1 Introduction

These notes are miscellaneous remarks on some technical questions applicable to version 2.0 of AMSP-TEX.

2 Deprecated and disallowed commands

Certain commands that were in the amstex package (the predecessor, in AMSP-TEX 1.0 and 1.1, of the amsmath package) were moved into amsxtra because they seemed to be little-used relics: \accentedsymbol, “sup accents” (\sptilde, \sphat, etc.).

Certain other commands—e.g., \over, \pmatrix, and \cases—are changed by the amsmath package to produce warning messages or error messages, depending on their history; these are discussed in more detail in the following sections.

3 Why do \matrix, \pmatrix, and \cases stop working when I add the amsmath package?

If you used the plain.tex versions of \matrix, \pmatrix, or \cases in a document and then later converted the document to use the amsmath package (or one of the AMS documentclasses, which automatically call the amsmath package internally), the instances of those commands will produce error messages. The problem is that when \TEX was originally created, it adopted most of its mathematics features straight from plain.tex. But in the case of \matrix, \pmatrix, \cases this was a mistake—the plain.tex syntax for them is decidedly non-\TEX in style, for example the fact that they use \cr instead of \ to mark line breaks, and they don’t use \begin and \end. In basic \TEX this mistake will be perpetuated at least until \TEX3 appears, in order to avoid breaking existing documents. But no existing documents that were written with the amsmath package have that syntactic problem, as amsmath provides proper \TEX-syntax versions of \matrix and the others. The possibility of optionally allowing the plain.tex variants to make document conversion easier seems ill-advised since those variants are so blatantly wrong in a \TEX context. The
array environment ought to have been used instead.

4 Why do $\textbackslash$over, $\textbackslash$atop, $\textbackslash$above [...withdelims] give a warning when I add the amsmath package?

Using the six generalized fraction commands $\textbackslash$over, $\textbackslash$overwithdelims, $\textbackslash$atop, $\textbackslash$atopwithdelims, $\textbackslash$above, $\textbackslash$abovewithdelims is not really a good idea in \TeX, for reasons explained below. I construe \TeX's provision of $\textbackslash$frac, and the lack of any mention in the \TeX book of the primitive fraction commands, as an implicit injunction against their use, although I don't think Lamport actually spent a lot of time pondering the issue, and the basic \TeX version of $\textbackslash$frac provides access only to $\textbackslash$over, not to $\textbackslash$atop, $\textbackslash$above, or the withdelims variants. The amsmath package, however, provides a $\textbackslash$genfrac command that gives user-level access to all six of the generalized fraction primitives in a way that conforms to the syntactic conventions followed by all other \TeX commands.

Not only is the unusual syntax of the \TeX primitives rather out of place in \TeX, but furthermore that syntax seems to be 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. To cite two of the worst side effects: $\textbackslash$mathchoice must actually typeset all four of its arguments, instead of being able to immediately select only one; and, were it possible to always know the current math style at a given point, math font selection would be greatly simplified and the upper limit of 16 different math font $\textbackslash$fams would never be a problem as $\textbackslash$text,$\textbackslash$script,$\textbackslash$scriptfont assignments for any $\textbackslash$fam could take immediate effect and therefore could be changed arbitrarily often within a single formula. More concretely, math font selection difficulties are responsible for many of the more convoluted passages in the source code of \TeX's NFSS (that does font loading on demand) and of the amsmath package, and by extension it has historically been responsible for significant delays in making new features available to end users and for making those features more prone to bugs.

There are additional bad consequences following from the syntax of those generalized fraction commands that only become evident when you do some writing of nontrivial macros for math use. For example, as things currently stand you cannot measure the size of any object in math without going through $\textbackslash$mathchoice and leaving and reentering math mode via $\textbackslash$hbox{$\textbackslash$} (which then introduces complications regarding $\textbackslash$everymath and $\textbackslash$mathsurround). And it seems that uncertainty about the current mathstyle is the only barrier to allowing the use of \mu units with $\textbackslash$vrule, to make vertical struts in constructing compound symbols or notation. And so on and so forth.
5 The fleqn option and \mathindent

Strictly speaking, the amsmath package doesn’t use \mathindent to control the left indent of displayed equations when the fleqn option is in effect: it uses an internal parameter @mathmargin instead. However, for compatibility with existing $\LaTeX$ documentation, amsmath turns \mathindent into an alias for @mathmargin. There is a small risk here: In the plain $\LaTeX$ implementation, \mathindent is a dimen register, but with amsmath @mathmargin is a skip register and, by association, so is \mathindent. If any package or documentclass uses \mathindent in a way that depends on it being a dimen register, when used in conjunction with the amsmath package it may be vulnerable to a well-known pitfall having to do with the primitive $\TeX$ lookahead for a plus or minus key word. However if the standard $\LaTeX$ commands \setlength and \addtolength are used to modify \mathindent then this problem will not arise.

6 Why can’t I use abbreviations for \begin{align} ... \end{align}?

Authors often like to use abbreviations such as \beq and \eeq instead of \begin{equation} ... \end{equation}. For some environments defined by the amsmath package, such as align, gather, multline, and others of the same general type, this does not work: An attempt to define \bal and \eal as shorthand for \begin{align} ... \end{align} will fail with a puzzling error message. This has to do with unfortunately nontrivial technical complications: the given environments must read their contents as a delimited macro argument because they do multipass processing of the contents using algorithms inherited from Spivak’s amstex.tex. The obvious solution—substitution of different algorithms that do box shuffling instead of token shuffling for the multipass calculations—would require rewriting these display environments from the ground up; while that is a worthy goal, it was beyond the original scope of the AMS-$\LaTeX$ project. Work is under way on an auxiliary package called breqn that addresses not only this problem but a number of others; at the time of this writing, however [September 1999] it has only progressed as far as a beta release.

Some workarounds:

- \def\bal#1\eal{\begin{align}#1\end{align}}
- Define \newcommand{\env}[2]{\begin{#1}#2\end{#1}} and then use \env{align}{...}

7 The upref package

The reason for splitting out the upref package instead of automatically incorporating it in the amsart and amsbook classes is this: It involves low-level surgery on an important $\LaTeX$ command. This means that if ever this command changes in the future (as it did between versions 2.09 and 2e of $\LaTeX$) we have a maintenance problem. And the benefit that upref provides is something that most end users don’t care much about.
8 The amsintx package

After a very preliminary trial release the amsintx package was withdrawn to await further development. It is still considered a worthy project but has simply not yet made it to the head of the priority queue in the AMS \LaTeX{} development program, as it has had to contend with a number of other equally worthy projects for development time.

9 Hyphenation in the documentation

Hyphenation was allowed for certain long command names in amsldoc.tex; this presented technical difficulties because \LaTeX{} normally deactivates hyphenation for tt fonts. The method chosen to reinstate hyphenation was to turn off the encoding-specific function \OT1+cmtt that disables the \hyphenchar for tt fonts; see the definition of \allowtthyphens in amsdtex.dtx. Then a list of all tt words in the document was gathered (from the .idx file, produced by the \cn, \fn, \pkg, etc. commands) and \showhyphens was applied to this list. The result was another list in the resulting TeX log, containing those words in a form suitable for the argument of \hyphenation. That list was then edited by hand to overrule undesirable hyphenations; words with acceptable hyphenations were dropped from the list, as they don’t need to be repeated there.