The multirow, bigstrut and bigdelim packages

Pieter van Oostrum*
Øystein Bache
Jerry Leichter†

Released 2021/03/15, version v2.8

Contents

1 Introduction 2

2 Changes in version 2 2

3 Using multirow 3
   3.1 Package Options 4
   3.2 Examples 5
   3.3 Fine-Tuning 8
   3.4 Multirow and colored cells 8
   3.5 Fine-tuning the ⟨bigstruts⟩ argument 10
   3.6 Use with longtable 11
   3.7 Use with supertabular 12
   3.8 Dealing with tall entries 12

4 Using bigstrut 16

5 Using bigdelim 17

6 Contact Information 19

7 Implementation 19
   7.1 The multirow package 19
   7.2 The bigstrut package 24
   7.3 The bigdelim package 25

A Appendix 25
   A.1 Case ⟨nrows⟩ > 0 26
   A.2 Case ⟨nrows⟩ < 0 28
   A.3 Overview 29

*catalogued “active author”
†Documentation originally put together by Robin Fairbairns
1 Introduction

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

\textbf{multirow} which provides a construction for table cells that span more than one row of the table;

\textbf{bigstrut} which creates struts which (slightly) stretch the table row in which they sit.

\textbf{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

2 Changes in version 2

\textbf{version 2.8}

- Optional argument \langle\texttt{vmove}\rangle for \texttt{\ldelim} and \texttt{\rdelim}.

\textbf{version 2.7}

- Make \texttt{@xmultirow} and \texttt{\multirow@setcolwidth \long} to allow multi-paragraph text. See \url{https://github.com/pietvo/multirow/issues/1}

\textbf{version 2.6}

- Make the \texttt{supertabular} option compatible with newer versions of the \texttt{supertabular} package

- Initialize \texttt{@arstrutbox} when not defined, to enable some uses of the big delims outside of an \texttt{array} or \texttt{tabular}.

\textbf{version 2.5a}

- Changed contact information

\textbf{version 2.5}

- Solve a clash with the \texttt{colortbl} package, when multirow uses the \texttt{longtable} option. There was a clash with both packages redefining \texttt{@cline}.

\textbf{version 2.4}

- Add a \texttt{\leavevmode} in \texttt{bigstrut} to force horizontal mode

- Make \langle\texttt{width}\rangle and \langle\texttt{vmove}\rangle in \texttt{\multirow calc} compatible

\textbf{version 2.3}

- Replaced \texttt{\texttt{textrm}} with \texttt{\texttt{textnormal}} in text beside big braces in bigdelim.sty.
version 2.2
• Support for fractional values of ⟨nrows⟩.

version 2.0
• \multirow now has an first optional parameter [⟨vpos⟩].
• The ⟨width⟩ parameter can be specified as = to use the defined width of the column in which the \multirow appears.
• Optional prefix letters (t, b) for the ⟨bigstruts⟩ parameter (see section 3.5).
• Package option debug.
• Package option longtable to work around a bug in longtable. See section 3.6.
• Package option supertabular to better support supertabular. See section 3.7.
• Better positioning in some cases.
• Lots of documentation.
• The distribution is now based on a .dtx file.
• Backwards compatible with v1.6.

3 Using multirow
\multirow \multirow sets a piece of text in a tabular or similar environment, spanning multiple rows. We will call the block of rows and columns that the text spans the multirow block. Usually this covers one column, but by combining it with \multicolumn more columns can be covered.

The basic syntax is:

\multirow[⟨vpos⟩]{⟨nrows⟩}⟨bigstruts⟩[⟨width⟩][⟨vmove⟩]{⟨text⟩}

where

⟨vpos⟩ defines the vertical positioning of the text in the multirow block. The default is [c] which means the text will be vertically centered. Other options are [t] for top alignment and [b] for bottom alignment.

⟨nrows⟩ is the number of rows to span. You should leave the other rows empty at this column, otherwise 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. Fractional values are permitted for ⟨nrows⟩; this allows for some fine-tuning.
\langle bigstruts \rangle is mainly used if you’ve used the bigstrut package. 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.

The \langle bigstruts \rangle parameter can optionally be preceded by a prefix letter t, b or tb for finer control. See section 3.5 for details. The letter may be separated from the number by a space character.

\langle width \rangle is the width to which the text is to be set. Special values are * to indicate that the text parameter’s natural width is to be used, and = to indicate that the specified width of the column in which the \multirow entry is set should be used.

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

\langle text \rangle 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 could use a \tabular or \array environment in the text parameter. See for example the \minitab below.

The width can also be given as = when the \multirow entry is given in a column that has a defined width, for example in a p(\{) column, an X column in \tabularx or a L, C, R or J column in a \tabulary environment. The text will be set in a \parbox of that width. If you give “=” in other situations, you will get strange results (usually a too wide column).

In \multirow version 2.4 and later, the \langle width \rangle and \langle vmove \rangle arguments can be given as calc expressions if the calc package is loaded. It is the responsibility of the document writer to include the calc package; \multirow does not do this.

N.B. \multirow can be used in the \tabular environment and most derivatives of it, for example \tabularx, \tabulary, \supertabular, \ltablex, \xtab, \longtable, \tabu, \booktabs and \ctable. For some of these you have to pay special attention to certain cases, see below.

\begin{verbatim}
\multirowsetup
\end{verbatim}

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

If you want to use both \multirow and \multicolumn on the same entry, you must put the \multirow inside the \multicolumn. The other way around will not work. For example:

\begin{verbatim}
\multicolumn{2}{c}{\multirow{3}{*}{Multi-multi}}
\end{verbatim}

3.1 Package Options

\begin{verbatim}
multirowdebugtrue
multirowdebugfalse
\end{verbatim}

The following options are defined:
**debug** This option causes information about multirow boxes to be written to the log file. This is done by the TeX \showbox command. Note: this will cause the \LaTeX\ compilation to be considered failed, even if there is no real error. This option can also be activated anywhere in the document with the command `\multirowdebugtrue` and deactivated with `\multirowdebugfalse`. When such a command is placed just before a `\multirow`, it applies only to that specific `\multirow` entry.

**longtable** The longtable option redefines the \cline macro to work around a bug in the longtable package. See section 3.6.

### 3.2 Examples

An example with both multirow and bigstrut):

```latex
\newcommand{\minitab}[2]{\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\\
& Column g2b \bigstrut\\
& Column g2c \bigstrut\\
\hline
\multirow{4}\[8\]*{1in}{Common g text, but a bit longer.} & Column g2a}\bigstrut\\
& Column g2b \bigstrut\\
& Column g2c \bigstrut\\
& 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:
All human beings are born free and equal in dignity and rights. & &rognośa, they shall be held in slavery or servitude; slavery and the
slave trade shall be prohibited in all their forms.

Everyone is entitled to all the rights and freedoms set forth in this Declaration, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status.

Everyone has the right to life, liberty and security of person. & no one shall be subjected to torture or to cruel, inhuman or degrading treatment or punishment.
A few observations about this example:

1. The middle column is the \texttt{\multirow}. You would expect it to be vertically centered, but it isn’t. This is because \texttt{\multirow} doesn’t know the height of the box. The only estimate \texttt{\multirow} can make about the height is the number of rows $\times$ the normal height of a row. It tries to center the text in that space, but that space is too low in this example. Therefore the text is at the top of the box. If you want it to be centered, you have to supply a \texttt{\vspace} argument to shift it down.

2. We have used an \texttt{\extrarowheight} of 2pt, to make a bit room between the \texttt{\hline} and the following text. However, this is not applied to the \texttt{\multirow}, because this is thought to be centered. In this case you can give the \texttt{\vpos} argument as [t], in which case \texttt{\multirow} will do the proper positioning.

Now with a negative \texttt{\nrows}.

\begin{tabulary}{11cm}{|L|L|L|}
\hline
All human beings are born free and equal in dignity and rights. & Everyone has the right to life, liberty and security of person. & \texttt{\multirow{-2}{=}[12mm]{Everyone is entitled to all the rights and freedoms set forth in this Declaration, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status.}} \\
\cline{1-1}\cline{3-3}
No one shall be held in slavery or servitude; slavery and the slave trade shall be prohibited in all their forms. & & No one shall be subjected to torture or to cruel, inhuman or degrading treatment or punishment. \\
\hline
\end{tabulary}

In this case the text would be centered somewhere in the bottom row, which would make it stick out of the bottom. Therefore we applied a \texttt{\vspace} of 12mm.

The \texttt{\vspace} usually requires some experimentation.

\footnote{This is only available with the \texttt{array} package, which \texttt{tabulary} includes automatically.}
3.3 Fine-Tuning

If any of the spanned rows are unusually large, or if you’re using the \bigstrut package and \bigstruts are used asymmetrically about the centerline of the spanned rows, the vertical centering may not come out right. Use the \langle vmove \rangle parameter in this case. Sometimes it may be more helpful to just use a larger value for \langle nrows \rangle, including fractional values. See an example in section 3.8.

It’s just about impossible to deal correctly with descenders. The text will be set up centered, 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 \langle text \rangle does not. This may result in a small misalignment. About all that can be done is to do a final touchup on \langle text \rangle, using the \langle vmove \rangle optional parameter. (Hint: If you use a measure like .1ex, there’s a reasonable chance that the \langle vmove \rangle will still be correct if you change the point size.)

\multirow is mainly designed for use with \texttt{tabular}, as opposed to \texttt{array}, environments. It might not work well in an array environment if there are big formulas in some rows; in that case you can use the \langle vmove \rangle parameter to refine the result.

In some cases you might want to align the multirow entry with the top of the other row cells, for example if you have a large capital in it. When you use \langle vpos \rangle = \{t\}, the baselines will be aligned, which is the wrong thing in this case. You can then do the positioning with the \langle vmove \rangle parameter and let \LaTeX calculate the amount. For example:

\usepackage{calc}
\newlength{\shiftdown}
\setlength{\shiftdown}{\heightof{\Huge\bfseries B}-\heightof{f}}
\begin{tabular}{cll}
\toprule
\multirow[t]{5}{*}{\Huge\bfseries B} & foo & Lorem ipsum dolor sit \\
& bar & Maecenas sed purus \\
& baz & Nullam luctus id tellus \\
& qux & Aenean consequat commodo \\
\bottomrule
\end{tabular}

\begin{tabular}{ccc}
B & foo & Lorem ipsum dolor sit \\
& bar & Maecenas sed purus \\
& baz & Nullam luctus id tellus \\
& qux & Aenean consequat commodo \\
\end{tabular}

In multirow version 2.4 and later you can also directly use the expression -\heightof{\Huge\bfseries B}-\heightof{f} instead of -\shiftdown for the \langle vmove \rangle argument.

3.4 Multirow and colored cells

If you use \texttt{\multirow} with the \texttt{colortbl} package (or the xcolor package with the [table] option) you have to take precautions if you want to color the column
that has the \texttt{\multirow} in it. The \texttt{colortbl} package works by coloring each cell separately. So if you use \texttt{\multirow} with a positive \langle \texttt{nrows} \rangle value, \texttt{colortbl} will first color the top cell, then \texttt{\multirow} will typeset \langle \texttt{nrows} \rangle cells starting with this cell, and later \texttt{colortbl} will color the other cells, effectively hiding the text in that area. This can be solved by putting the \texttt{\multirow} in the last row with a negative \langle \texttt{nrows} \rangle value. See, for example:

\begin{verbatim}
\begin{tabular}{|>{\columncolor{yellow}}l|}
  aaaa & \ \ \\
  cccc & \ \\
  dddd & \texttt{\multirow{-3}{*}{bbbbb}} \\
\end{tabular}
\end{verbatim}

which will produce:

\begin{tabular}{l}
  aaaa \\
  cccc \\
  dddd \texttt{bbbbb}
\end{tabular}

When you use colored multirow cells together with the \texttt{hhline} package you may find some white stripes in your colored multirow cell. For example:

\begin{verbatim}
\begin{tabular}{|c|>{\columncolor{red}}c|}
\hline
\bfseries ColumnOne & \bfseries ColumnTwo \\
\hline
First data & 932 \\
& 239 & \hline
& 137 & \hline
\texttt{\multirow{-3}{*}{More data}} & 319 & \hline
Last data & 132 \\
\hline
\end{tabular}
\end{verbatim}

This can be solved by putting colored horizontal rules with the same color in the colored multirow cell.

\begin{verbatim}
\begin{tabular}{|>{\columncolor{red}}c|>{\columncolor{red}}c|}
\hline
\bfseries ColumnOne & \bfseries ColumnTwo \\
\hline
First data & 932 \\
& 239 & \hline
& 137 & \hline
\texttt{\multirow{-3}{*}{More data}} & 319 & \hline
Last data & 132 \\
\hline
\end{tabular}
\end{verbatim}
3.5 Fine-tuning the \textit{\texttt{bigstruts}} argument

\texttt{\textbackslash multirow} can calculate the height of the required multirow box from \texttt{\textlangle\texttt{nrows}} and \texttt{\textlangle\texttt{bigstruts}}, supposed that all the rows don’t have “unusual” heights. However, there are cases when this is not enough to properly position the box, especially when there is a \texttt{\textbackslash bigstrut} on top of the first row and/or one on the bottom of the last row. In that case \texttt{\textbackslash multirow} should be given additional information. This is done by prefixing the \texttt{\textlangle\texttt{bigstruts}} argument with a letter (or two) indicating which of these two are present.

See the following examples:

\begin{tabular}{|c|c|}
\hline
\multirow{3}[1]{*}{Multirow} & T \bigstrut[t] \\
\cline{2-2}
& X \\
\cline{2-2}
& B \\
\hline
\end{tabular}

In the top box in the above example the text “Multirow” should be centered, but it is a bit below the center, because of the \texttt{\bigstrut[t]} in the top row. We can correct this by giving the \texttt{\textlangle\texttt{bigstruts}} parameter as “t 1", indicating a bigstrut in the top. This is done in the bottom box, where \texttt{\multirow[t]{3}[t 1]{*}{Multirow}} is used.

A second example:

\begin{tabular}{|c|c|}
\hline
\multirow{-3}[1]{*}{Multirow} & B \bigstrut[b] \\
\hline
\end{tabular}

In the top box the \texttt{\multirow[t]} should be positioned on the same height as the T, but it is too high, because there is a \texttt{\bigstrut} in the bottom. We
can correct that by specifying the \texttt{\bigstrut\texttt{t}} argument as “b 1”, i.e. using \texttt{\multirow[t]{-3}[b 1]{\texttt{\bigstrut\texttt{}}}}.

The possibilities for the prefix are:

\textbf{t} There is a bigstrut in the top, i.e. a \texttt{\bigstrut} or \texttt{\bigstrut[t]} in the top row.

\textbf{b} There is a bigstrut in the bottom, i.e. a \texttt{\bigstrut} or \texttt{\bigstrut[b]} in the bottom row.

\textbf{tb} They are both present. Note: this cannot be given as \texttt{bt}.

The space between the letter(s) and the number is optional. Please note that the prefix does not depend on whether the \texttt{\multirow} is in the top or the bottom row.

### 3.6 Use with \texttt{longtable}

It is possible to use \texttt{\multirow} in a \texttt{longtable} environment (as well as in its descendent \texttt{longtabu}). However, care must be taken that the longtable doesn’t break the multirow entry when it is near the bottom of the page. For example:

\begin{longtable}{|l|l|l|}
\hline
\ldots & \ldots & \ldots \\
Sept. 21 & 09:00 & event 1 \\
Sept. 22 & 10:00 & event 2 \\
Sept. 23 & 10:00 & event 3 \\
& 12:00 & event 4 \\
& 15:00 & event 5 \\
Sept. 24 & 09:00 & event 6 \\
\ldots & \ldots & \ldots \\
\hline
\end{longtable}

In this case if the “Sept. 23” entry comes close to the bottom of the page, you want to prevent the pagebreak to occur in the middle of this entry. You can do this by ending the intermediate rows with \texttt{\ldots} instead of \texttt{\ldots}.

\begin{longtable}{|l|l|l|}
\hline
\ldots & \ldots & \ldots \\
\multirow{3}*{Sept. 23} & 10:00 & event 3 \texttt{\ldots} & 12:00 & event 4 \texttt{\ldots} & 15:00 & event 5 \texttt{\ldots} \\
\cline{2-3}
\hline
\ldots & \ldots & \ldots \\
\end{longtable}

There is, however, a bug in \texttt{longtable}, that causes the \texttt{\ldots} not to work if it is followed by a \texttt{\cline}, like in the following example:

\begin{longtable}{|l|l|l|}
\hline
\ldots & \ldots & \ldots \\
\multirow{3}*{Sept. 23} & 10:00 & event 3 \texttt{\ldots} & 12:00 & event 4 \texttt{\ldots} & 15:00 & event 5 \texttt{\ldots} \\
\cline{2-3}
\cline{2-3}
\hline
\ldots & \ldots & \ldots \\
\end{longtable}
\multirow has a package option \texttt{longtable} that redefines \texttt{\cline} so that the \\
\* will also work when followed by \texttt{\cline}. The code comes from David Carlisle.

\section*{3.7 Use with \texttt{supertabular}}

With the package \texttt{supertabular} (or the augmented version \texttt{xtab}) there is the same requirement to keep the rows of a multirow together when a pagebreak occurs. Unfortunately, \texttt{supertabular} does not have a way to specify that a pagebreak should be suppressed. I.e. \texttt{\*} does not suppress a pagebreak. Therefore \texttt{multirow} provides a package option \texttt{supertabular} that redefines \texttt{\*} inside a \texttt{supertabular} to suppress the pagebreak. You should use this to end the intermediate rows in a multirow block. However, this does not cause \texttt{supertabular} to consider breaking the page before the \texttt{\multirow}, contrary to \texttt{longtable}. Thus the table may become too long.

Therefore when the \texttt{supertabular} option is given, \texttt{multirow} also provides a command \texttt{\STneed} to be used in a \texttt{supertabular} that specifies how much space we need on the page. Then if there is not enough space, a pagebreak will occur at that place. For example:

\begin{verbatim}
\tabletail{\hline}
\begin{supertabular}{|l|l|l|}
\ldots & \ldots & \ldots \\
\hline
Sept. 20 & 10:00 & event 1 \\
\hline
Sept. 21 & 10:00 & event 2 \\
\hline
Sept. 22 & 10:00 & event 3 \\
\hline
\STneed {2cm}
\multirow{3}*{Sept. 23} & 10:00 & event 4 \\
\cline{2-3} & 12:00 & event 5 \\
\cline{2-3} & 15:00 & event 6 \\
\hline
Sept. 24 & 09:00 & event 7 \\
\hline
\ldots & \ldots & \ldots \\
\end{supertabular}
\end{verbatim}

You will have to experiment a bit with the value to see what works. Sometimes it is better to exaggerate the required space a bit.

\section*{3.8 Dealing with tall entries}

Sometimes there are rows that are taller than what is expected. This section gives some hints how to deal with these situations. There are two cases:

1. When there is an exceptionally tall row outside of the multirow block the positioning of the \texttt{\multirow} might be wrong. This is because \texttt{\multirow} does not have information about the heights of the rows. This can happen for
example when a large formula is entered in a cell, or a multi-line paragraph in a \{\{} column. An example:

\begin{tabular}{| l | l | p{4cm} |}
\hline
\multirow{3}*{Week 38} & Monday & Rain most of the day\ \cline{2-3}
& Tuesday & Sunny with some clouds\ \cline{2-3}
& Wednesday & A clear day with a lot of sunshine. However, the strong wind will bring down the temperature. \ \hline
\end{tabular}

| Week 38 | Monday       | Rain most of the day          |
|---------|--------------|
|         | Tuesday      | Sunny with some clouds        |
|         | Wednesday    | A clear day with a lot of sunshine. However, the strong wind will bring down the temperature. |

The \texttt{\multirow} is positioned on the second row, because it specifies that it should cover 3 rows. However, the second row is not the vertical center in this case because the third row is much taller.

To remedy this, the \texttt{\langle vmove \rangle} parameter could be used. However, in this case it would be easier to pretend that \texttt{\multirow} spans 6 rows (the total number of lines in the last column). So use \texttt{\multirow{6}}... and we get:

\begin{tabular}{| p{2mm} l | p{5cm} |}
\hline
\multicolumn{2}{|l|}{\textbf{Medicine \\& dose}} & \textbf{Possible Side effects} \ \hline
\multicolumn{2}{|l|}{Spirino} & Confusion, hallucinations, rapid breathing, \ \hline
\end{tabular}

2. The second case is when the \texttt{\multirow} entry is taller than the surrounding normal rows. In that case the multirow text will stick out of its block. We must now enlarge the other rows, and that is something \texttt{\multirow} cannot do.

An example: (Don’t take this as a medical advice. The names are fake anyway.)

\begin{tabular}{| p{2mm} l | p{5cm} |}
\hline
\multicolumn{2}{|l|}{} & \textbf{Possible Side effects} \ \hline
\multicolumn{2}{|l|}{} & \textbf{Confusion, hallucinations, rapid breathing,} \ \hline
\end{tabular}

\begin{tabular}{| p{2mm} p{2mm} p{2mm} l |}
\hline
\hline
\multicolumn{2}{|l|}{} & \textbf{Possible Side effects} \ \hline
\end{tabular}

13
<table>
<thead>
<tr>
<th>Medicine &amp; dose</th>
<th>Possible Side effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spirino</td>
<td>Confusion, hallucinations, rapid breathing, seizure (convulsions); upset stomach, heartburn; severe nausea, vomiting, or stomach pain or mild headache.</td>
</tr>
<tr>
<td>initial: 200 mg/day</td>
<td></td>
</tr>
<tr>
<td>maintenance: 100-400 mg/day</td>
<td></td>
</tr>
<tr>
<td>Conzac</td>
<td>Anxiety; nervousness; insomnia; anorexia; mild bradycardia; SA node slowing; weight loss; solar photosensitivity; hyponatremia; sexual dysfunction (both genders); may alter glycemic control in diabetic patients.</td>
</tr>
<tr>
<td>initial: 10 mg/day</td>
<td></td>
</tr>
<tr>
<td>maintenance: 10-40 mg/day</td>
<td></td>
</tr>
</tbody>
</table>

Both `\multirow` entries are too high; the first sticks out into the second entry, and the second one sticks out of the table.

There are two ways we can correct this: The simplest would be to add extra empty rows to cover the overlapping space. For the first entry that would be 2 extra rows; for the second 4. So we add twice `& &` before the third `\hline`, and four of these before the last `\hline`. This gives us just the correct table:
<table>
<thead>
<tr>
<th>Medicine &amp; dose</th>
<th>Possible Side effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spirino</td>
<td>Confusion, hallucinations, rapid breathing, seizure (convulsions); upset stomach, heartburn; severe nausea, vomiting, or stomach pain or mild headache.</td>
</tr>
<tr>
<td>initial: 200 mg/day</td>
<td></td>
</tr>
<tr>
<td>maintenance: 100-400 mg/day</td>
<td></td>
</tr>
<tr>
<td>Conzac</td>
<td>Anxiety; nervousness; insomnia; anorexia; mild bradycardia; SA node slowing; weight loss; solar photosensitivity; hyponatremia; sexual dysfunction (both genders); may alter glycemic control in diabetic patients.</td>
</tr>
<tr>
<td>initial: 10 mg/day</td>
<td></td>
</tr>
<tr>
<td>maintenance: 10-40 mg/day</td>
<td></td>
</tr>
</tbody>
</table>

The second way is to stretch the normal rows vertically, such that they fit with the multirow entry. In this table, where the font size is 10pt, each row has a total height of 12pt. For the first entry we need 24pt extra (2 rows). Because this space must be divided over 3 rows that is 8pt per row, making the total height of the row 20pt. The normal row has a height of 8.4pt and a depth of 3.6pt (total 12pt). We can add 4pt on the top and 4pt on the bottom, or any other combination that adds up to 8pt. In this case I have chosen to make the height 12pt and the depth 8pt. We do this with a \rule with 0 width. \newcommand{\mystrut}{\rule[-8pt]{0pt}{20pt}} and put \mystrut in each of the first 3 rows. By defining your own struts you have complete control over the layout. You can choose to give some rows more space than others, or to put all the space in the last row, for example.

For the second entry we need 48pt extra (4 rows). We will use \bigstrut in each row, that is 16pt per row, and as a \bigstrut is 2\bigstrutjots, we set \bigstrutjot to 8pt. The booktabs package adds some extra vertical space around the rules, therefore when using the normal tabular environment, it is probably better to make the struts a little bit bigger, or a bit smaller with booktabs. After some experimentation it appeared that a \bigstrutjot of 7pt was enough. Of course we added the \langle bigstruts \rangle argument of \[tb6\] to the second multirow. Please note that this is not possible with our own struts, unless we cheat.

Now with booktabs the code becomes:

```latex
\newcommand{\mystrut}{\rule[-8pt]{0pt}{20pt}}
\setlength{\bigstrutjot}{7pt}
\begin{tabular}{ p{2mm} l p{5cm} }
\hline
\multicolumn{2}{l}{\textbf{Medicine \\
& dose}} & \textbf{Possible Side effects} \\
\cmidrule(r){1-2} \cmidrule(l){3-3}
\multicolumn{2}{l}{Spirino} \mystrut & Confusion, hallucinations, rapid breathing, seizure (convulsions); upset stomach, heartburn; severe nausea, vomiting, or stomach pain or mild headache. \\
\cmidrule(r){1-2}
```

15
<table>
<thead>
<tr>
<th>Medicine &amp; dose</th>
<th>Possible Side effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spirino</td>
<td>Confusion, hallucinations, rapid breathing, seizure (convulsions); upset stomach, heartburn; severe nausea, vomiting, or stomach pain or mild headache.</td>
</tr>
<tr>
<td>initial: 200 mg/day</td>
<td></td>
</tr>
<tr>
<td>maintenance: 100-400 mg/day</td>
<td></td>
</tr>
<tr>
<td>Conzac</td>
<td>Anxiety; nervousness; insomnia; anorexia; mild bradycardia; SA node slowing; weight loss; solar photosensitivity; hyponatremia; sexual dysfunction (both genders); may alter glycemic control in diabetic patients.</td>
</tr>
<tr>
<td>initial: 10 mg/day</td>
<td></td>
</tr>
<tr>
<td>maintenance: 10-40 mg/day</td>
<td></td>
</tr>
</tbody>
</table>

4 Using bigstrut

\bigstrut \bigstrut[(x)] produces a strut (a rule with width 0) which is \bigstrutjot (2pt by default) higher, lower, or both than the standard array/tabular strut. Use it in table entries that are adjacent to \hline 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 \hlines 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 \unskip).

\bigstrutjot Note: The multirow package makes use of \bigstrutjot. If both packages 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 \texttt{multirow} is used, \bigstrutjot will be set to 3pt. If \texttt{bigstrut} is used, with or without \texttt{multirow}, \bigstrutjot will be 2pt.

5 Using \texttt{bigdelim}

The package is for working in a \texttt{tabular} or \texttt{array} environment, in which the \texttt{multirow} package is also used.

\begin{equation}
\begin{array}{ccccccc}
\ldelim{(4)}{4mm} & x & x & x & x & \rdelim){4}{4mm} & \\
\ldelim{} & x & x & x & x & & i \\
\ldelim{} & x & x & x & x & & j \\
\ldelim{} & x & x & x & x & & \\
\ldelim{} & k & u & v & & \\
\end{array}
\end{equation}

The commands are used in a column of a \texttt{tabular} or \texttt{array}; they create a big parenthesis, brace or whatever delimiter that extends over the \langle n \rangle 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 \texttt{\left} or \texttt{\right}, as appropriate.

Here is an example:

\begin{verbatim}
\begin{tabular}{p{2em}l}
\ldelim\{{3}{*}[type] & dvi \\
  & \texttt{\left}\texttt{\left}\texttt{\left}\texttt{\left}dvi \\
  & \texttt{\right}\texttt{\right}\texttt{\right}\texttt{\right}dvi \\
  & \texttt{\left}\texttt{\left}\texttt{\left}\texttt{\left}ps \\
  & \texttt{\right}\texttt{\right}\texttt{\right}\texttt{\right}ps \\
  & \texttt{\left}\texttt{\left}\texttt{\left}\texttt{\left}pdf \\
  & \texttt{\right}\texttt{\right}\texttt{\right}\texttt{\right}pdf \\
\end{tabular}
\end{verbatim}

The optional parameter \langle vmove \rangle is a length used for fine-tuning: the delimiter (with the optional \langle text \rangle) will be raised (or lowered, if \langle vmove \rangle is negative) by that length above (below) wherever it would otherwise have gone. This is just like with \texttt{\multirow}, but note that here the \langle vmove \rangle goes before the \langle width \rangle.

When \texttt{\ldelim} is used, the optional \langle text \rangle is set centred to the left of \texttt{\ldelim}. If \texttt{\rdelim} is used it is set to the right of \texttt{\rdelim}. The \langle width \rangle parameter is the space that is reserved for the delimiter and its text; as with the \texttt{multirow} package, the \langle width \rangle may be given as *. Compare for example these:

\begin{verbatim}
\begin{tabular}{p{2em}l}
\ldelim\{{3}{*}[type] & dvi \\
  & \texttt{\left}\texttt{\left}\texttt{\left}\texttt{\left}dvi \\
  & \texttt{\right}\texttt{\right}\texttt{\right}\texttt{\right}dvi \\
  & \texttt{\left}\texttt{\left}\texttt{\left}\texttt{\left}ps \\
  & \texttt{\right}\texttt{\right}\texttt{\right}\texttt{\right}ps \\
  & \texttt{\left}\texttt{\left}\texttt{\left}\texttt{\left}pdf \\
  & \texttt{\right}\texttt{\right}\texttt{\right}\texttt{\right}pdf \\
\end{tabular}
\end{verbatim}

\end{document}
In the first example we cheated: by using a column width that is too small, we swallowed up some of the intercolumn space, at the cost of an “Overfull \hbox” message. In the second example we did it the proper way by inserting @{} to replace the default intercolumn space with a narrow space.

Also the commands may be used in the last row of the extension with a negative \langle n \rangle argument. This is useful in combination with the \texttt{colortbl} or \texttt{xcolor} packages (see the discussion in section 3 on \texttt{multirow}). If there are unusually tall rows you may have to enlarge \langle n \rangle (you can use fractional values). If you have horizontal lines that interact with the braces you are advised to use the \texttt{hhline} package to make the lines.

If you decrease or eliminate the intercolumn space with a\{} and use colored backgrounds with the \texttt{colortbl} or \texttt{xcolor} packages (commands \texttt{columncolor}, \texttt{rowcolor}, \texttt{rowcolors} or \texttt{cellcolor}), you will notice that part of the brace will be cut off. In reality it will be overwritten with the color of the next cell. See this example:

\begin{tabular}{ c @{}c c c }
\rowcolors{2}{green!25}{green!75}
\begin{tabular}{|l|l|l|}
\begin{tabular}{|l|l|l|}
\hline
1 & 2 & 3 \\
4 & 5 & 6 \\
\hline
\end{tabular}
\end{tabular}
\end{tabular}

This is not a problem of \texttt{multirow} or \texttt{bigdelim}; it will also happen if there is normal text in the column before the a\{}. The reason is that these color commands extend the color to cover the intercolumn spaces on both sides to prevent gaps in the color. The size of these so-called overhangs is \texttt{tabcolsep} (or \texttt{arraycolsep} when an \texttt{array} is used) on each side. However, when a\{} is used there is no such intercolumn space and the extension covers parts of the previous column. This can be cured by setting the left overhang explicitly to 0pt with a \texttt{columncolor} command in the tabular header, like >{\texttt{\columncolor\[white\]0pt}}[\texttt{tabcolsep}]. Unfortunately the explicit color \texttt{white}, removes the transparency of the column, so if there are cells in that column that have no explicit color, these cells are affected. If the background of the \texttt{tabular} is white, this normally will not be noticed, but if the background color is changed, for example with the \texttt{pagecolor} command, then that color should be used rather than \texttt{white}. Unfortunately, there is no command to specify the overhangs without also specifying a color.

In the following example we have done this. In order to keep the table header compact, we put the definition in a \texttt{newcolumn} command (using the \texttt{array} package).
In case you want to have a paragraph type text as optional parameter you could put it in a `\parbox`. Alternatively you could add an extra column with the text in a `\multirow`, like in

```
\begin{tabular}{l@{}l@{}l}
dvi & \rdelim\){3}{1em} & \multirow{3}{4cm}{These are the output types, that are commonly used for \TeX.} \\
ps & & \\
pdf & & \\
\end{tabular}
```

These are the output types, that are commonly used for \TeX.

Note that we used `{@{}}` to eliminate the intercolumn space to get the text tight to the brace.

## 6 Contact Information

Pieter van Oostrum  
email: pieter@vanoostrum.org  
www: http://pieter.vanoostrum.org

The source code can be found on Github:  
https://github.com/pietvo/multirow  
Bugs can be reported at  
https://github.com/pietvo/multirow/issues

## 7 Implementation

### 7.1 The multirow package

This is a boolean to [de]activate debugging (showing the generated box contents). It is activated by the `debug` package option. The `\newif` initializes it to false.

```
1 \newif\multirowdebug
2 \DeclareOption{debug}{\multirowdebugtrue}
```

The package option `longtable` redefines the `\cline` macro to work around a bug in `longtable`. See section 3.6². First we check if the macro `\CT@arc` is defined. If

²Thanks to David Carlisle. See his answer on stackexchange.
so, this indicates that the \colortbl package is loaded. As \colortbl also redefines \@cline, we must take this into account with our own redefinition of \@cline.

\begin{verbatim}
\DeclareOption{longtable}{%
  \AtBeginDocument{%
    \@ifundefined{CT@arc}{\def\@cline#1-#2\@nil{%
      \omit
      \@multicnt#1%
      \advance\@multispan\m@ne
      \ifnum\@multicnt=\@ne\@firstofone{&\omit}\fi
      \@multicnt#2%
      \advance\@multicnt-#1%
      \advance\@multispan\@ne
      \leaders\hrule\@height\arrayrulewidth\hfill
      \cr
    \noalign{\nobreak\vskip-\arrayrulewidth}}}
  \def\@cline#1-#2\@nil{%
    \omit
    \@multicnt#1%
    \advance\@multispan\m@ne
    \ifnum\@multicnt=\@ne\@firstofone{&\omit}\fi
    \@multicnt#2%
    \advance\@multicnt-#1%
    \advance\@multispan\@ne
    {\CT@arc@\leaders\hrule\@height\arrayrulewidth\hfill}%
    \cr
  \noalign{\nobreak\vskip-\arrayrulewidth}}
  \}}
\end{verbatim}

The package option \texttt{supertabular} redefines \texttt{\textbackslash*} inside a \texttt{supertabular}. The redefinition is delayed until the \texttt{\begin{document}}.

\texttt{Supertabular} version 4.1f and later need a call to \texttt{\ST@save@lineno} to function properly, but earlier versions cannot use this as it doesn’t exist in these versions. So the definitions of both \texttt{\ST@tabularcr} and \texttt{\MRST@cr} are different depending on whether \texttt{\ST@save@lineno} is defined. There are also some other differences, so some intermediate versions of \texttt{supertabular} might need more subtle adaptations, but for now we leave it at that.

\begin{verbatim}
\DeclareOption{supertabular}{%
  \AtBeginDocument{%
  \ST@tabularcr
  This macro is the definition of \textbackslash* inside a \texttt{supertabular}. We check for a \texttt{*}, and if it is present we call our own version, otherwise the \texttt{supertabular} version. First we get the older version for pre-4.1f \texttt{supertabular}, then the newer version.

  \ifx\ST@save@lineno\undefined
    \def\ST@tabularcr{%
      \ifnum0='\fi
      \@ifstar{\MRST@xtabularcr}{\ST@xtabularcr}}
  \else
    \MRST@cr
    \MRST@cr
    is a truncated copy of \texttt{\ST@cr}. It does all the bookkeeping about the space the \texttt{longtable} occupies, but it doesn’t do the pagebreaking part.

  \def\MRST@cr{%
\end{verbatim}

20
These are the newer versions.

\def\ST@tabularcr{%
  \ifnum0='\fi
  \ST@save@lineno
  \ifstar{\MRST@xtabularcr}{\ST@xtabularcr}
\}

\def\MRST@xtabularcr{%
  \ifnum0='\fi
  \ifdim \ST@pboxht > \z@ \unskip \MRST@xargarraycr{\ST@pboxht} \else \MRST@yargarraycr{\ST@pboxht} \fi
\}

\def\MRST@xargarraycr#1{%
  \@tempdima #1 \advance \@tempdima \dp \@arstrutbox \vrule \@height \z@ \@depth \@tempdima \@width \z@ \cr
  \noalign{\global \ST@toadd=#1} \MRST@cr
}

\def\MRST@yargarraycr#1{%
  \ifnum0='\fi
  \ifdim #1 > \z@ \unskip \MRST@xargarraycr{#1} \else \MRST@yargarraycr{#1} \fi
\}

\def\MRST@xtabularc{\MRST@xtabularcr}
\def\MRST@argtabularc{\MRST@xargtabularcr}
\def\MRST@yargtabularc{\MRST@yargtabularcr}

These are copies of the corresponding macros from supertabular, but instead of \ST@cr they call \MRST@cr.
This macro can be used in a supertabular to indicate how much space a multirow entry needs. See section 3.7.

\multirow@colwidth is a length that is used to implement the “=” variant of \langle width \rangle.

\multirow@colwidth is a length that is used to implement the “=” variant of \langle width \rangle.

\multirow@setcolwidth This macro calculates \multirow@colwidth for an entry that has the \langle width \rangle given as “=”. We check if we are inside a tabulary environment, by checking if \TY@final is defined. If not, then \multirow@colwidth = \hsize. The tabulary environment will make two passes. On the first pass, we set \multirow@colwidth to the size that the text would have in LR mode (with newlines replaced by spaces), so that tabulary will gives us enough space. On the second pass (characterized by \TY@box = \TY@box@v) we use the value that tabulary has given us in \hsize. This algorithm is not perfect, but good enough in most cases.

\multirow@setcolwidth This macro calculates \multirow@colwidth for an entry that has the \langle width \rangle given as “=”. We check if we are inside a tabulary environment, by checking if \TY@final is defined. If not, then \multirow@colwidth = \hsize. The tabulary environment will make two passes. On the first pass, we set \multirow@colwidth to the size that the text would have in LR mode (with newlines replaced by spaces), so that tabulary will gives us enough space. On the second pass (characterized by \TY@box = \TY@box@v) we use the value that tabulary has given us in \hsize. This algorithm is not perfect, but good enough in most cases.

\multirowsetup is executed at the beginning of each \multirow.

\multirowsetup is executed at the beginning of each \multirow.

\multirow@vbox This creates the \vbox. Parameters:
#1 = \langle vpos \rangle, #2 = initialization code (for example to set the width of the \parbox), #3 = box contents.

Depending on the \langle vpos \rangle parameter, it will be top-aligned, vertically centered, or bottom-aligned. This is done by inserting \vfill in the proper places. Note: the \relax is to protect against an empty \langle vpos \rangle argument.
\texttt{\multirow} Make an entry that will span multiple rows of a table. First collect all the arguments and replace missing optional arguments by their default values.

\begin{verbatim}
106 \% \texttt{\multirow} \{vpos\} \{nrows\} \{bigstruts\} \{width\} \{vmove\} \{text\}
107 \newcommand\multirow\{2\}[c]\{\@multirow\[#1\]{#2}\}
108 \def\@multirow\[#1\]#2{\@ifnextchar[\{\@@multirow\[#1\]#2}{\@@multirow\[#1\]#2[0]}}
109 \def\@@multirow\[#1\]#2[#3]#4{\@ifnextchar[\{\@xmultirow\[#1\]{#2}[#3]{#4}}%
110 \{\@xmultirow\[#1\]{#2}[#3]{#4}[0pt]\}}
\end{verbatim}

This macro splits off a \texttt{t}, \texttt{b}, or \texttt{tb} prefix of the \langle \texttt{bigstruts} \rangle argument, and sets \texttt{\multirow@cntb} to the numerical value. The prefix is remembered in two booleans: \texttt{\multirow@prefixt} and \texttt{\multirow@prefixb}.

\begin{verbatim}
111 \newif\ifmultirow@prefixt
112 \newif\ifmultirow@prefixb
113 \def\multirow@piii#1#2#3\end{\multirow@prefixtfalse\multirow@prefixbfalse
114 \if t#1\multirow@prefixttrue
115 \if b#2\multirow@prefixbtrue \multirow@cntb=#3%
116 \else \multirow@cntb=#2#3%
117 \fi
118 \else
119 \if b#1\multirow@prefixbtrue \multirow@cntb=#2#3%
120 \else \multirow@cntb=#1#2#3%
121 \fi
122 \fi}
\end{verbatim}

This is the real workhorse. It starts with splitting the \langle \texttt{bigstruts} \rangle argument, and then calculating the height of the \texttt{\multirow} box. Because \langle \texttt{nrows} \rangle \ (#2) can be fractional, we cannot use \texttt{\ifnum} to test for positive or negative. Therefore we use \texttt{\ifdim} by putting a unit (pt) after the number.

\begin{verbatim}
123 \long\def\@xmultirow\[#1\]#2[#3]#4[#5]#6{\expandafter\multirow@piii#3\relax\end%
124 \setlength\multirow@dima{#2\ht\@arstrutbox}%
125 \addtolength\multirow@dima{#2\dp\@arstrutbox}%
126 \ifdim#2pt<\z@ \setlength\multirow@dima{-\multirow@dima}fi
127 \addtolength\multirow@dima{\multirow@cntb\bigstrutjot}%
\end{verbatim}

The text is set in a \texttt{\vbox} by calling \texttt{\multirow@vbox}. If the \langle \texttt{width} \rangle argument is * set just the text in the \texttt{\vbox}.

\begin{verbatim}
129 \if*#4\multirow@vbox{#1}{}\hbox{\strut#6\strut}\par
\end{verbatim}

Otherwise set it in a \texttt{\parbox} inside a \texttt{\vbox}. If the \langle \texttt{width} \rangle argument is given as "\texttt{=}", we calculate \texttt{\multirow@colwidth} and use that as width of the \texttt{\parbox}.

\begin{verbatim}
130 \else \if#4\multirow@setcolwidth#6\%
131 \multirow@vbox{#1}\{\setlength\hspace{\multirow@colwidth}\parboxrestore\{\strut#6\strut\par\}
\end{verbatim}

Otherwise the given argument is used as the width of the \texttt{\parbox}.

\begin{verbatim}
132 \else \multirow@vbox{#1}\{\setlength\hspace{#4}\parboxrestore\{\strut#6\strut\par%}
133 \fi \fi
\end{verbatim}

Now position the \texttt{\vbox} properly. More details are given in the appendix. The overview of the calculation of the shift amount can be found in section A.3.

If \langle \texttt{nrows} \rangle > 0:
If \langle \texttt{vpos} \rangle = [t], then the box is already positioned correctly (the baseline is on
the baseline of the row). However, later the top of the box will be taken as the reference point (instead of the baseline), therefore we take the height of the box (h) as the shift amount. See fig. 1.

If \(vpos = [c]\) we shift it up \(h_1\) (see fig. 2), where \(h_1 = \text{ht} \@arstrutbox + (\text{bigstrutjot} \text{ifmultirow}@\text{prefixt})\).

If \(vpos = [b]\) we shift it up \(h_1 + h_2\) (see fig. 3), where \(h_2 = \text{dp} \@arstrutbox + (\text{bigstrutjot} \text{ifmultirow}@\text{prefixb})\).

We calculate the required shift in \text{multirow}@dima.

\begin{verbatim}
\textbf{134} \text{ifdim}#2pt>\z@ \text{\if#1t\relax\setlength\multirow@dima{\ht0}\else}
\text{\setlength\multirow@dima{\ht\@arstrutbox}\%}
\text{\ifmultirow@prefixt \addtolength\multirow@dima{\bigstrutjot}\fi}
\text{\if#1b\relax \addtolength\multirow@dima{\dp\@arstrutbox}\%}
\text{\ifmultirow@prefixb \addtolength\multirow@dima{\bigstrutjot}\fi}
\text{\fi}
\text{\fi}
\textbf{151}\fi
\end{verbatim}

If \(\langle vpos \rangle < 0:\)
If \(\langle vpos \rangle = [t]\), shift the box up \(H - h_1 - h_2 + h\). See fig. 4.
If \(\langle vpos \rangle = [c]\), shift the box up \(H - h_2\). See fig. 5.
If \(\langle vpos \rangle = [b]\), shift the box up \(H\). See fig. 6.

\(H\) is the current value of \text{multirow}@dima.

Finally, we add the \(\langle \text{vmove} \rangle\) argument (\#5), and go into horizontal mode. Then we shift the box up by putting a \texttt{\vskip} above it, and add it to the output. Because of the \texttt{\vskip} the resulting box will have a height 0. We set the depth of the \texttt{\vbox} to 0, so that it will not influence the depth of the current row.

If \text{multirowdebug} is true, we show the box.

\begin{verbatim}
\textbf{152} \addtolength\multirow@dima{\#5}\%
\text{\leavevmode
\setbox0\vtop{\vskip-\multirow@dima\box0\vss}\dp0=\z@\}
\text{\setlength{\vboxdepth}{5} \setlength{\vboxbrehth}{10} \vbox{0}
\vbox0}
\textbf{157}
\end{verbatim}

\texttt{\bigstrutjot} Define \texttt{\bigstrutjot} if not already defined.

\begin{verbatim}
\textbf{158}\texttt{\ifdefined(\bigstrutjot)\{\newdimen\bigstrutjot \bigstrutjot=\jot\}\}}
\end{verbatim}

\subsection{7.2 The \texttt{bigstrut} package}

\texttt{\bigstrutjot} This is a length. By default it is set to 2pt. You can change it with the \texttt{\setlength} command.

\begin{verbatim}
\textbf{159}\texttt{\ifdefined(\bigstrutjot)\{\newdimen\bigstrutjot\} \texttt{\bigstrutjot=2pt}
\end{verbatim}
\bigstrut This macro inserts a strut. Depending on the optional parameter it extends above and/or below the standard array/tabular strut.

\newcommand{\bigstrut}{\leavevmode\unskip\@tempdima=\ht\@arstrutbox \@tempdimb=\dp\@arstrutbox \ifx #1b\relax \else \advance\@tempdima by \bigstrutjot\fi \ifx #1t\relax \else \advance\@tempdimb by \bigstrutjot\fi \hbox{\vrule \@height\@tempdima \@depth\@tempdimb \@width\z@} \ignorespaces}

7.3 The bigdelim package

\ldelim This macro typesets a left delimiter. It calls \multirow with the proper arguments.

The size of the delimiter is determined by putting a \vbox with the proper height and zero width next to it. The height is the one that \multirow already has calculated in \@arstrutbox. That calculation uses the size of \@arstrutbox, which is set by tabular or array environments. In case it is not set, we initialize it to a default value.

\newcommand{\ldelim}{\@ifnextchar{[}{\@ldelim{#1}{#2}}{\@ldelim{#1}{#2}[0pt]}}
\def{\@ldelim#1#2[#3]#4}{\@ifnextchar{[}{\@@ldelim{#1}{#2}{#3}{#4}}{\@@ldelim{#1}{#2}{#3}{#4}[\null]}}
\def{\@@ldelim#1#2#3#4[#5]}{\ifvoid\@arstrutbox\setbox\@arstrutbox\hbox{\strut}\fi \multirow{#2}{#4}{[#3]}{\ensuremath{\left.\vcenter{\hsize=0pt\vrule \@height\multirow@dima \@width\z@}\textnormal{#5}\right.}}}

\rdelim This macro typesets a right delimiter. It calls \multirow with the proper arguments, similar to \ldelim.

\newcommand{\rdelim}{\@ifnextchar{[}{\@rdelim{#1}{#2}}{\@rdelim{#1}{#2}[0pt]}}
\def{\@rdelim#1#2[#3]#4}{\@ifnextchar{[}{\@@rdelim{#1}{#2}{#3}{#4}}{\@@rdelim{#1}{#2}{#3}{#4}[\null]}}
\def{\@@rdelim#1#2#3#4[#5]}{\ifvoid\@arstrutbox\setbox\@arstrutbox\hbox{\strut}\fi \multirow[#2]{#4}{#3}{\ensuremath{\left#1\vcenter{\hsize=0pt\vrule \@height\multirow@dima \@width\z@}\textnormal{#5}\right.}}}

A Appendix

In this section we explain the \vbox positioning in \multirow. The positioning depends on the \(\langle\text{nrows}\rangle\), \(\langle\text{vpos}\rangle\), \(\langle\text{bigstruts}\rangle\) and \(\langle\text{vmove}\rangle\) arguments. The box is constructed with \vtop. The algorithm of \vtop is described in The \TeX\book, p. 81.

Each case is described by a figure. In the figure the lefthand column indicates the context of the tabular in which the multirow appears, i.e \(\langle\text{nrows}\rangle\) rows. The righthand column is the multirow box that is to be inserted. The baseline is the natural position where the material will be positioned in the first place. Later it will be shifted up to the desired location.

H is the calculated height of the box: \(\langle\text{nrows}\rangle\) × the natural height of a row + \langle\text{bigstruts}\rangle × \bigstrutjot.
topstrut = \bigstrutjot if there is a \bigstrut on the top of the first row (as indicated by the t prefix in the \langle bigstruts \rangle argument), otherwise 0.
botstrut = \bigstrutjot if there is a \bigstrut on bottom of the last row (as indicated by the b prefix in the \langle bigstruts \rangle argument), otherwise 0.
h1 = height of a tabular row + topstrut
h2 = depth of a tabular row + botstrut

Note: the following descriptions describe the vertical shift of the box without taking the \langle vmove \rangle into account. In all cases \langle vmove \rangle has to be added if it is given.

A.1 Case \langle nrows \rangle > 0
\langle vpos \rangle = [t]
In this case the \vbox contains the text followed by a \vfill. Such a \vbox has a height that is the height of the top line of the text (h). H = height + depth of the box. This means that the box is already positioned correctly. However, later we will put the box inside another \vbox, with a \vskip on to of it, and this will make the top of the box its reference point. Therefore we will have to shift it up again over a distance h (which probably will be different from the height of the tabular row). So the total shift becomes h. See fig. 1.

Alternatively, we could have omitted the \vskip in this case, thereby leaving the baseline undisturbed, but this would make the code unsymmetrical. Moreover, this would not work when a non-zero \langle vmove \rangle is present. Therefore we choose to set the shift amount to h here.

A.1 Case \langle nrows \rangle > 0
\langle vpos \rangle = [c]
In this case the \vbox contains a \vfill, the text, and another \vfill. Such a \vbox has a height 0, i.e. the top of the box is on the baseline. Because both boxes have the same size (H), they can be aligned by shifting the \vbox up over h1. See fig. 2.

\begin{figure}
\centering
\includegraphics[width=0.8\textwidth]{figure1}
\caption{Case \langle nrows \rangle > 0, \langle vpos \rangle = [t]}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=0.8\textwidth]{figure2}
\caption{Case \langle nrows \rangle > 0, \langle vpos \rangle = [c]}
\end{figure}
\( \langle vpos \rangle = [b] \)

Now the \texttt{\vbox} contains a \texttt{\vfill}, followed by the text. Because it ends with the text, it gets an additional depth equal to the depth of the last line of the text. Such a \texttt{\vbox} has a height 0, i.e. the top of the box is on the baseline, but its depth is \( H + \) that depth. In other words the baseline of the last text line is \( H \) below the top.

Because \( \langle vpos \rangle = [b] \) we want the baseline of the last text line to shift to the baseline of the last tabular row. The amount of the shift is \( h1 + h2 \). See fig. 3.

\( \langle vpos \rangle = [c] \)

Figure 2: Case \( \langle nrows \rangle > 0, \langle vpos \rangle = [c] \)

Figure 3: Case \( \langle nrows \rangle > 0, \langle vpos \rangle = [b] \)
A.2 Case $\langle n\text{rows} \rangle < 0$

$\langle n\text{rows} \rangle < 0$ when the multirow is positioned in the last row of the multirow block.

$\langle v\text{pos} \rangle = [t]$

In this case the $\vbox$ contains the text followed by a $\vfill$. Such a $\vbox$ has a height that is the height of the top line of the text. The baseline is aligned with the baseline of the last row. Because $\langle v\text{pos} \rangle = [t]$, we want it to be aligned with the baseline of the first row. Therefore it has to be shifted up $H - h1 - h2$. But because later the height of the box will be set to 0, we must also add the current height $h$. Therefore the total shift becomes $H - h1 - h2 + h$. See fig. 4.

$\langle v\text{pos} \rangle = [c]$

In this case the $\vbox$ contains a $\vfill$, the text, and another $\vfill$. Such a $\vbox$ has a height 0, i.e. the top of the box is on the baseline. Because both boxes have the same size ($H$), they can be aligned by shifting the $\vbox$ up over $H - h2$. See fig. 5.

$\langle v\text{pos} \rangle = [b]$

The $\vbox$ contains a $\vfill$, followed by the text. Because it ends with the text, it gets an additional depth equal to the depth of the last line of the text. Such a $\vbox$ has a height 0, i.e. the top of the box is on the baseline, but its depth is $H$ + that depth. In other words the baseline of the last text line is $H$ below the top.
Because $\langle vpos \rangle = [b]$ we want the baseline of the last textline to shift to the baseline of the last tabular row. The amount of the shift is $H$. See fig. 6.

### A.3 Overview

<table>
<thead>
<tr>
<th>$\langle vpos \rangle$</th>
<th>$\langle nrows \rangle &gt; 0$</th>
<th>$\langle nrows \rangle &lt; 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[t]</td>
<td>$h$</td>
<td>$H - h1 - h2 + h$</td>
</tr>
<tr>
<td>[c]</td>
<td>$h1$</td>
<td>$H - h2$</td>
</tr>
<tr>
<td>[b]</td>
<td>$h1 + h2$</td>
<td>$H$</td>
</tr>
<tr>
<td>$x$</td>
<td>$H - h1 - h2 + x$</td>
<td></td>
</tr>
</tbody>
</table>

### Change History

**bigdelim v0.0**
- General: bigbrace.sty by Øystein Bache
  
**bigdelim v1.0**
- General: Initial version
  
**bigdelim v2.3**
- \$\l delim: Replace $\text{\texttrm{xtm}}$ by $\text{\textnormal{}xtm}$\$ 25
- \$\r delim: Replace $\text{\texttrm{xtm}}$ by $\text{\textnormal{}xtm}$\$ 25

**bigdelim v2.6**
- \$\l delim: Initialize $\@arstrutbox$ if not defined\$ 25
- \$\r delim: Initialize $\@arstrutbox$ if not defined\$ 25

**bigdelim v2.8**
- \$\l delim: Add optional argument $\langle vmove \rangle$\$ 25
- \$\r delim: Add optional argument $\langle vmove \rangle$\$ 25

29
Figure 6: Case $\langle \text{nrows} \rangle < 0, \langle \text{vpos} \rangle = [b]$

bigstrut v1.0
General: Initial version 24

bigstrut v2.4
General: Add $\backslash$leavevmode at the beginning to force horizontal mode 25

multirow v1.0
General: distributed anonymously, based on a Usenet posting 19

multirow v1.1
General: allow it to work without bigstrut.sty (Pieter van Oostrum) 19

multirow v1.2
General: modified by Jerry Leichter for the same goal, but using a different approach which will work properly with bigstrut.sty 19

multirow v1.2a
General: modified by Pieter van Oostrum to use \vskip instead of \raise in positioning, avoiding making rows too high when the adjustment is large 19

multirow v1.3
General: modified by Pieter van Oostrum to work properly in a $p$ column ($\backslash$leavevmode added) 19

multirow v1.4
General: modified by Pieter van Oostrum to check for the special case that the width is given as an *. In this case the natural width of the text argument will be used and the argument is processed in LR-mode. 19

multirow v1.5
General: modified by Pieter van Oostrum: Added a $\%$ after $\hbox{#5}\backslash vfill$. Added $\struts$ around #5 for better vertical positioning. Additional coding for negative value of $\langle \text{nrows} \rangle$. 19

multirow v1.6
General: modified by Pieter van Oostrum: Replace a space by
\relax after
\advance\multirow@dima#4 . . 19

v1.7
General: Give all the files the same
version number . . . . . . . . . . . . 1

v1.8
\multirow: Add the optional first
parameter (vpos) . . . . . . . . . . . 23

v1.9
General: Give \multirow its own
temp registers, so that we can
safely pass the box height to
bigdelim . . . . . . . . . . . . . . . . . 22
Implement the “a” option for
\multirow’s (width) parameter. . . . . 22

v1.9a
\multirow: Add the optional prefix
to the (bigstruts) parameter . . 23
Redo the \vbox calculation and
positioning. . . . . . . . . . . . . . . . 23
General: Implement the debug
option. . . . . . . . . . . . . . . . . . 19

v1.9b
General: Implement the longtable
option. . . . . . . . . . . . . . . . . . 19
Implement the supertabular
option and the \STneed
command. . . . . . . . . . . . . . . . . . 20

v2.0
General: Release v2.0 . . . . . . . 1

v2.1
\multirow: Set depth of final
\vbox to 0, to prevent a tall
multirow line to push the
following rows downwards. . . . . 24
General: Rename (fixup) to
(vmove) in the documentation
as in The LaTeX Companion. . . 3

v2.2
\multirow: Support fractional
values for (nrows). . . . . . . . . . . 23
General: Eliminate
\multirow@cnta . . . . . . . . . . . . 22

v2.3
General: Small bugfix . . . . . . . 1

v2.4
\multirow: Support calc
compatible expressions for
(width) and (vmove). . . . . 23
General: Add in bigstrut.sty . . 1
Make (width) and (vmove) in
\multirow calc compatible . . . . . 1

v2.5
General: Make the redefinition of
\cline compatible with the
colortbl package . . . . . . . . . . . . 19
Solve a clash with the colortbl
package . . . . . . . . . . . . . . . . . . . . 1

v2.6
General: Adapt the definition to be
compatible with modern
versions of supertabular . . . . . . 20

v2.7
\multirow: Make \@multirow
\long to allow multi-paragraph
text . . . . . . . . . . . . . . . . . . . . 23
\multirow@setcolwidth: Make
\multirow@setcolwidth \long
to allow multi-paragraph text 22

Index

Numbers written in italic refer to the page where the corresponding entry is
described; numbers underlined refer to the code line of the definition; numbers in
roman refer to the code lines where the entry is used.

\begin{tabular}{|lll|}
\hline
B & E & L \\
\bigstrut & \extrarowheight & \delim \\
\bigstrutjot & 158, 159 & \longtable \\
booktabs & 15 & \longtabu \\
& & \hline
\hline
C & I & M \\
\cellcolor & 18 & \MRST@argtabularc \\
\cline & 3 & \ifmultirow@prefixb \\
colortbl & 8, 18 & \ifmultirow@prefixt \\
\columncolor & 18 & \ifmultirow@debug \\
\hline
\end{tabular}