The **arraycols** package*

Antoine Missier
antoine.missier@ac-toulouse.fr
June 26, 2019

1 Introduction

This package provides new predefined column types to typeset tables in addition to the **array** package by Frank Mittelbach and David Carlisle [1] (loaded by **arraycols**) and also a command to draw wide horizontal rules. Here is a summary of the column types and macro, defined by **arraycols**, which we detail in the next section.

<table>
<thead>
<tr>
<th>Column definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>R</td>
</tr>
<tr>
<td>t{⟨width⟩}</td>
</tr>
<tr>
<td>x</td>
</tr>
<tr>
<td>y</td>
</tr>
<tr>
<td>z{⟨width⟩}</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>Z</td>
</tr>
<tr>
<td>I</td>
</tr>
<tr>
<td>V{⟨thickness⟩}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Horizontal rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>\whline</td>
</tr>
</tbody>
</table>

Note that if a column type has already been defined previously, it will be overwritten with a warning message.

---

*This document corresponds to **arraycols** v1.0, initial version dated 2019/06/26. Thanks to François Bastouil for English translation.
Besides `array`, `arraycols` loads the `cellspace` package, necessary for the x, y, z and Z types of columns and `tabularx` [3], necessary for T and Z, as well as `makecell` [4] for various alignments of multilined table cells.

This package, with a very short code, does not claim to develop new macros. Its main action is to combine and set features coming from other packages.

2 Usage

Referring to an example from the `array` package documentation, `arraycols` provides the L, C, R type of columns which reverse the mathematical mode. Then we can use these column types to get centred, left-aligned or right-aligned LR-mode in an `array` or math-mode in a `tabular` environment. For instance, declaration 
\begin{tabular}{|l|C|r|}...
produces the second column in a centred mathematical mode and declaration \begin{array}{|L|c|c|}...
produces the first column in text mode, left aligned. We add the definition \texttt{t\{⟨width⟩\}} (text in LR-mode) producing a horizontal and a vertical centering in the column unlike the classics \texttt{p\{⟨width⟩\}} and \texttt{m\{⟨width⟩\}} which produce left aligned text.

To ensure a sufficient height for rows, for instance for formulas in displaymath mode, we have defined the column types x (centred) and y (left aligned), based on the `cellspace` package by Josselin Noirel [2]. They allows an automatic adjustment of the rows heights to avoid touching the horizontal rules when content is too high. Although `cellspace` is defined \textit{a priori} for `tabular` environments, the new x and y column types, defined by `arraycols`, produce a column in mathematical mode with the right adjustment, either with a `tabular` or with an `array` environment.

Look at the following examples produced with \begin{array}{|c|}...
and with \begin{array}{|x|}...
bad
\begin{align*}
\lim_{x\to 1} \log_b \left( \frac{x^2}{x-1} \right) \\
\int_1^x \frac{1}{t} \, dt
\end{align*}
good
\begin{align*}
\lim_{x\to 1} \log_b \left( \frac{x^2}{x-1} \right) \\
\int_1^x \frac{1}{t} \, dt
\end{align*}

The `cellspace` package is loaded with the `math` option\textsuperscript{2} for a good management of rows heights in matrix tables.

\textsuperscript{1}The declarations L, C, R do not work in a `tabularx` environment. Note that the `tabulary` package by David Carlisle [5] already defines the L, C, R, J column types for particular alignments in tables of same type as `tabularx`, but there is no incompatibility because these column definitions only apply in `tabulary` environments.

\textsuperscript{2}The `math` option loads the `amsmath` package. As mentionned in the `cellspace` package documentation: “the `amsmath` package can be loaded beforehand with other packages (such as `empheq` or `mathtools`), were an incompatibility to arise from one’s loading it later”.
Notice that another package, `booktabs` [6], also provides an excellent adjustment for rows heights, but unfortunately, it doesn’t handle the height of vertical separators. To get the same vertical adjustment as `booktabs`, we set the `cellspace` parameters:
\begin{verbatim}
\setlength{\cellspacetoplimit}{3pt},
\setlength{\cellspacebottomlimit}{2pt}.
\end{verbatim}

We should also mention the `tabls` package by Donald Arneseau [7] making a good adjustment of rows heights as well, but it’s incompatible with the `array` and `numprint` packages.

At last, it is also possible to make manual adjustments with the \`vstrut\` command from the `spacingtricks` package [9], or \`gape\` and \`Gape\` from the `makecell` package [4], or \`bigstrut\` from the `bigstrut` package [8].

The column definition `z{⟨width⟩}` enables, just as `t{⟨width⟩}`, to set the column width, but also activates the math mode and adjusts the row height, as `x`.

The `tabularx` package by David Carlisle [3] provides the `X` column definition whose width is calculated according to the required width for the whole table, and with left alignment as for `p{⟨width⟩}`. \begin{verbatim}
\begin{tabularx}{8cm}{|c|X|X|}
\end{tabularx}
\end{verbatim}
adjusts the width of the `X` columns for a total width of the table equals to 8 cm. As a complement, we propose the `T` declaration, doing the same thing but with horizontal centering and `Z` which furthermore activates the mathematical mode and adjusts the height of the line (as `x` or `z`). Here is an example with \begin{verbatim}
\begin{tabularx}{\linewidth}{|T|y|x|Z|T|}
\end{tabularx}.
\end{verbatim}

To keep the perfect alignment of fraction bars in mathematical formulas, cells are not vertically centred, however, to get a proper vertical positioning in the last cell, we have used the powerful \`makecell\` command of the `makecell` package by Olga Lapko [4]: \begin{verbatim}
\makecell{a multiline \ piece of text}.
\end{verbatim}

The column definition `I` is suggested in The \LaTeX\ Companion [10] and enables to draw a thick vertical line (1 pt thick) instead of the one obtained with standard declaration `|`. To choose thickness, we propose further column definition `V{⟨thickness⟩}`\footnote{The definition of `V` would have been simplified by using an optional argument for `I` but this way out is not working.}.

Likewise, the `\whline` command, proposed in The \LaTeX\ Companion, enables to draw a thick horizontal line (1 pt thick) instead of the one obtained with `\hline` and the `makecell` package provides further command `\Xhline{⟨thickness⟩}` enabling to choose the thickness of the horizontal rule.

The introduction table has been typeset with a column declaration `I` as separator between the two columns of text, and with `\whline` for the horizontal rules at
the begin and at the end of the table, and \Xhline{0.8pt} for the one following
the legend rows. Formatting header lines has been done with \thead command
from the makecell package. For this command \arraycolsep sets by default:
\renewcommand\theadfont{\footnotesize\sffamily}
(originally \footnotesize only, without \sffamily). At last, according to a rec-
commendation of the array package [1], 1 pt has been added to the normal height of
every row of the table, with the command \setlength{\extrarowheight}{1pt}.

3 Implementation

\begin{verbatim}
1 \RequirePackage{array}
2 \RequirePackage{math}{cellspace}
3 \RequirePackage{tabularx} % must be loaded after cellspace
4 \RequirePackage{makecell}

5 \newcolumntype{C}{>{$}c<{$}}
6 \newcolumntype{L}{>{$}l<{$}}
7 \newcolumntype{R}{>{$}r<{$}}
8 \newcolumntype{t}{>{\centering\arraybackslash}m{#1}}

9 \newcolumntype{x}{>{$}Sc<{$}}
10 \newcolumntype{y}{>{$}Sl<{$}}
11 \newcolumntype{z}{>{\centering\arraybackslash}p{#1}<{$}}
12 \newcolumntype{T}{>{\centering\arraybackslash}X}
13 \newcolumntype{Z}{>{$}ST<{$}}

The cellspace package provides the S modifier enabling, when placed before a
column declaration, to adjust the height of the content of the cells to avoid to
 touch horizontal rules. Spacing between the content and the rules is controlled by
the parameters \cellspacetoplimit and \cellspacebottomlimit.
14 \setlength{\cellspacetoplimit}{3pt}
15 \setlength{\cellspacebottomlimit}{2pt}
16 \newcolumntype{z}{|c}{>{\centering\arraybackslash}m{#1}|}
17 \newcolumntype{I}{|c}{>{\centering\arraybackslash}X}\
18 \newcolumntype{V}{|c}{>{\centering\arraybackslash}X}\
19 \newcolumntype{W}{|c}{>{\centering\arraybackslash}X}\
20 \newcolumntype{H}{|c}{>{\centering\arraybackslash}X}\
21 \newcolumntype{J}{|c}{>{\centering\arraybackslash}X}\
22 \newcolumntype{K}{|c}{>{\centering\arraybackslash}X}\
23 \newcolumntype{L}{|c}{>{\centering\arraybackslash}X}\
24 \newcolumntype{M}{|c}{>{\centering\arraybackslash}X}\
25 \newcolumntype{N}{|c}{>{\centering\arraybackslash}X}\
26 \newcolumntype{O}{|c}{>{\centering\arraybackslash}X}\
27 \newcolumntype{P}{|c}{>{\centering\arraybackslash}X}\
28 \newcolumntype{Q}{|c}{>{\centering\arraybackslash}X}\
29 \newcolumntype{R}{|c}{>{\centering\arraybackslash}X}\
30 \newcolumntype{S}{|c}{>{\centering\arraybackslash}X}\
31 \newcolumntype{T}{|c}{>{\centering\arraybackslash}X}\
32 \newcolumntype{U}{|c}{>{\centering\arraybackslash}X}\
33 \newcolumntype{V}{|c}{>{\centering\arraybackslash}X}\
34 \newcolumntype{W}{|c}{>{\centering\arraybackslash}X}\
35 \newcolumntype{X}{|c}{>{\centering\arraybackslash}X}\
36 \newcolumntype{Y}{|c}{>{\centering\arraybackslash}X}\
37 \newcolumntype{Z}{|c}{>{\centering\arraybackslash}X}\

For the z definition of column, we use p and not m (which automatically centers)
in order to keep a correct alignment for mathematical expressions in the cells of a
same row.

18 \newcolumntype{T}{|c}{>{\centering\arraybackslash}X}\
19 \newcolumntype{Z}{|c}{>{\centering\arraybackslash}X}\

The T columns are not automatically centred. It would be possible to do it with
the command \renewcommand{\tabularxcolumn}{|c}{>{\centering\arraybackslash}m{#1}|} (with m instead of
default value p), but unfortunately this has a global effect for all the declarations
of columns based on X, so T but also Z, and this would lead to disturb alignment
of mathematical expressions in the cells of a same row.

17 \newcolumntype{T}{|c}{>{\centering\arraybackslash}X}\
16 \newcolumntype{Z}{|c}{>{\centering\arraybackslash}X}\

\end{verbatim}
References


[3] The tabularx package, David Carlisle, CTAN, v2.11.b 2016/02/03.

[4] The makecell package, Olga Lapko, CTAN, v0.1e 2009/08/03.


[8] The multirow, bigstrut and bigdelim packages, Piet van Oostrum, Øystein Bache, Jerry Leichter, CTAN, v2.4 2019/01/01.
