Some Problems with the INRSTEX Table Making Macros

Michael J. Wichura
University of Chicago

The INRSTEX table making macros distributed with TEXniques Number 2 greatly simplify the task of making ruled tables in TEX. There are, however, some (admittedly uncommon) circumstances under which the macros don't work as advertised or have adverse consequences. This article brings to light several such problems and suggests ways to correct them. Be forewarned that the issues here are fairly TEXnical — the kind of material that is marked with dangerous bends in *The TEXbook*.

1 A problem with \left, \right, and \-

The INRSTEX macros break what I would propose as the first commandment for authors of macro packages: *Thou shalt not redefine a TEX primitive.* `\left` and `\right` are TEX primitives that are used to make variable size delimiters in math mode. Within the table environment, however, INRSTEX preempts the meanings of `\left` and `\right` to control the positioning of items within columns. The original meanings of `\left` and `\right` are gone, and macros, such as Plain TEX's `\big`, `\bigg`, `\Big`, and `\Bigg`, that rely on those meanings being in force won't function properly. You'd be in for 'big' trouble if you tried to make a table whose entries involved some heavy math.

There's a similar problem with `\-`, which INRSTEX uses to draw horizontal lines in tables. `\-` is TEX's primitive for a discretionary hyphen. If you planned to make a table having (presumably narrow) paragraphs as entries, you might well want to use `\-`'s to assist TEX in hyphenation.

These difficulties wouldn't exist if INRSTEX had chosen slightly different names, say '`\Left`', '`\Right`', and '`\Hr`', for its commands. Name changing is not a feasible option at this point because the macros are already widely used. The macros can and should be supplemented by a command which makes the original meanings of `\left`, `\right`, and `\-` available within the table making environment. The command `\restoreTeXprimitives` (`\rTp` for short) defined below does just that.

```
\let\T@Xleft=\left
\let\T@Xright=\right
\let\T@Xdiscretionaryhyphen=\-
\def\restoreTeXprimitives{%
  \let\left=\T@Xleft
  \let\right=\T@Xright
  \let\-=\T@Xdiscretionaryhyphen}
\let\rTp=\restoreTeXprimitives
```

This code should be inserted at the end of the INRSTEX macros, just before the line that reads '`\catcode\@=12`'. With this code in place, you could, for example, typeset the expression

$$\left(\frac{a-1}{b-1}\right)$$

in a column with a display-style math template by entering '`\rTp \left( {a-1\over b-1} \right)`'. It's not necessary to enclose such usages of `\rTp` in a group (`{...}`), since an INRSTEX table is created with an `\halign` and since TEX automatically enters an additional level of grouping when it works on each individual entry to an `\halign`.

There's a related problem concerning `|` and `\|`, which Plain TEX uses as delimiters in math mode, but which INRSTEX preempts to draw vertical rules in tables. One could augment the definition of `\restoreTeXprimitives` to cover `|` and `\|` as well, but there's a good reason not to do so. The table making macros set things up so that `|` and `\|` signal the end of a data column. If you were to restore the original meanings of these control sequences with `\rTp`, then you would have to enclose every usage of `\rTp` in a group. That's a greater burden than having to use Plain TEX's synonym `\vert` for `|`, and `\Vert` for `\|`.

2 Problems with \zerocenteredbox

The command `\zerocenteredbox` (`\zb` for short) centers its argument vertically in a box of zero height and depth. INRSTEX provides this feature as a means of doing some makeshift vertical spanning within tables. The manual asserts that `\zb` works correctly "even with display math templates". This, however, is not the case. The problem arises because display math templates have the form `$\displaystyle{#}$` (and so should more properly be called display-style math templates), whereas `\zb` uses an `\if` type test for display math mode (what you get between pairs of `$$`'s). There is, in fact, no `\if` type test for display style. One has to use, instead, a `\mathchoice` or `\mathpalette` construction, as explained on page 151 of *The TEXbook*. The following code (re-)defines `\zb` correctly, in the way Plain TEX defines `\phantom`s and `\smash`es.

```
\def\zerocenteredbox{%
  \relax
  \ifmmode
    \expandafter\mathpalette
    \expandafter\m@thzb
  \else
    \expandafter\m@kezb
  \fi}
\def\m@kezb#1{%
  \setbox\z@=\hbox{#1}%
  \fin@shzb}
\def\m@thzb#1#2{%
  \setbox\z@=\hbox{$\m@th#1{#2}$}%
  \fin@shzb}
\def\fin@shzb{%
  \vbox to\z@{\vss\box\z@\vss}}
```

The \relax on the second line of the code is important, for reasons explained on page 240 of *The TEXbook*. There's no such \relax in INRSTEX's definition of \zb, and that version of the macro malfunctions when it appears as the first token in a table entry for a column with a math- or display-style math template.

3 Problems with \modifystrut (alias \mst) and \sa

These very useful commands allow you to fine tune the vertical and horizontal spacing in a table through appropriate struts. The commands, however, don't work in math modes. The INRSTEX manual sidesteps this issue by using constructions like '\mst{$\int$}{0pt}{3pt}'. In a column with a math template it would be preferable to enter just '\mst{\int}{0pt}{3pt}'. The following macros (re-)define \modifystrut and \sa so that they work in math modes as well as in horizontal mode. The idea is to again use a \smash-type construction; the macros for \mst are a little more complicated because they have to take into account additional arguments (e.g., the '0pt' and '3pt' in the example above).

```
\def\modifystrut#1#2#3{%
  % #1 = original
  % #2 = add to height
  % #3 = add to depth
  \relax
  \ifmmode
    \def\next{%
    \mathchoice
    {\m@thmst\displaystyle{#1}{#2}{#3}}
    {\m@thmst\textstyle{#1}{#2}{#3}}
    {\m@thmst\scriptstyle{#1}{#2}{#3}}
    {\m@thmst\scriptscriptstyle
      {#1}{#2}{#3}}}
  \else
     \def\next{\m@kemst{#1}{#2}{#3}}%
  \fi
  \next}
\def\m@kemst#1#2#3{%
  \setbox\z@=\hbox{#1}%
  \fin@shmst{#2}{#3}}
\def\m@thmst#1#2#3#4{%
  \setbox\z@=\hbox{$\m@th#1{#2}$}%
  \fin@shmst{#3}{#4}}
\def\fin@shmst#1#2{%
  \dimen\z@=\ht\z@
  \advance\dimen\z@ by #1\relax
  \dimen\tw@=\dp\z@
  \advance\dimen\tw@ by #2\relax
  \vrule width\z@
    height\dimen\z@ depth\dimen\tw@}
\let\sa=\hphantom
```

The INRSTEX manual states that the first argument to \mst can "even be a duplication of the row, as long as the row contains no explicit & characters and excluding the commands \br and \er." This assertion is correct if you use just the simple \left, \right, and \center templates, and if, as in all the examples in the INRSTEX manual, \mst appears as an argument to \br or \er. Otherwise, the assertion may be false, since \mst doesn't take templates into account, and since the definitions of the active characters | and " involve &'s when INRSTEX is working between \br and \er.

4 A problem with \use

\use{<number of data columns>} is asserted to "merge the next <number of data columns> into one and use the format or template of the last one." And so it does, unless <number of data columns> is one: '\use{1}' results in an error message from TEX. According to the intended use of \use, '\use{1}' should be the same as ⟨null⟩. To achieve this, \use should be (re-)defined as follows:

```
\def\use#1{%
  \ifnum #1>\@ne
    \omit
    \mscount=#1
    \advance\mscount by \m@ne
    \multiply\mscount by \tw@
```

```
    \loop
      \ifnum\mscount>\@ne
      \sp@n
    \repeat
    \span
  \fi}
```

5 Problems with \thrule

The command \thrule{<height>} is supposed to insert a horizontal rule of thickness <height> across an entire data column. But if you enter, say,

... \thrule{2pt}| ...

no rule is drawn; the rule *is* drawn if you enter

... \thrule{2pt}␣| ...

This puzzling discrepancy arises because \thrule uses a \leaders construction, and because INRSTEX makes | an active character whose expansion begins with \unskip when INRSTEX is working between \br and \er. \unskip removes any glue item that immediately precedes it; this is what allows the INRSTEX manual to state that the command | "removes spaces to its left." (Remember that TEX treats two or more spaces the same as one space, and that a space is (normally) a glue item.) Leaders, however, are themselves a special kind of glue, so in the first example above the \unskip removes the leaders and no rule is drawn. By contrast, in the second example the \unskip removes just the ␣ and the rule is drawn. The way to solve this problem is to place an invisible non-glue item after the leaders, as in the following re-definition of \thrule:

```
\def\thrule##1{%
  \omit \leaders
  \hrule height ##1\hfill\null}
```

This will also fix similar problems that occur with constructions such as '\use{3} \-|'.

There is another problem with \thrule. When INRSTEX's \midtabglue is non-zero, the horizontal lines \thrule draws across data columns don't join up with vertical lines in neighboring rule columns, since the horizontal lines don't span the tabskip glue. Unfortunately, there's no easy way to modify \thrule so as to fix this problem.

6 A problem with \everycr

Within the table making environment, the INRSTEX macros set the \everycr token list to

```
\noalign {\global \a@lignstate=0}
```

Consequently, if you use an ordinary \halign within a table, you'll throw off INRSTEX's accounting and the table won't come out right. To get around this problem, you should use Plain TEX's \ialign in place of \halign; moreover, you should specify \normalbaselines (minimally) before the \ialign since INRSTEX turns off normal line spacing with \offinterlineskip.

7 A problem with \sp and \om

At the end of the INRSTEX macros, \sp and \om are \let equal to \span and \omit, respectively. This is not mentioned in the manual, nor are these abbreviations used anywhere in the macros themselves. Since they serve no useful purpose, they are best deleted.

8 A test table

When the INRSTEX macros are modified in the ways suggested above, this little contrived table

hyphen-ation	$\left[\frac{A+1}{B+1}\right]$
Here's an ialign	$\left[\frac{A+1}{B+1}\right]$

results from the following code:

```
\begintable
  \def\P{\vtop{\normalbaselines
    \hsize=.5in \raggedright \noindent
    \restoreTeXprimitives
    hy\-phen\-ation}}
  \def\PP{\P \mst{\P}{3pt}{3pt}}
  \def\M{\restoreTeXprimitives
    \left [ {A+1 \over  B+1} \right ]}
  \def\MM{\M \mst{\M}{20pt}{5pt}}
  \def\I{\vbox{\normalbaselines
    \ialign{##\hfil\cr
    Here's an\cr
    ialign\cr}}}
  \begintableformat
    \center " \displaymath\center
  \endtableformat
  \-
  \br{}            " \sa{\M\M} \er{}
  \br{|} \PP       |           \er{|}
  \br{|} \use{1} \-| \zb{\M}   \er{|}
  \br{|} \I        | \MM       \er{|}
  \-
\endtable
```