Some commands for differential notation are also provided. If you are interested in this possibility, run LaTeX on the documentation file `amsintx.dtx` to get the most up-to-date information on usage.

—8—

Commutative diagrams (`amscd` package)

Some commutative diagram commands like the ones in AMS-TeX are available as a separate package, `amscd`. For complex commutative diagrams authors will need to turn to more comprehensive packages like XY-pic (see §C.5), 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".

—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* [3]. Many users of AMS-LaTeX also obtain an auxiliary collection of math fonts known as 'AMSFonts'. The basic set of math font commands in LaTeX includes `\mathbf`, `\mathrm`, `\mathcal`, `\mathsf`, `\mathtt`, `\mathit`. Additional math alphabet commands are available through the packages `amsfonts` and `eucal`, if the requisite fonts are installed on your system (see §C.2).

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{\vec}[1]{\mathbf{#1}}
```

whereupon[1] you can write `\vec{a} + \vec{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 `\vec`; 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}}
```

—10—

Theorems and related structures (`amsthm` package)

10.1 Introduction

The `amsthm` package provides an enhanced version of the LaTeX command `\newtheorem` for defining theorem-like environments. The `amsthm` version of the `\newtheorem` command recognizes a `\theoremstyle` specification (as in Mittelbach's `theorem` package) and has a `*` form for defining unnumbered environments. The `amsthm` package also defines a `proof` environment that automatically adds a Q.E.D. symbol at the end. AMS document classes automatically load the `amsthm` package, so everything described here applies to them as well. An example file `thmtest.tex` is provided in the AMS-LaTeX distribution.

[1]If you actually tried this example you would discover that the command `\vec` is already defined. It produces a different sort of notation for vectors: a small over-arrow $\vec{x}$. The solution is to use `\renewcommand` (if you expect that you will never need the over-arrow version of the notation) or to choose a different name for your new vector command.

10.2 The `\newtheorem` command

In mathematical research articles and books, theorems and proofs are among the most common elements, but authors also use many others that fall in the same general class: lemmas, propositions, axioms, corollaries, conjectures, definitions, remarks,, cases, steps, and so forth. As these elements form a slice of the text stream with well-defined boundaries, they are naturally handled in LaTeX as environments. But LaTeX document classes normally do not provide predefined environments for theorem-like elements because (a) that would make it difficult for authors to exercise the necessary control over the automatic numbering, and (b) the variety of such elements is so wide that it's just not possible for a document class to provide every one that will ever be needed. Instead there is a command `\newtheorem`, similar to `\newenvironment` in effect, that makes it easy for authors to set up the elements required for a particular document.

The `\newtheorem` command has two mandatory arguments; the first one is the environment name that the author would like to use for this element; the second one is the heading text. For example,

```
\newtheorem{lem}{Lemma}
```

means that instances in the document of

```
\begin{lem} Text text ... \end{lem}
```

will produce

> **Lemma 1.** *Text text ...*

where the heading consists of the specified text 'Lemma' and an automatically generated number and punctuation.

If `\newtheorem*` is used instead of `\newtheorem` in the above example, there will not be any automatic numbers generated for any of the lemmas in the document. This form of the command can be useful if you have only one lemma and don't want it to be numbered; more often, though, it is used to produce a special named variant of one of the common theorem types. For example, if you have a lemma whose name should be 'Klein's Lemma' instead of 'Lemma' + number, then the statement

```
\newtheorem*{KL}{Klein's Lemma}
```

would allow you to write

```
\begin{KL} Text text ... \end{KL}
```

and get the desired output.

10.3 Numbering modifications

In addition to the two mandatory arguments, `\newtheorem` has two mutually exclusive optional arguments. These affect the sequencing and hierarchy of the numbering.

By default each kind of theorem-like environment is numbered independently. Thus if you have three lemmas and two theorems interspersed, they will be numbered something like this: Lemma 1, Lemma 2, Theorem 1, Lemma 3, Theorem 2. If you want lemmas and theorems to share the same numbering sequence—Lemma 1, Lemma 2, Theorem 3, Lemma 4, Theorem 5—then you should indicate the desired relationship as follows:

```
\newtheorem{thm}{Theorem}
\newtheorem{lem}[thm]{Lemma}
```

The optional argument `[thm]` in the second statement means that the `lem` environment should share the `thm` numbering sequence instead of having its own independent sequence.

To have a theorem-like environment numbered subordinately within a sectional unit—e.g., to get propositions numbered Proposition 2.1, Proposition 2.2, and so on in Section 2—put the name of the parent unit in square brackets in final position:

```
\newtheorem{prop}{Proposition}[section]
```

With the optional argument `[section]`, the `prop` counter will be reset to 0 whenever the parent counter `section` is incremented.

10.4 Changing styles for theorem-like environments

10.4.1 The `\theoremstyle` command

The `amsthm` package supports the notion of a current theorem style, which determines what will be produced by a given `\newtheorem` command. The three theorem styles provided—`plain`, `definition`, and `remark`—receive different typographical treatment that gives them visual emphasis corresponding to their relative importance. The details of this typographical treatment may vary depending on the document class, but typically the `plain` style produces italic body text, while the other two styles produce roman body text.

To create new theorem-like environments in the different styles, divide your `\newtheorem` commands into groups and preface each group with the appropriate `\theoremstyle`. If no `\theoremstyle` command is given, the style used will be `plain`. Some examples:

```
\theoremstyle{plain}% default
\newtheorem{thm}{Theorem}[section]
\newtheorem{lem}[thm]{Lemma}
\newtheorem{prop}[thm]{Proposition}
\newtheorem*{cor}{Corollary}
\newtheorem*{KL}{Klein's Lemma}

\theoremstyle{definition}
\newtheorem{defn}{Definition}[section]
\newtheorem{conj}{Conjecture}[section]
\newtheorem{exmp}{Example}[section]
```

```
\theoremstyle{remark}
\newtheorem*{rem}{Remark}
\newtheorem*{note}{Note}
\newtheorem{case}{Case}
```

10.4.2 Number swapping

A not uncommon style variation for theorem heads is to have the theorem number on the left, at the beginning of the heading, instead of on the right. As this variation is usually applied across the board regardless of individual `\theoremstyle` changes, number-swapping is done by placing a `\swapnumbers` command at the beginning of the list of `\newtheorem` statements that should be affected. For example:

```
\swapnumbers
\theoremstyle{plain}
\newtheorem{thm}{Theorem}
\theoremstyle{remark}
\newtheorem{rem}{Remark}
```

After the above declarations, theorem and remark heads will be printed in the form **1.4 Theorem.**, *9.1. Remark.*

10.4.3 Further customization possibilities

More extensive customization capabilities are provided by the `amsthm` package in the form of a `\newtheoremstyle` command and a mechanism for using package options to load custom theoremstyle definitions. As these capabilities are somewhat beyond the needs of the average user, discussion of the details is consigned to the example file `thmtest.tex` and to the commentary in `amsthm.dtx`.

10.5 Proofs

A predefined `proof` environment provided by the `amsthm` package produces the heading "Proof" with appropriate spacing and punctuation. The proof environment is primarily intended for short proofs, no more than a page or two in length; longer proofs are usually better done as a separate `\section` or `\subsection` in your document.

A 'Q.E.D.' symbol, □, is automatically appended at the end of a `proof` environment. To substitute a different end-of-proof symbol, use `\renewcommand` to redefine the command `\qedsymbol`. For a long proof done as a subsection or section instead of with the `proof` environment, you can obtain the symbol and the usual amount of preceding space by using `\qed`.

Placement of the Q.E.D. symbol can be problematic if the last part of a `proof` environment is a displayed equation or list environment or something of that nature. Adequate results can sometimes be obtained by using `\qed` at the appropriate spot and then undefining `\qed` just before the end of the proof. (The effect will be automatically localized to the current proof by normal LaTeX scoping rules.) For example:

```
\begin{proof}
...
\begin{equation}
G(t)=L\gamma!\,t^{-\gamma}+t^{-\delta}\eta(t) \qed
\end{equation}
\renewcommand{\qed}{}\end{proof}
```

An optional argument of the proof environment allows you to substitute a different name for the standard “Proof”. If you want the proof heading to be, say, “Proof of the Main Theorem”, then write

```
\begin{proof}[Proof of the Main Theorem]
```

—Appendix A—

Installation instructions

A.1 Introduction

To use version 1.2 of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX it is necessary for you to have a recent version of LaTeX (June 1994 or later, 'LaTeX 2_ε'). If you're not sure about the version, look at the startup message that is printed on screen and in the TeX log when you run LaTeX. It should mention the LaTeX version number and date somewhere in the first ten lines. If your version of LaTeX is older than June 1994, we suggest getting the latest version from the Comprehensive TeX Archive Network (CTAN), directory `tex-archive/macros/latex`, ftp addresses `ftp.cdrom.com` (US), `ftp.dante.de` (Germany), or `ftp.tex.ac.uk` (UK). If ftp file transfer is not an option for you, contact the source from which you originally obtained LaTeX.

A.2 Putting files in a suitable place on your system

See the `READ.ME` file for possible updates about installation procedures.

There are two 'areas' (directories or folders) on your system that are involved in installing $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX: an $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX source files area, and a LaTeX input files area. All files in the `inputs` subdirectory of the $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX distribution should be placed in the LaTeX input directory or folder on your system. Consult your TeX documentation if you don't know where this is. (You could also try looking for the file `article.cls`; the place where you find it is almost surely your LaTeX input files area.)

All other files in the $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX distribution (the ones in the `math` and `classes` subdirectories) can be placed in an $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX source files area; if you are installing $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX for the first time, create a new folder or directory for this purpose.

A.3 Testing

For a quick test of the installation, try printing the test file `subeqn.tex`. For more extensive tests print the $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX user's guide (`amsldoc.tex`) or `test-math.tex`.

A.4 Extra math fonts

For information on the AMSFonts collection, a set of extra math fonts that supplements the standard set of LaTeX math fonts, see §C.2.

A.5 Memory requirements

TeX divides up the memory available to it into various categories. On most systems the sizes of these categories are fixed at the beginning of a TeX run and cannot dynamically grow to meet unexpected demands. (In fact certain implementations of TeX have the sizes fixed at the time a format file is created, or even when the TeX program is compiled.) Use of extra packages places burdens on certain memory categories (string pool, hash size, main memory) in proportion to the total size of the packages. Table A.1 lists the recommended capacities in various categories for successful use of the $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX major documentstyles or the `amsmath` package. Not all categories are listed; the ones that appear are the ones where problems tend to occur nowadays.

Note in particular that the base value for string pool needs to be much larger than the values typically found at the end of a LaTeX log. This is because the string pool capacity reported by TeX in response to a `\tracingstats` command is not the base value, but the result of subtracting from the base value the number of characters in TeX's built-in error messages, the names of primitive control sequences, and the names of all additional control sequences defined in the format file (in our case, the whole of LaTeX), not to mention font names and file names. Thus the reported value only measures the amount of string capacity that remains to the user after the format file is loaded. The reported value for number of strings is reduced in the same way.

Table A.1. Recommended values for selected TeX memory categories

	Capacity		
Category	Adequate	Generous	WEB variable
strings	5 000	30 000	`max_strings`
string characters	80 000	300 000	`pool_size`
macro string pool*	50 000	270 000	`string_vacancies`
main memory	80 000	250 000	`main_mem`
control sequences	5 000	20 000	`hash_size`
font information	60 000	300 000	`font_mem_size`
number of fonts	128	256	`font_max`
input buffer	1 000	5 000	`buf_size`
save stack	2 000	10 000	`save_size`

*The number of string characters left for macro packages and user commands, after all primitives and built-in error messages have been loaded—i.e., the total number of string characters available for a format file and individual documents using that format file.

A.6 Files included in this distribution

As files are occasionally added or removed from the distribution, you should check the `READ.ME` file if you want the most up-to-date possible list.

File name	**Description**
`amsldoc.tex`	this user's guide
`amslatex.faq`	frequently asked questions

`amslatex.bug`	description of bug fixes and other changes
`diff12.tex`	description of differences between versions 1.1 and 1.2
`technote.tex`	some technical notes
`amslatex.ins`	installation file
`testmath.tex`	test file for general math features
`subeqn.tex`	test file for 'subequations' environment
`amsbsy.dtx`	for `\boldsymbol` and `\pmb`
`amscd.dtx`	for commutative diagrams
`amsgen.dtx`	auxiliary file
`amsintx.dtx`	alternative command syntax for integrals and sums
`amsmath.dtx`	equations and other math
`amsopn.dtx`	operator names
`amstext.dtx`	`\text` command
`amsxtra.dtx`	misc rarely used commands
`amstex.sty`	frozen version of old `amstex` package
`amsdtx.dtx`	document class for printing AMS `.dtx` files
`instr-l.tex`	instructions for using AMS document classes
`amsclass.dtx`	source for `amsart`, `amsbook`, and `amsproc` document classes
`amsthm.dtx`	provides `\theoremstyle`, `\newtheorem*`
`upref.dtx`	makes `\ref` always produce roman/upright numbers
`thmtest.tex`	test file for the `amsthm` package
`amsalpha.bst`	Bibliography style for BibTEX
`amsplain.bst`	Bibliography style for BibTEX
`mrabbrev.bib`	BibTEX abbreviations for MR journal names; not to be used directly with BibTEX but as a source for entries to be extracted

—Appendix B—

Error messages and output problems

B.1 General remarks

This is a supplement to Chapter 8 of the LaTEX manual [5] (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.

B.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.

`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.

`Foreign command \over; use \frac or \genfrac instead`
Example:

```
! Package amsmath Error: Foreign command: \over; use \frac or \gen-
frac instead.

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

l.49 Fractions ${1\over
                        2}$ and
?
```

Explanation: The primitive generalized fraction commands of TeX are disallowed by the `amsmath` package because their syntax is foreign to LaTeX. The substitutes `\frac` and `\genfrac` are provided instead. See §3.11.1 for further information.

`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` 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 [5, §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 [5, §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 §3.14.1); 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 \equation 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{equation}` command, e.g.,

```
\begin{equation}
...
\end{equatin}
```

or by using abbreviations such as `\beq` and `\eeq` for `\begin{equation}` and `\end{equation}`:

```
\beq
...
\eeq
```

That kind of abbreviation works in ordinary LaTeX but for technical reasons it does not work with any of the displayed equation environments (`gather`, `align`, `split`, `equation`, etc.) when the `amsmath` package is used. Work to re-enable the use of such abbreviations is under way, but nontrivial technical complications are involved (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', perhap-
s 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.

B.3 Wrong output

B.3.1 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:

```
\numberwith{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

```
\numberwith{equation}{section}
```

B.3.2 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 §B.3.1.

B.3.3 Double accent command failed to position the second accent properly

The capitalization of the command names is important. Check whether all of the accent commands are capitalized. Compare

$$\hat{\hat{A}}\quad\hat{\hat{A}}\quad\hat{\hat{A}}$$

```
\[\hat{\hat{A}}\quad\Hat{\hat{A}}\quad\Hat{\Hat{A}}\]
```

B.3.4 The `\boldsymbol` command didn't work

This probably means that the font necessary to produce a bold version of the symbol in question is not available on your system or not installed in a way that LaTeX can use it. For example, bold versions of the AMS extra symbol fonts `msam` and `msbm` do not currently exist, so `\boldsymbol{\square}` will merely produce the same result as `\square` alone.

Also, the weight of the fonts `cmmib` and `cmbsy` is 'bold' rather than 'bold extended' as for `cmbx`. Depending on the symbol in question and the resolution of your printer (or previewer), a symbol that was taken from one of those two bold symbol fonts might be nearly indistinguishable from the non-bold version.

—Appendix C—

Other useful items for mathematical documents

C.1 AMS documentclasses (`amsart`, `amsbook`, `amsproc`)

The American Mathematical Society provides custom $\mathcal{AMS}$-LaTeX document classes for AMS journals and book series. These may obtained by anonymous FTP from the e-MATH system `e-math.ams.org`, directory, in the `author-info` area. (Note that directory structures may occasionally change; if you have trouble locating files you need, please contact AMS Technical Support; see Appendix E.) Specific and generic classes can be requested by e-mail to `ams-latex@ams.org` or ordered on floppy disk through the AMS Customer Services Department at `cust-serv@ams.org` or 800-321-4267.

C.2 Extra math fonts (the AMSFonts collection)

The 'AMSFonts' font collection is a set of extra math fonts that supplements the standard set of LaTeX math fonts. It includes LaTeX packages called `amsfonts` and `amssymb` that make it easy to use the extra fonts in a LaTeX document. If you don't have the AMSFonts collection already on your system, and would like to use some of the symbols in it, it can be obtained by anonymous FTP from CTAN (see §A.1) or from the e-MATH system at `e-math.ams.org`. It can also be ordered on disk from the AMS Customer Services Department at `cust-serv@ams.org` or 800-321-4267.

C.3 Syntax checking (the `syntonly` package)

If you are working on a relatively slow computer system you might want to try the `syntonly` package. This makes LaTeX skim through your document only checking for proper syntax and usage of the commands, but not producing any printed pages. As LaTeX runs quite a bit faster under those conditions, for early error-checking runs of a document this can save you valuable time. The `syntonly` package is a standard LaTeX package so you should find it already on your system; to use it, write

```
\usepackage{syntonly}
\syntaxonly
```

When you want to produce pages, comment out the second line by adding a percent character:

```
%\syntaxonly
```

C.4 Verbatim and comments (the `verbatim` package)

The `verbatim` package is another standard LaTeX package, like `syntonly`. In addition to some minor enhancements for the `verbatim` environment, it provides a `comment` environment that skips everything between `\begin{comment}` and the next `\end{comment}`. A command `\verbatiminput` is also provided for typesetting a whole file in verbatim style.

C.5 Commutative diagrams and other diagrams (packages `diagram`, `xypic`, `pstricks`)

The `amscd` package for commutative diagrams that is included in the AMS-LaTeX distribution is quite limited. The following packages are worth investigation for authors who need more powerful diagram features: `diagram`, `xypic`, `pstricks`. These are available by anonymous FTP from CTAN (see §A.1). The `diagram` package, by Michael Barr, focuses on commutative diagrams and is thus smaller than the others; it uses LaTeX arrows as produced by `\vector` for diagram construction. The `xypic` package, by Kristoffer Rose, is a more general diagram package; it requires special arrow fonts (included in the `xypic` distribution) or PostScript capabilities in your system. The `pstricks` package, by Timothy van Zandt, also provides general diagram object commands; it is strictly for PostScript use, as the name implies.

—Appendix D—

Where to find other information

D.1 Technical notes

The file `technote.tex` contains some remarks on miscellaneous technical questions related to AMS-LaTeX; they are relegated to a separate document because they are unlikely to be of interest to most users.

D.2 Differences between AMS-LaTeX version 1.1 and AMS-LaTeX version 1.2

Version 1.2 of AMS-LaTeX fills in some gaps and corrects some mistakes in the AMS-LaTeX 1.1 feature set. The significant differences, from a user's perspective, are described in the document `diff12.tex`.

—Appendix E—

Getting help

Questions or comments regarding the AMS-LaTeX package 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.

E.1 Further information

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 $\mathcal{AMS}$-LaTeX 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
1850 Union Street, #1637
San Francisco, CA 94123
Phone: (415) 982-8449
Internet: `tug@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—$\mathcal{AMS}$-LaTeX*, Amer. Math. Soc., Providence, RI, 1996.

[3] Michel Goossens, Frank Mittelbach, and Alexander Samarin, *The LaTeX companion*, Addison-Wesley, Reading, MA, 1994.

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

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

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

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

Index