# Les extensions **multirow**, **bigstrut** et **bigdelim**

Piet van Oostrum [*]
Øystein Bache
Jerry Leichter [†]

## 1 Introduction

✖These packages offer a series of extensions to the standard LaTeX `tabular` environment. Their respective functions are :

**multirow** which provides a construction for table cells that span more than one row of the table ;

**bigdelim** which creates an appropriately-sized delimiter (for example, brace, parenthesis or bracket) to fit in a single multirow, to indicate a relationship between other rows ; and

**bigstrut** which creates struts which (slightly) stretch the table row in which they sit.

## 2 Using **multirow**

The basic syntax is :

```
\multirow{nrows}[bigstruts]{width}[fixup]{text}
```

where

***nrows*** is the number of rows to span. It's up to you to leave the other rows empty, or the stuff created by `\multirow` will over-write it. With a positive value of *nrows* the spanned columns are this row and (*nrows*-1) rows below it. With a negative value of *nrows* they are this row and (1-*nrows*) above it.

***bigstruts*** is mainly used if you've used the **bigstrut**. In that case it is the total number of uses of `\bigstrut` within the rows being spanned. Count 2 uses for each `\bigstrut`, 1 for each `\bigstrut[x]` where $x$ is either `t` or `b`. The default is 0.

***width*** is the width to which the text is to be set, or `*` to indicate that the text argument's natural width is to be used.

***text*** is the actual text of the construct. If the width was set explicitly, the text will be set in a `\parbox` of that width ; you can use `\\` to force linebreaks where you like.

If the width was given as `*` the text will be set in LR mode. If you want a multiline entry in this case you should use a `tabular` or `array` environment in the text parameter.

***fixup*** is a length used for fine tuning : the text will be raised (or lowered, if *fixup* is negative) by that length above (below) wherever it would otherwise have gone.

For example (using both multirow and bigstrut) :

[*] « active author »

[†] Documentation rassemblée par Robin Fairbairns

```
\newcommand{\minitab}[2][l]{\begin{tabular}{#1}#2\end{tabular}}
\begin{tabular}{|c|c|}
\hline
\multirow{4}{1in}{Common g text} & Column g2a\\
      & Column g2b \\
      & Column g2c \\
      & Column g2d \\
\hline
\multirow{3}[6]*{Common g text} & Column g2a\bigstrut\\\cline{2-2}
      & Column g2b \bigstrut\\\cline{2-2}
      & Column g2c \bigstrut\\
\hline
\multirow{4}[8]{1in}{Common g text} & Column g2a\bigstrut\\\cline{2-2}
      & Column g2b \bigstrut\\\cline{2-2}
      & Column g2c \bigstrut\\\cline{2-2}
      & Column g2d \bigstrut\\
\hline
\multirow{4}*{\minitab[c]{Common \\ g text}} & Column g2a\\
      & Column g2b \\
      & Column g2c \\
      & Column g2d \\
\hline
\end{tabular}
```

which will appear as :

| | |
|---|---|
| Common g text | Column g2a |
| | Column g2b |
| | Column g2c |
| | Column g2d |
| Common g text | Column g2a |
| | Column g2b |
| | Column g2c |
| Common g text | Column g2a |
| | Column g2b |
| | Column g2c |
| | Column g2d |
| Common<br>g text | Column g2a |
| | Column g2b |
| | Column g2c |
| | Column g2d |

If any of the spanned rows are unusually large, or if you're using the bigstrut and `\bigstruts` are used asymetrically about the centerline of the spanned rows, the vertical centering may not come out right. Use the fixup argument in this case.

Just before *text* is expanded, the `\multirowsetup` macro is expanded to set up any special environment. Initially, `\multirowsetup` contains just `\raggedright`. It may be redefined with `\renewcommand`.

It's just about impossible to deal correctly with descenders. The text will be set up centred, but it may then have a baseline that doesn't match the baseline of the stuff beside it, in particular if the stuff beside it has descenders and *text* does not. This may result in a small misalignment. About all that can be done is to do a final touchup on *text*, using the fixup optional argument. (Hint : If you use a measure like `.1ex`, there's a reasonable chance that the fixup will still be correct if you change the point size.)

`\multirow` is mainly designed for use with `table`, as opposed to `array`, environments. It will not work well in an array environment since the lines have an extra `jot` of space between them which it won't account for. Fixing this is difficult in general, and doesn't seem worth it. The *bigstruts* argument may be used to provide a semi-automatic fix : First set `\bigstrutjot` to `.5\jot`. Then simply repeat *nrows* as the bigstruts argument. This will be close, but probably not exact ; you can use the fixup argument to refine the result. (If you do this repeatedly, you'll probably want to wrap these steps up in a simple macro. Note that the modified `\bigstrutjot` value will not give reasonable results if you have bigstruts and use this argument for its intended purpose elsewhere. In that case, you might want to set it locally.)

If you use `\multirow` with the colortbl package you have to take precautions if you want to colour the column that has the `\multirow` in it. colortbl works by colouring each cell separately. So if you use `\multirow` with a positive *nrows* value, colortbl will first color the top cell, then `\multirow` will typeset *nrows* cells starting with this cell, and later colortbl will color the other cells, effectively hiding the text in that area. This can be solved by putting the `\multirow` in the last row with a negative *nrows* value. See, for example :

```
\begin{tabular}{l>{\columncolor{yellow}}l}
  aaaa & \\
  cccc & \\
  dddd & \multirow{-3}*{bbbb}\\
\end{tabular}
```

which will produce :

| | |
|---|---|
| aaaa | |
| cccc | bbbb |
| dddd | |

## 3 Using bigstrut

`\bigstrut[`*x*`]` produces a strut which is `\bigstrutjot` (`2pt` by default) higher, lower, or both than the standard array/table strut. Use it in table entries that are adjacent to `\hline`s to leave an extra bit of space — according to the TeXbook (page 246), "This is a little touch that improves the appearance of boxed tables ; look for it as a mark of quality."

Although you could use `\bigstrut` in an array, there isn't normally much point since arrays are 'opened up' by `\jot` anyway.

`\bigstrut[t]` adds height ; `\bigstrut[b]` adds depth. Just `\bigstrut` adds both. So : Use `\bigstrut[t]` in the row just *after* an `\hline` ; `\bigstrut[b]` in the row just *before* ; and `\bigstrut` if there are `\hline`s both before and after.

Spaces after the `\bigstrut` are ignored, even if it has an optional argument. Spaces before the `\bigstrut` are generally ignored (by a single).

Note : The multirow package makes use of `\bigstrutjot`. If both styles are used, they can be used in either order, as each checks to see if the other has already defined `\bigstrutjot`. However, the default values they set are different : if only multirow is used, `\bigstrutjot` will be set to `3pt`. If bigstrut is used, with or without multirow, `\bigstrutjot` will be `2pt`.

## 4 Using bigdelim

The package is for working in a `table` or `array` environment, in which the multirow packages is also used.

Syntax of use is

```
\ldelim({n}{width}[text]
\rdelim){n}{width}[text]
```

The commands are used in a column of a `tabular` or `array` ; they create a big parenthesis, brace or whatever delimiter that extends over the *n* rows starting at the one containing the command. Corresponding cells in the following rows must be explicitly given (as empty cells).

The first parameter is a delimiter to be used, e.g., `\{ \} [ ] ( )` — in fact, anything that can be used with `\left` or `\right`, as appropriate.

The optional *text* is set centred to the left of `\ldelim` or to the right of `\rdelim`. The *width* is that reserved for the delimiter and its text ; with a current copy of the multirow package, the *width* may be given as `*`, but that may cause the delimiters to be too small.

Also with a recent version of multirow the commands may be used in the last row of the extension with a negative *n* parameter. This is useful in combination with colortbl (see the discussion in section 2 on multirow). If there are unusually tall rows you may have to enlarge *n* (you can use non-integral values).