SUMMARY OF $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX

MICHAEL SPIVAK
September 13, 1983

This is a brief summary of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX, as it is now written for TEX82. The old "Joy of TEX" manual is woefully out of date, but I will not be able even to begin on the new version until November, so this summary has to be used as a supplement. It probably won't be of much help for learning $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX from scratch, but people who are already familiar with the old $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX can use it to see what new things are available, and what changes have been made. (Some of the more esoteric features of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX aren't mentioned here, and will have to wait for the manual.)

BASICS

Control Sequences. Control sequences are of two types. The first, like `\par`, consists of \ followed by any number of *letters*. The second, like `\$`, consists of \ followed by just one *non-letter*. A control sequence of the first kind is ended by any non-letter, including a space, and a space after such a control sequence is *not* considered as a space within the text. When an interword space does have to be indicated after such a control sequence, you can use \␣, where ␣ indicates a typed space; this is a control sequence of the second type. A control sequence of the second kind needs no special ending, since it is only one character long. A space after such a control sequence *does* count as a space in the text. Multiple spaces count as one space, so \$␣␣ gives only one space after the printed $. A special exception is made for the control sequence \␣; here a space after the control sequence *is* ignored, so that \␣␣ still gives one space (this of course complies with the general rule that multiple spaces count as one space). Also, spaces in math mode still don't count, so it doesn't matter about spaces after control sequences of this sort.

Document Formatting. Your file should begin

```
\input amstex
\documentstyle{amsppt}
...
\document
```

with the line

```
\enddocument
```

at the very end of the file. (If you are using a program `amstex`, which consists of `tex` with the file `amstex.tex` preloaded, then you shouldn't have the initial line `\input amstex`.)

The control sequence `\documentstyle` is supposed to allow you to select the particular format, but at the moment the only style available is the `amsppt` (AMS preprint) style. When you use this style you will get the question

```
Do you want output? (y or n, follow answer by return)
```

on your terminal. If you type anything other than `y` or `n` (or `Y` or `N`), followed by (carriage-return), the question will be repeated.

The other things that go at the beginning of the paper, between `\documentstyle` and `\document`, will be covered later.

DEFINING CONTROL SEQUENCES

To define your own control sequence `\cs` you type

```
\define\cs{...}
```

`\define` will check for you whether `\cs` already has a meaning, and if so it will issue an error message. It's actually rather improbable that you will stumble upon one of TEX's control sequences, since most of them have long and unlikely names. Most of the control sequences consisting of \ followed by a single non-letter do have meanings, but `\0`, `\1`, ...,`\9` and `\.` don't, and they are convenient ones to choose, especially for temporary definitions in a complicated math formula (definitions made inside $ signs disappear at the closing $ sign). If you want to redefine one of your own control sequences, you can use `\redefine`. You might even want to `\redefine` some of TEX's control sequences, provided you can be reasonably sure that they are not used in some hidden way somewhere else. For example, we'll see later that `\b` and `\d` are used for certain accents under letters, and you might want to use them for something else, say for `\beta` (β), and `\delta` (δ). So you would say

```
\redefine\b{\beta}
\redefine\d{\delta}
```

(using `\define` would give an error message). If it turns out that you also need the original `\d` (which gives an under-dot accent), you can use `\predefine`:

```
\predefine\underdot{\d}
\redefine\d{\delta}
```

Then you can type '`\underdot x`' to get 'x̣' if you ever need it. The only important thing is that the `\predefine` comes before the `\redefine` for `\d`.

When you are defining control sequences with arguments,

```
\define\cs#1#2{...}
```

you must be careful not to have spaces in the expression `#1#2{`. But you *don't* have to worry about spaces when using `\cs`. TEX will ignore spaces between arguments.

ORDINARY TEXT

Control Sequences for Special Symbols. If your keyboard is missing certain symbols you might be able to make do with

```
\plus     +
\equal    =
\less     <
\more     >
\lq       `
\rq       '
\vert     |
```

(If you need `\vert` for the |, then you will also need `\Vert`, for `\|`.)

More important, the following symbols are special in $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX:

```
\   {   }   $   &   #   %   "   @   ~   ^   _
```

Some of these have printed versions that can appear as symbols in ordinary text, and you get them by typing the obvious things:

```
\{    {
\}    }
\$    $
\&    &
\#    #
\%    %
\@    @
```

(You can also use `\_` to get a printed _, for things like *first_letter*, which computer scientists like to use.)

Paragraphing and special text symbols. New paragraphs are indicated by either a blank line, or the specific control sequence \par. Double quotes " and " are produced simply by typing two single quotes in a row: `` and ''. If you don't have ` or ' you can use \lq and \rq. These work even in pairs—\lq \lq gives ". Hyphens, en-dashes and em-dashes should be distinguished as follows

hyphen (-)	-
en-dash (–)	--
em-dash (—)	---

A hyphen in math mode (between $ signs) becomes a minus sign −.

Italic type is indicated by \it and **boldface** type by \bf, while \sl gives type that is *slanted*, and \rm indicates a return to ordinary (roman) type. You should add the "italic correction" \/ after something in italic or slanted type unless it is followed by a period or a comma:

```
{\it Italic\/} type ... type that is {\sl slanted}, and ...
```

Accents. Here are the common accents:

\`o	ò
\'o	ó
\^o	ô
\"o	ö
\~o	õ
\=o	ō
\.o	ȯ
\v o	ǒ
\u o	ŏ
\H o	ő (long Hungarian umlaut)
\t oo	o͡o (tie-after accent)
\c o	o̧ (cedilla)
\d o	ọ
\b o	o̲

You can also type \`{o} or \v{o}, etc., but you should accent only single letters.

And here are the special letters:

\oe, \OE	œ, Œ
\ae, \AE	æ, Æ
\aa, \AA	å, Å
\o, \O	ø, Ø
\l, \L	ł, Ł
\ss	ß

Also, for accents over 'i' and 'j' one needs the dotless 'ı' and 'ȷ', which you obtain by typing \i and \j.

Miscellaneous considerations for ordinary type.

1) TEX puts more space after the ends of sentences than between words, so it needs to know which periods represent ends of sentences. Periods after upper-case letters are not considered to be ends of sentences, since they are usually initials. But periods after abbreviations like "Mr." cause problems. So you can use \␣ to indicate an ordinary interword space: `Mr.\ Jones`. Actually, in this case it would be better to type `Mr.@Jones`, using AMS-TEX's "tie" @, which indicates an ordinary interword space and also tells TEX to try not to break the line at that point.

On those rare occasions when you need a period after a capital letter, and it really is the end of a sentence, use \period:

```
Supported by the NSF \period And numerous others.
```

3) To get a footnote at some point,[1] you just type

```
... at some point,
\footnote {Here is a footnote}
you just type
```

and sure enough it will appear, at the bottom of the page. The particular style you are using will determine what symbols to use and whether numbering is consecutive or begins anew on each page. In the amsppt style, superscript numbers are used, and they begin anew on each page. On rare occasions the numbering won't be right, because TEX will have decided that a footnote should have number 3, say, before realizing that it will actually wind up on the next page and should therefore have the number 1. On such occasions you can say

```
\adjustfootnote{-2}
```

right before the footnote, to get the numbering decreased by 2 as needed.

Notice that the space or ⟨carriage-return⟩ after the final } of the \footnote is what provides the space before the next word "you". If you don't want the space[2], you just omit it:

```
the space \footnote {...},
you just omit it:
```

4) Use \dots for the three dots ... that indicate an ellipsis:

```
the three dots \dots that indicate an ellipsis:
```

5) A % on a line of input causes TEX to ignore everything on the line from that % to the end, *including* the ⟨carriage-return⟩ at the end. So

```
... by Schwartz's Lemma%   Is this how you spell Schwartz?
we have
```

prints out

by Schwartz's Lemmawe have

which isn't quite what you had in mind when you wrote that reminder to yourself; be sure to type

```
... by Schwartz's Lemma %Is this how you spell Schwartz?
we have
```

to get the space after "Lemma", since the ⟨carriage-return⟩ that normally gives it has been obliterated by the %. You can take advantage of this behavior of % to break a long word near the end of a line right in the middle.

If you want to "comment out" a large portion of text, instead of putting % in front of each line, you can type

```
\comment
...
...
\endcomment
```

and everything between \comment and \endcomment will be ignored. But the \endcomment must be on a line by itself (although it can be preceded by blank spaces).

---

[1] Here is a footnote

[2] When TEX set the footnote at the top of this page it was still working on the previous page (since it didn't know yet just where the best page break would be). Luckily, that footnote still got the number 1, because there were no footnotes on the previous page. But after TEX had completed the previous page, it started numbering footnotes over again, so *this* footnote originally got the number 1 also! I had to put \adjustfootnote{1} right before it.

6) For the proper spacing between quotes within quotes, use \qspace:

```
``They call this `typesetting'\qspace'' he sneered.
```

This will work for 'qspace" or "', and `" or "`.

7) Use \dash for an em-dash (—) that can go at the beginning of a line, if you really need it to avoid a bad line break (normally, TEX will only put an em-dash at the end of a line, not at the beginning of one). Similarly, use \slash instead of / in a word like and/or if you want to allow the "and/" to be at the end of one line, with the "or" at the beginning of the next.

8) To get an paragraph to be unindented, use \noindent at the beginning of it.

9) Between paragraphs you can use \smallvspace, \medvspace and \bigvspace to get a small, medium or big amount of vertical space; there is a \bigvspace before this \noindent'ed paragraph, a \medvspace before the previous one, and a \smallvspace before that one.

10) Between paragraphs if you type \midspace{⟨dim⟩} for some "dimension" ⟨dim⟩ you will get a space of that dimension, if it will fit on the page; otherwise it will be on the top of the next page. A "dimension" is a number followed by a unit like in (inches), cm (centimeters), pt (point) or pc (pica), to name the ones most likely to be used. If you type \topspace instead of \midspace the space will be at the top of the current page if it fits, or at the top of the next page.

11) \linebreak breaks a line right where it is typed, without spreading the line out. \newline spreads the line out.

\pagebreak breaks the page immediately after the present line is completed. \newpage does the same, but fills up the remainder of the page with blank space, instead of spreading out the lines if the page is short.

12) Use \TeX to get the TEX logo, and \AmSTeX to get the $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX logo.

More About Document Formatting

Topmatter. At the beginning of your file, right after the \documentstyle line, there will normally be

```
\topmatter
...
...
\endtopmatter
```

for the "topmatter", like the title, author, etc. (some of which may actually end up at the bottom of the title page, or even at the end of the paper). Here are the things you can have (the order is irrelevant):

```
\title...\endtitle
\author...\endauthor
\affil...\endaffil
\address{...}
\date{...}
\thanks{...}
\keywords{...}
\subjclass{...}
\abstract{...}
```

The first three use a different syntax because each of them can be made multi-line, by inserting \\ between the lines. \address can be used several times in a row, for different authors, say. In the amsppt style, the addresses appear at the end, along with the \keywords, prefaced by "*Keywords.*" and the \subjclass, prefaced by "1980 *Mathematics subject classifications.*" The abstract begins with the word "ABSTRACT.", so this shouldn't be typed in.

Sometimes acknowledgements and affiliations are handled as footnotes on the title and/or author, instead of using \thanks and \affil...\endaffil. That's no problem—\footnote can be used in \title...\endtitle and \author...\endauthor (and most other places, as well, although the footnote may disappear if you bury it really deeply within some complicated construction—don't worry about it until it happens).

There's also \TITLE...\endTITLE and \AUTHOR...\endAUTHOR to automatically print the title and author in capital letters. Normally, these will be used by various journal styles, when it is desirable to have the title and author appearing in the file the same way that it may appear in some index, say. In fact, various journal styles will probably just have \title mean \TITLE, so that there really isn't any need to use \TITLE yourself.

There are a couple of other neat little things that you can do near the beginning of the file. First of all, you can say \guidelines before \document to get horizontal lines at the top and bottom of the page—these can be useful when your output come out on a roll of paper, and you can't figure out just where to cut it. The guidelines are normally about two lines above and below the actual page. You can say \guidelinegap⟨dim⟩ to specify some other dimension ⟨dim⟩ for the gap.

In a few places the amsppt style assumes that your paper is in English, for it begins the abstract with the word "ABSTRACT.", puts the logo "Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX" at the bottom of the first page, labels the bibliography (which we'll get to later) as "REFERENCES", and prefaces the \keywords and \subjclass with appropriate English words. If you put \german before the \document, however, all of these will come out in German. Actually, at the moment only "ABSTRACT." will be changed, to "ZUSAMMENFASSUNG.", since I don't know what the proper German for the other things is. But this will be changed as soon as someone tells me, and other languages will be added as requested.

Incidentally, if you can't stand having the "Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX" logo at the bottom, no matter what language it is in, put \nologo at the beginning.

The paper proper. The heading for this section was made by typing

```
\heading More About\\Document Formatting\endheading
```

with \\ indicating linebreaks, as usual. \subheading{...} provides the boldface flush-left headings like '**The paper proper.**' (the period is supplied automatically); this can't be multi-line, so the syntax is different.

To state theorems, lemmas, and other proclamations, you just say

```
\proclaim{Theorem 4} Blah, blah, blah.
\finishproclaim
```

and you will get

THEOREM 4. *Blah, blah, blah.*

Notice that the punctuation after the label of the theorem comes for free. All \finishproclaim really does is switch back to regular type, and perhaps provide a little extra spacing, so if you forget it the paper will look strange, but it won't be catastrophic. We use \finishproclaim instead of \endproclaim because the \end... syntax is reserved for things with \\ in them; but amstex recognizes \endproclaim as a synonym for \endproclaim, in case you forget.

For the proofs, use

```
\demo{Proof} Here is the proof.
\finishdemo
```

Once again, omitting \finishdemo will not be catastrophic—all it will lose is the special spacing after the end of the proof.

Mathematicians frequently list things like this:

(a) The first thing on the list. We will make this very long so that you can see what happens when the condition is more than one line long,
(b) The second thing,
(c) The final thing.

You get this by typing

```
\conditions
(a) : The first thing on the list.  We will make this very long so that you
can see what happens when the condition is more than one line long,\\
(b): The second thing,\\
(c): The final thing.\endconditions
```

As usual, \\ indicates the line breaks in this construction: the colon separates the label for the condition from the condition itself. In the **amsppt** style the label gets printed in \rm, unless some other font is explicitly given, while the condition is printed in whatever font is being used at the time (so that \conditions in a \proclaim will get printed in \sl). When you need "runin" conditions like (a) The first,
(b) The second,
(c) The last.
just type \runin right before the \conditions.

By the way, sometimes you may not want to start a paragraph after the statement of a theorem, but you may find that \finishproclaim does it for you, even though you didn't leave a blank line. The same problem will arise with \finishdemo and \endconditions. You will then need to use \noindent; in later versions of $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX this annoyance will go away.

There are \proclaimnp and \demonp, if you need to have the \proclaim or \demo supply No Punctuation.

Finally, there's \refto. \refto{10} gives [**10**], while \refto{10, Theorem 4} gives [**10**, Theorem 4], with only the stuff before the comma in boldface.

BIBLIOGRAPHY

When you are ready to type the bibliography, you first type \Refs, which produces the heading REFERENCES, or whatever the particular style uses, and sets things up for typing individual references. A typical individual reference would be

```
\ref \no 9 \by S. S. Chern \pages 947--055
\paper Integral formulas for hypersurfaces in Euclidean space and
  their applications to uniqueness theorems
\yr 1959 \vol 8
\jour J. Math. Mech.
\endref
```

which in the **amsppt** style will produce

9. S. S. Chern, *Integral formulas for hypersurfaces in Euclidean space and their applications to uniqueness theorems*, J. Math. Mech. **8** (1959), 947–055.

Notice that there is no need to specify the various pieces of the reference in the order that they will be printed, nor do you have to worry about the proper font or details like enclosing parentheses; all this is done by the style file. Notice also that we don't have \␣ after Math. and Mech.; within \Refs all spaces will count as usual interword spaces, even those after periods (if you do need the usual space after a period, for example if a title consists of two sentences, you can always use \period, which has already been mentioned in another context).

There is no specific amount of information that you have to include in each \ref...\endref. The style file will do the reasonable thing with the information you give. For example, if you leave out the \vol, then it just won't get printed, but if you leave out \jour, then \vol and \yr will be ignored, even if you put them in.

Other things you can have within \ref...\endref are \toappear, which typesets "(to appear)" and \issue, which might be needed for some sort of special issue. If the next reference is by the same author, you can type \bysame instead of \by ...; in the **amsppt** style this will produce a line just the length of the author's name in the previous reference instead of repeating it.

Instead of \no for a number you can use \key for some other sort of "key", like \key[{\bf C1}] to get "[**C1**]" instead of a number in front of the reference. Both \key and \no can be omitted.

If the paper is just one page long, you might use **\page** instead of **\pages**; this will make sure that the single page gets "p. " printed in front of it to avoid confusion.

Finally, there is indeed a use for \\ in **\ref**...**\endref**, though it's for special things, where two references by the same author are combined because they are closely related. For example,

```
\ref\no4 \by L. Auslander \paper On the Euler characteristic of compact
locally affine spaces \jour Comment. Math. Helv. \vol 35 \yr 1961
\pages 25--27\\ \paper II \jour Bull. Amer. Math. Soc.
\vol 67 \yr1961\pages 405--406 \endref
```

will give

4. L. Auslander, *On the Euler characteristic of compact locally affine spaces*, Comment. Math. Helv. **35** (1961), 25–27; *II*, Bull. Amer. Math. Soc. **67** (1961), 405–406.

For book references there are \book, \publ and \publladdr. For example,

```
\ref\no7\by H. Bass\book Algebraic $K$-theory\publ W. A. Benjamin
\publladdr New York\yr 1968\pages 15--19\endref
```

gives

7. H. Bass, "Algebraic K-theory", W. A. Benjamin, New York, 1968, pp. 15–19.

Notices that in this cases \pages gave "pp. " before the pages.

For a paper that appears in a book, rather than in a journal, just use **\paper** together with **\inbook**.

Finally, there are **\paperinfo**, **\bookinfo** and **\finalinfo** to put in extra information after the paper, after the book, or at the very end. Punctuation around **\paperinfo** and **\bookinfo** is handled automatically. Punctuation for **\finalinfo** has to be supplied.

MATHEMATICS

Getting Into Mathematics. Math formulas in text are indicated by **$**...**$**; displayed math formulas by **$$**...**$$**. Within a math formula, spaces are completely irrelevant (except, of course, the spaces needed at the end of control sequences).

Special Symbols. Aside from the symbols <, >, +, = and |, which are probably on your keyboard, you will need control sequences for most of the other special symbols of mathematics. They are listed in Appendix F of The TEXbook. Some important ones are

\le or \leq	$\leq$ ("less than or equal")
\ge or \geq	$\geq$ ("greater than or equal")
\ne or \neq	$\neq$ ("not equal")
\in	$\in$
\notin	$\notin$
\infty	∞

All the Greek letters have special control sequences to name them, like **\alpha** (α), **\beta** (β), **\gamma** (γ) and **\Gamma** (Γ). Several characters have variants:

```
$$(\epsilon,\phi, \theta,       \pi)$$
```

yields

$$(\epsilon, \phi, \theta, \pi)$$

while

```
$$(\varepsilon, \varphi, \vartheta,\varpi)$$
```

yields

$$(\varepsilon, \varphi, \vartheta, \varpi)$$

Switching Fonts. You can also switch to other fonts within a math formula, exactly as in text. If you type

```
$${\rm a + \bf b - c}/d$$
```

you will get

$$\mathrm{a} + \mathbf{b} - \mathrm{c}/d$$

Notice that, just as in text, font changes work for the whole formula, not just for the next letter. Only the "variables" change fonts; symbols like + and − stay the same.

There's a font called \cal that gives "calligraphic" characters:

```
$${\cal A} + B$$
```

gives

$$\mathcal{A} + B$$

This font should only be used for upper case letters. However, there's also a character ℓ, which you get with the control sequence \ell.

One other font change that's useful for math is \mit, the "math italic" font. This should be used only for capital Greek letters, to give things like $\mit\Gamma$ (\$\mit \Gamma\$), as compared to the ordinary Γ, which you get simply by typing \$\Gamma\$ (or even \$\rm \Gamma\$).

Bigger Parentheses, etc. In addition to the ordinary size parentheses that you get in math formulas, like $f(x+y)$, you can get slightly larger ones with \bigl and \bigr, which also apply to things like [and] and \{ and \}:

\bigl(X \bigr)	$\bigl(X \bigr)$
\bigl[X \bigr]	$\bigl[X \bigr]$
\bigl\{X \bigr\}	$\bigl\{X \bigr\}$
\bigl|X\bigr|	$\bigl\|X\bigr\|$
\bigl\\|X\bigr\\|	$\bigl\\|X\bigr\\|$

These are useful for formulas like

$$\bigl(x - f(y)\bigr)\bigl(x + f(y)\bigr)$$

There are also \biggl and \biggr sizes, which fit nicely around two-line fractions:

\biggl(\biggr) $\biggl(\biggr)$

Superscripts and Subscripts.

Superscripts are indicated by ^, subscripts by _ (that's the "underscore" key, not the hyphen). You can also use \sp and \sb. The symbols ^ and _ apply only to the next character, so you need braces if you want them to apply to more:

\$x^\alpha\$	x^α
\$x_{x+y}\$	x_{x+y}
\$x^{(x+y)}\$	$x^{(x+y)}$
\$x^{12}\$	x^{12}

Notice that \alpha is just one thing in TEX's mind, so no braces are needed there, but the (typeset) number 12 is two things.

For things like

$$a^{b^c}$$

just think of the formula as a mathematician does: the a has a superscript, and this superscript is—not b, but the whole formula b^c. So you type

```
$$a^{b^c}$$
```

You can have simultaneous superscripts and subscripts:

```
$A_a^b$
```
A_a^b

Remember that \prime ($\prime$) is special, because you need to superscript it to get it the right size:

```
$f^\prime$
$f^{\prime\prime}$
```
$f^\prime$
$f^{\prime\prime}$

But any number of single quote marks (') in a row in a math formula will translate into ^{\prime\prime...} with the appropriate number of \prime's:

```
$f''_2$
```
f''_2

(You'll still need \prime for the occasional formula like

```
$f^{\prime2}$
```
$f^{\prime 2}$

which is the common way to typeset such things.)

Fractions, etc. To get a fraction like

$$\frac{a}{b}$$

you type

```
$$\frac ab$$
```

and to get

$$1+\frac{a+b}{c+d}$$

you type[1]

```
$$1+\frac {a+b}{c+d}$$
```

Notice that

```
$$\frac {\frac ab}{c+d}$$
```

gives

$$\frac{\frac{a}{b}}{c+d}$$

where the numerator is now in the size normally used for text. TEX uses four sizes for mathematics formulas: \dsize is the size used for display, \tsize the size for text, \ssize the size for superscripts, and \sssize the size for superscripts to superscripts (the letter c in a^{b^c}). You can get the fraction $\frac{a}{b}$ in the above formula bigger by typing

```
$$\frac {\dsize\frac ab}{c+d}$$
```

to get

$$\frac{\dsize\frac{a}{b}}{c+d}$$

The size change macros \dsize, etc., work just like font changes—they change sizes permanently within a formula, or in some subformula delimited by braces. These are the only macros that work this way.

An easier way to get the above formula is to use \Frac, which automatically makes the \frac into \dsize:

```
$$\frac {\Frac ab}{c+d}$$
```

[1] If you are enamored of $\mathcal{AMS}$-TEX's old way of doing things, just use TEX's primitive \over; in the old $\mathcal{AMS}$-TEX \frac was just a new name for \over, but \over seems to cause so much confusion that it's been replaced by a control sequence that works like all the others. Sometimes \over will save a pair of braces (though sometimes the new \frac will also), but you often have to think ahead and put in certain braces before their time has come, and it seems easier to do things the new way.

In addition to \frac, there's \stack and \binom:

```
$$\stack ab$$
$$\binom {n+1}k$$
```

$$\stack ab$$

$$\binom {n+1}k$$

\binom is common; \stack occurs in only a few special situations, that we'll see later. There's also \Binom to get the \binom in \dsize (but there's no special \Stack, since that's almost never wanted). Finally, there's \thickfrac for a fraction with a thick line (rarely used):

```
$$\thickfrac {\Frac ab +\Frac cd}{C+D}$$
```

$$\thickfrac {\Frac ab +\Frac cd}{C+D}$$

Variable Size Symbols. You can underline and overline formulas:

```
$$\underline{x+y+z}$$
$$\overline{x+y+2^{\underline 2}}$$
```

$$\underline{x+y+z}$$

$$\overline{x+y+2^{\underline 2}}$$

There are also \overrightarrow, a.k.a. \overarrow, \overleftarrow, \overleftrightarrow, \underrightarrow, \underleftarrow and \underleftrightarrow, all of which do the obvious thing.

\sqrt gives square roots automatically of the right size:

```
$$\sqrt 2$$
$$\sqrt{\frac ab+1}$$
```

$$\sqrt 2$$

$$\sqrt{\frac ab+1}$$

Most important, you can get variable size parentheses, etc., using \left and \right:

```
$$\left(\frac 1{1-x^2}\right)$$
$$\left[\frac 1{1-x^2}\right)$$
```

$$\left(\frac 1{1-x^2}\right)$$

$$\left[\frac 1{1-x^2}\right)$$

Note that the \left and \right things don't have to match; you can even do things like \left]. But every \left needs a matching \right, so for special things like

$$\frac{dy}{dx}\Big|_{x=a}$$

there are \left. and \right. to give a \left or \right empty symbol:

```
$$\left.\frac {dy}{dx}\right|_{x=a}$$
```

Other possible delimiters are:

| (or \vert)	$\|$
\\| (or \Vert)	$\|\|$
\{	$\{$
\}	$\}$
\lfloor	$\lfloor$
\rfloor	$\rfloor$
\langle	$\langle$

`\rangle`	$\rangle$
`\slash`	$/$

The last three are special, because they come in only a limited number of sizes; the others are, in the largest sizes, built up of extensible pieces:

$$\left(\right)$$

Large Operators. The next thing you want to know about are "large operators", which include things like $\int$ (`\int`) and $\sum$ (`\sum`), as well as related things to be found in Appendix F of The TEXbook. The important thing to remember about `\sum` is that when you type `\sum_{i=1}^na_i` between dollar signs it comes out as $\sum_{i=1}^n a_i$, as expected, but when you type the very same thing as a displayed equation it comes out as

$$\sum_{i=1}^{n} a_i$$

which is the way such formulas are usually displayed: Not only does the $\sum$ sign change size, but the sub- and super-script are set as "limits".

`\int`s are similar, except that they merely change sign; normally

```
$$\int_a^b f(x)dx$$
```

gives

$$\int_a^b f(x)dx$$

Some printers set things differently, so the format you are using may use "limits" for `\int`s, and not use them for `\sum`s. You can change things yourself in the **amsspt** style by typing

```
\LimitsOnInts
```

to get limits on `\int`s and

```
\NoLimitsOnSums
```

to avoid having limits on `\sum`s. (And, `\NoLimitsOnInts` and `\LimitsOnSums` get you back again.) But these should be used only for changing the **amsppt** style; most journals insist on their own styles, and their style files will then simply ignore these instructions

Occasionally, no matter what style is being used, you need to change the way an individual $\sum$ or $\int$ is set. This is easily done by following it with `\limits` or `\nolimits` to tell TEX that limits should or should not be used. This works even in text. For example, `$\sum\limits_{i=1}^N a_i$` gives $\sum\limits_{i=1}^{N} a_i$, as opposed to $\sum_{i=1}^{N} a_i$, which we got with `$\dsize\sum_{i=1}^N a_i$`.

Whatever convention is being used for `\sum` will also hold for most of the other large operators, like $\prod$ (`\prod`). But the symbol $\oint$ (`\oint`) will be treated like $\int$. The same is true for

`$$\intii$$`	$$\iint$$
`$$\intiii$$`	$$\iiint$$
`$$\intiv$$`	$$\iiiint$$

```
$$\intdotsint$$
```
$$\intdotsint$$

One of the things to note about \sum is that you usually *don't* want to type

```
$$\left(\sum_{i=1}^N a_i\right)$$
```

because it gives

$$\left(\sum_{i=1}^N a_i\right)$$

instead of the preferred

$$\left(\sum_{i=1}^N a_i\right)$$

This formula was typed using \biggl and \biggr.

One other thing you sometimes see is something like

$$\sum_{\substack{1\le i\le N\\ 1\le j\le M}} a_{ij}$$

Here we use \stack:

```
$$\sum_{\stack{\ssize1\le i\le N}{\ssize1\le j \le M}}a_{ij}$$
```

Notice that \ssize was needed for both 1\le i\le N and 1\le j \le M, since they would normally be in \ssize, since the whole stack is in \ssize.

Spacing. You seldom have to tell TEX to put in spacing in math formulas. But there is one case where you always should. The formula

$$\int_a^b f(x)dx$$

given above really isn't right because it's conventional to leave a certain amount of space before the dx. This space, which printers call a "thin space", is called \, by TEX. So you should type

```
$$\int_a^bf (x) \, dx$$
```
$$\int_a^b f(x)\,dx$$

The other case where you need to specify spacing is in a formula like

$$f(x,y) = 0 \qquad x,y > 0$$

The space before the "side-condition" $x, y > 0$ is traditionally two "quads", where a quad is another standard printer's space. $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX has \quad for this amount of space and \qquad for the more usual \quad\quad, so the above formula was typed as

```
$$f(x,y)=0\qquad x,y>0$$
```

Ordinary type. Sometimes you have a formula like

$$f(x,y) = 0 \qquad \text{for all } x,y > 0$$

that includes ordinary type. One way to get this is to switch to \rm:

```
$$f(x,y)=0\qquad{\rm for all\ }x, y > 0$$
```

Notice that we had to sequester the `\rm` so that it didn't affect the x and y. Notice also that we had to use `\␣`, since spaces are ignored in math. For such situations it is almost always better to use $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX's `\text{...}`, which puts you back into ordinary text:

```
$$f(x,y)=0\qquad\text{for all }x,y>0$$
```

Now all we need to remember is that space after the `all` (it doesn't have to be `\␣` since we're out of math when we're in `\text`). Even easier is to go into `\text` and then within `\text` switch back to math!

```
$$f(x,y)=0\qquad\text{for all $x,y>0$}$$
```

Just be sure you get the order of `$` signs and `}`s correct.

Sometimes roman type is used in a different way, as in the formula

$$\sin(x+y) = \sin x \cos y + \cos x \sin y$$

Here sin stands for a special mathematical operator, and the spacing follows certain important rules; note, for example, that there is no space between sin and $(x+y)$, but there is space between sin and x. You don't have to worry about any of this, just use `\sin`:

```
$$\sin  (x+y) = \sin x \cos y + \cos x\sin y$$
```

Other familiar operators are

```
\arccos  \cos   \csc  \exp  \ker     \limsup  \min  \sinh
\arcsin  \cosh  \deg  \gcd  \lg      \ln       \Pr   \sup
\arctan  \cot   \det  \hom  \lim     \log      \sec  \tan
\arg     \coth  \dim  \inf  \liminf  \max      \sin  \tanh
```

Some of these have "limits" just like $\sum$:

```
$$\max_{1\le n\le m}\log_2P_n$$
```

$$\max_{1\le n\le m} \log_2 P_n$$

The control sequence `\operatorname` produces new gadgets like `\sin`, and `\operatornamewithlimits` produces new gadgets like `\max`. If you say

```
\define\gadget{\operatorname{gadget}}
\define\Gadget{\operatornamewithlimits{Gadget}}
```

then you can type

```
$$\gadget^2(x+y)-\gadget x + \Gadget_{i<j}a_{ij}$$
```

to get

$$\operatorname{gadget}^2(x+y) - \operatorname{gadget} x + \operatorname*{Gadget}_{i<j} a_{ij}$$

Numbered equations. If you type

```
$$x=y\tag 3--1$$
```

you will get

$$x = y \tag{3–1}$$

or

$$x = y \tag{3–1}$$

depending on the format style. The **amsppt** style puts tags on the left. Typing `\TagsOnRight` puts them on the right (and `\TagsOnLeft` get them back to the left again). Like the control sequences `\LimitsOnInts`, etc., `\TagsOnRight` and `\TagsOnLeft` will be ignored by the style files for journals.

Note that the parentheses, or whatever, around the tags are put in automatically by the style file. Note also that the tag was processed as text, rather than as mathematics—so the `--` gave an en-dash rather than two hyphens. For a tag like "2′" you will need to type

```
$$x=y\tag {$2'$}$$
```

(making sure to have the math part `$2'$` inside braces). However `\TagsAsMath` automatically treats all tags as math, so that `...\tag 2'$$` will give the tag (2′) [while `...\tag \text{2--3}$$` will give the tag (2–3)]. `\TagsAsText` gets you back to tags treated as text. Journal formats will not ignore the control sequences `\TagsAsMath` and `\TagsAsText`, since these do not actually effect the form of the output, merely the way that it is specified.

Finally, we ought to mention that a tag might end up on the line just before an equation (or on the line just after it, in the case of right-tagging), if the tag and the equation won't fit nicely on one line. But TEX does all this for you automatically, so you don't have to worry.

Aligned Equations. You get

$$\begin{aligned} x &= y = z \\ x^2 &= y^2 \\ x^3 &\le y^3 \end{aligned}$$

by typing

```
$$
\align x&=y=z\\
 x^2 &=y^2\\
 x^3 &\le y^3
  \endalign
$$
```

The `&`'s come right before the symbols which get lined up, and the `\\`'s separate individual lines.

An `\align`'ed equation is a unit, so that you can get

$$\begin{cases} a = b \\ a^2 = b^2 \end{cases} \qquad \begin{cases} c = d \\ c^2 = d^2 \\ c^3 = d^3 \end{cases}$$

by typing the following, which incidentally illustrates how a couple of on-the-spot `\define`'s can make life easier:

```
$$
\define\1{\align a&=b\\a^2&=b^2\endalign}
\define\2{\align c&=d\\c^2&=d^2\\c^3&=d^3\endalign}
\left\{ \1 \right.\qquad\left\{ \2 \right.
$$
```

Similarly, you can tag an `\align`. For example,

```
$$\align a&=b\\a^2&=b^2\\a^3&=b^3\endalign \tag 3-1$$
```

comes out as

$$\begin{aligned} a &= b \\ a^2 &= b^2 \\ a^3 &= b^3 \end{aligned} \tag{3-1}$$

with the tag centered with respect to the **\align**, since it's the whole **\align** that's being tagged.

A long broken equation like

$$\begin{aligned}(a+b)(a-b) &= (a+b)a-(a+b)b\\ &= a^2+ab-ab-b^2\\ &= a^2-b^2\end{aligned}$$

can be gotten by typing

```
$$
\align (a+b)(a-b) &=(a+b)a - (a + b)b\\
   & = a^2 +ab-ab-b^2\\
  &=a^2-b^2
\endalign
$$
```

You will get the same result if you use **\broken**...**\endbroken**:

```
$$
\broken (a+b)(a-b) &=(a+b)a - (a + b)b\\
   & = a^2 +ab-ab-b^2\\
  &=a^2-b^2
\endbroken
$$
```

But things work out quite differently when you put a **\tag** on a **\broken** equation, like

```
$$\broken a&=b\\&=c\\&=d\endbroken \tag 1--2$$
```

Instead of having the tag centered on the equation you will get either

$$\begin{aligned}a &= b\\ &= c\\ &= d\end{aligned} \qquad (1\text{–}2)$$

or

$$\begin{aligned}a &= b\\ &= c\\ &= d\end{aligned} \qquad (1\text{–}2)$$

depending on the format.

[Some formats center tags on broken equations, just as if they were an ordinary **\align**. Typing **\CenteredTagsOnBrokens** will cause the **amsppt** style to do this, and **\TopOrBottomTagsOnBrokens** will get the original behavior. These control sequences are, of course, ignored by a journal format style. You can get an individual **\broken** to be tagged in the center (to save some space, perhaps) by using the construction **\cbroken**...**\endcbroken**.]

\align or **\broken** can also be used for broken equations like

$$\begin{aligned}Z = (X+Y)(A+B+C+D+E+F\\ +G+H+K+L+M+N)\end{aligned}$$

which we got by typing

```
$$\align Z&=(X+Y)(A+B+C+D+E+F\\
    &\qquad +G+H+K+L+M+N)\endalign
$$
```

Sometimes an equation is broken as follows:

$$A+B+C+D+E+F+G+H+I+J+K+L+M+N+P+Q+R+S+T+U \\ = A'+B'+C'+D'+E'+F'+G'$$

with the first line beginning near the left margin, and the second line ending near the right margin. A long equation might even require three or more lines:

$$A+B+C+D+E+F+G+H+I+J+K+L+M+N \\ +A'+B'+C'+D'+E'+F'+G'+H'+I'+J'+K'+L'+M'+N'+P' \\ = P+Q+R+S$$

You get such displays by using `\multline`:

```
$$
\multline A+B+C+D+E+F+G+H+I+J+K+L+M+N\\
  + A'+B'+C'+D'+E'+F'+G'+H'+I'+J'+K'+L'+M'+N'+P'\\
     =P+Q+R+S
\endmultline
$$
```

The lines between the first and last are centered, but you can shove any of these to the left or right by typing

`\shoveleft{`second line`}`, `\shoveright{`third line`}`, etc.

When a `\multline` is tagged the tag will either be at the beginning of the first line or the end of the last line. One caution here: If the tag and the first line together won't fit, `\multline` *won't* put the tag on a separate line, you'll have to do it by hand (by having an empty first equation). In version 2 of $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX we hope to fix this problem.

Finally, every once in a while you will need a display consisting of a bunch of equations that are individually centered, instead of being aligned:

$$a=b+c \\ d=e \\ f+g=h$$

You shouldn't type

```
$$a=b+c$$ $$d=e$$ $$f+g=h$$
```

because you'll get too much space between the equations. Instead use `\bunch`:

```
$$\bunch a=b+c\\
  d=e\\
   f+g=h\endbunch$$
```

Notice that there *must not* be any `&`'s here, since nothing is being aligned. Of course, you might well have `&`'s within some `\align` appearing within the `\bunch`. For example,

```
$$
\bunch a=b+c\\
\align d&=e\\&=f\endalign\\
 f+g=h\endbunch
$$
```

gives

$$a=b+c \\ d=e \\ =f \\ f+g=h$$

Aligned and tagged equations. For something like

$$a = b = c \tag{1}$$
$$a^2 = b^2$$
$$a^3 = c^3 \tag{2}$$

you use \aligntag:

```
$$
\aligntag a&=b=c\tag 1\\
a^2&=b^2\\
a^3&=c^3\tag 2\endaligntag$$
```

Like \multline, \aligntag won't automatically put a tag on a separate line if necessary—you may get a tag overlapping an equation, without getting an Overfull box message! In version 2 this should be fixed.

You can have a \broken inside an \aligntag, with or without a tag, and *everything* will line up nicely. For example,

```
$$
\aligntag (a+b)(a+b)&=a^2+2ab+b^2,\tag 1\\
\broken (a+b)(a-b) &=(a+b)a - (a + b)b\\
   & = a^2 +ab-ab-b^2\\
  &=a^2-b^2
\endbroken\tag2\endaligntag$$
```

gives

$$(a+b)(a+b) = a^2 + 2ab + b^2, \tag{1}$$
$$\begin{aligned}(a+b)(a-b) &= (a+b)a - (a+b)b \\ &= a^2 + ab - ab - b^2 \\ &= a^2 - b^2\end{aligned} \tag{2}$$

with the = signs of the \broken lining up with the ones from the \align. (\broken wasn't designed to behave this way inside a bare \align since there's no point using \broken instead of \align if you don't have tags to worry about.)

Between lines of an \aligntag you can type \vspace{⟨dim⟩} to get extra vertical space of dimension ⟨dim⟩ between them. If you type \xtext{...} the extra text ... will appear on a separate line, starting at the left margin, without destroying the alignment of the other lines.

Although an \aligntag is usually treated as a unit, like an \align, you can put \allowbreak between lines to allow a page break, \goodbreak to indicate an especially nice place to break, and even \break to force a break.

There's also \bunchtag, which bears the same relation to \aligntag as \bunch bears to \align.

Matrices. A "matrix" is a square array like

$$\begin{matrix} a & b & c \\ a' & b' & c+c' \\ a'' & b'' & c+c'+c'' \end{matrix}$$

You get this by typing

```
$$\matrix
  a&b&c\\a'&b'&c+c'\\a''&b''&c+c'+c''\endmatrix$$
```

with \\ separating rows, as usual, and & separating elements of the various columns. The elements within each column are centered, and there is a quad of space between the columns. The number of columns will

be the largest number specified in all the rows; rows with fewer specified columns will simply have a blank formula at the corresponding place. Thus,

```
$$\matrix
  a\\a'&b'\\a''&b''&c+c'+c''\endmatrix$$
```

gives

$$\begin{matrix} a & & \\ a' & b' & \\ a'' & b'' & c+c'+c'' \end{matrix}$$

You can get a different format for a `\matrix` by using

```
\format ...\\
```

right after `\matrix`. For example, if you want the matrix

$$\begin{matrix} x & 1 & .1 \\ x+y & 11 & .11 \\ x+y+z & 111 & .111 \end{matrix}$$

with the first column centered, the second set flush right and the third set flush left, you type

```
$$\matrix \format \c & \quad\r & \quad\l\\
x&1&.1\\
x+y&11&.11\\
x+y+z&111&.111\endmatrix$$
```

In the format line `\format \c & \quad\r & \quad\l\\` the specifications for the columns are separated by `&`, just as the elements of each row are later; the `\c` indicates that the elements of the first column are centered, the `\quad\r` indicates that the elements of the second column are set flush right, with a `\quad` of space before the column, etc. Any other spacing, like `\,` or `\;` or `\qquad` can be used instead.

There is also a special way of abbreviating formats with many columns that are all treated the same. Suppose, for example, that we want all centered columns, but separated by a thin space `\,` instead of by a `\quad`. We can just use

```
\format \c && \,\c \\
```

The double `&&` means that the remaining specifications are repeated over and over, for as many columns as is necessary. Similarly,

```
\format \c && \quad\l & \quad\r \\
```

will center the first column, and then produce columns that are alternately set flush left and flush right, with all columns separated by a `\quad` of space. You mustn't have `&&` twice in a `\format...\\`.

If you have a special format that you use many times you unfortunately *can't* just say

```
\define\specialformat{\format ... \\}
```

and then save work by typing

```
\matrix\specialformat
  a & b & ...
   ...\endmatrix
```

`\matrix` has to see the explicit sequence `\format ... \\` if it is used. (However, you can say, for example,

```
\define\specialformat{\c&&\quad\l&\quad\r}
```

and then type

```
\matrix\format \specialformat \\
  ...
```

which is almost as good.)

In a `\matrix` the elements are set as math formulas in `\tsize`; use `\dsize` if you explicitly need it.

Usually a matrix will occur with parentheses or something else around it, like

$$\begin{pmatrix} a & b & c \\ a' & b' & c+c' \\ a'' & b'' & c+c'+c'' \end{pmatrix}$$

You can always put in the parentheses with `\left` and `\right`, but there's also `\matrixp` and `\endmatrixp` to do it:

```
$$\matrixp
  a&b&c\\a'&b'&c+c'\\a''&b''&c+c'+c''\endmatrixp$$
```

There's also `\matrixv` to give `\left|`, `\matrixvv` for `\left\|` and `\matrixb` for `\left[`, with corresponding `\endmatrix`... constructions. You can even mix them, to get one thing on the left and another on the right. But be sure not to begin with `\matrixp`, say, and end simply with `\endmatrix`, since this gives a `\left(` unbalanced by a `\right` (if you really want nothing on the right use `\endmatrix\right.`).

For a matrix like

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

you use `\hdots` for the horizontal dots, and `\vdots` for the vertical dots:

```
$$
\matrixp \format \c&&\;\c\\
a_{11}&a_{12}&\hdots&a_{1n}\\
a_{21}&a_{22}&\hdots&a_{2n}\\
\vdots&\vdots& &\vdots\\
a_{m1}&a_{m2}&\hdots&a_{mn}
\endmatrixp
$$
```

(Actually, at the moment you'll have to use `\ldots` (low dots), because I forgot to add the alternative name `\hdots`.)

You might want to get something like

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

To get the diagonal dots just use `\ddots`:

```
$$
\matrixp \format \c&&\;\c\\
a_{11}&a_{12}&\hdots&a_{1n}\\
a_{21}&a_{22}&\hdots&a_{2n}\\
\vdots&\vdots&\ddots &\vdots\\
a_{m1}&a_{m2}&\hdots&a_{mn}
\endmatrix
$$
```

If instead you want

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

you type

```
$$
\matrixp \format \c&&\;\c\\
a_{11}&a_{12}&\hdots&a_{1n}\\
a_{21}&a_{22}&\hdots&a_{2n}\\
\dotsfor 4\\
\dotsfor 4\\
a_{m1}&a_{m2}&\hdots&a_{mn}
\endmatrixp
$$
```

You have to tell `\dotsfor` how many columns it should put dots in for (since `\matrix` doesn't even know for sure how many columns there are going to be until it has read the final row).

A construction like

$$f(x) = \begin{cases} x+1, & \text{for } x > 0 \\ x-1, & \text{for } x \le 0 \end{cases}$$

can be constructed from `\matrix` using `\left\{` and `\right.` but there's also `\cases` to do it all for you:

```
$$
f(x)=\cases x+1, & \text{for $x>0$}\\
          x-1, & \text{for $x\le0$}\endcases
$$
```

A `\matrix` can be used inside another `\matrix` or anything similar. For example, you can get the following matrix of parenthesized matrices

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

by typing

```
$$
\def\1{\matrixp 1&0\\0&1\endmatrixp}
\def\2{\matrixp 0&1\\1&0\endmatrixp}
\def\3{\matrixp 1&1\\1&0\endmatrixp}
\def\4{\matrixp 1&0\\1&1\endmatrixp}
\matrix\format \c\qquad&\c\\
\1&\2\\
\vspace {20pt}
\3&\4
\endmatrix
$$
```

Here we've used `\vspace` to specify extra vertical space between lines.

Dots. For things like

$$x_1 + \cdots + \|(a,\ldots,b)\| + (y_1 y_2 \ldots y_n) + \int \cdots \int f(x,y)\,dx\,dy$$

just use \dots to get the dots:

```
$$x_1+\dots+\|(a,\dots, b)\|+ (y_1y_2\dots y_n) + \int\dots\int f(x,y)\,dx\,dy$$
```

\dots figures out what kind of dots to give on the basis of the next symbol, so it probably won't give you the right kind of dots if you type something like

```
$$x_1 + x+2 + x_3 +\dots$$
```

So as a last resort, use \dotsb, the dots $\cdots$, together with the proper spacing, that go between binary operators and relations, like + and =; or \dotsc, the dots and spacing that go before a comma; or \dotsj, the dots between juxtaposed things; or \dotsi, the dots between integral signs. If you do something weird (even though perfectly legitimate) you may confuse \dots so thoroughly that you'll just end up with an incomprehensible error message. Resort to specific \dots... in that case also.

Accents. The accents in math require different names, and there are some more of them:

`$\dot x$`	$\dot{x}$
`$\dotii x$`	$\ddot{x}$
`$\dotiii x$`	$\dddot{x}$
`$\dotiv x$`	$\ddddot{x}$
`$\hat x$`	$\hat{x}$
`$\check x$`	$\check{x}$
`$\tilde x$`	$\tilde{x}$
`$\breve x$`	$\breve{x}$
`$\vec x$`	$\vec{x}$
`$\bar z$`	$\bar{z}$

The choice between $\bar{z}$ (`$\bar z$`) and $\overline{z}$ (`$\overline z$`) is one of taste.

There are also variable size versions of \hat and \tilde, called \widehat and \widetilde:

`$\widehat x,\widetilde x$`	$\widehat{x},\widetilde{x}$
`$\widehat {xy},\widetilde {xy}$`	$\widehat{xy},\widetilde{xy}$
`$\widehat {xyz},\widetilde {xyz}$`	$\widehat{xyz},\widetilde{xyz}$

Another way of accenting a long formula is to put the accent as a superscript to the whole formula in parentheses:

$$(x_1 + x_2 + \cdots + x_n)^{\sim}$$

You can't type `^\tilde` because \tilde isn't a symbol, it's an instruction to put an accent on something. But there's \tildesymbol which is a symbol:

```
$$(x_1+x_2+\dots+x_n)^\tildesymbol$$
```

Similarly, there's \hatsymbol, \vecsymbol, \barsymbol and corresponding symbols for all the dot accents. There also should be \checksymbol and \brevesymbol, but I forgot to make them, so they'll come later.

Oversetting and Undersetting. To get something like

$$\underset{\alpha\beta}{\mathbf{M}}$$

type

```
$$\underset \alpha\beta \to {\bf M}$$
```

(The underset formula is between `\underset` and `\to`, so doesn't have to be in braces even if it's more than one symbol.) Similarly, you can `\overset` something.

You can also `\underbrace` or `\overbrace` something:

```
\underbrace{x_1+x_2+\dots+x_n}
```

$\underbrace{x_1+x_2+\dots+x_n}$

And to get something like

$$\underbrace{x+\cdots+x}_{n \text{ times}}$$

you don't have to `\underset` and `\underbrace`; you can simply use `\undersetbrace`:

```
$$
\undersetbrace n \rm\ times\to {x+\dots+x}
$$
```

Boxed formulas. `\boxed` makes a formula boxed. For example,

```
$$
\boxed{x\le y}\qquad\text{for all $x\in A$, $y\in B$}
$$
```

produces

$$\boxed{x \le y} \qquad \text{for all } x \in A,\ y \in B$$

Roots. To get

$$\sqrt[\alpha+\beta]{1+\frac{a}{b}}$$

you type

```
$$
\root \alpha + \beta \of {1+\frac ab}
$$
```

If you don't like the position of the root, you can `\uproot` it or `\rightroot` it. For example,

```
$$
\uproot 3\rightroot {-3}\root \alpha + \beta \of {1+\frac ab}
$$
```

gives

$$\sqrt[\alpha+\beta]{1+\frac{a}{b}}$$

The amount of `\uproot`'ing or `\rightroot`'ing is given in terms of an arbitrary unit, that you don't specify.

Phantoms, Smashes, Etc. As the old "Joy of TEX" points out, there are numerous nefarious tricks that one can play with `\phantom{...}`, which produces an "invisible symbol" that takes up just as much room as the formula .... Even more useful is `\vphantom{...}`, which takes up no horizontal space but sticks just as far above and below the line as ..., and there is also `\hphantom{...}`, which takes up no vertical space but as much horizontal space as .... You can do even more "eye-balling" tricks now that $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX has `\smash{...}`. This leaves the formula ... in place but makes TEX think that it doesn't stick above or below the line at all! And `\topsmash{...}` just smashes the part above the line, while `\botsmash{...}` smashes the part below. There's still `\shave`, a specialty discussed in the old "Joy of TEX", but it's much easier to use now: Just say

```
\shave{\sum_{i=1}^Na_i}
```

to shave off any excess space above the formula

$$\sum_{i=1}^{N} a_i$$

There's also `\topshave` and `\botshave`, as before.

ERRATA

Page 3, line –5, Change `\Mr.@Jones` to `\Mr.~Jones`.

Page 5, line 3, Change "This will work for 'qspace" or ... " to "This will work for '" or ... ".

Page 6, line 14, `\guidelinegap⟨dim⟩` needs braces: `\guidelinegap{⟨dim⟩}`.

Page 6, line –12, Replace "amstex" by the $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX logo.

Page 13, line 14, "`\ssize`, since ... " should be "`\sssize`, since ... ".

Page 16, lines –6–end of page,

$$\begin{aligned} Z = (X+Y)(A+B+C+D+E+F \\ +G+H+K+L+M+N) \end{aligned}$$

which we got by typing

```
$$\align Z&=(X+Y)(A+B+C+D+E+F\\
    &\qquad +G+H+K+L+M+N)\endalign
$$
```

is really supposed to be

$$\begin{aligned} Z = (X+Y)(A+B+C+D+E+F \\ +G+H+K+L+M+N) \end{aligned}$$

which we got by typing

```
$$\align Z=(X+Y)&(A+B+C+D+E+F\\
    &\qquad +G+H+K+L+M+N)\endalign
$$
```