The `equationarray` environment

Roland Winkler

`roland.winkler@physik.uni-regensburg.de`

March 11, 1996

Abstract

The `equationarray` combines the `eqnarray` and the `array` environment.

1 Introduction

Problem: The `eqnarray` environment is too restrictive because it is only prepared to accept three part equations. Some applications require more sophisticated mathematics, so three parts aren't enough. Simultaneously, we would like to have our equations numbered. If the equations aren't to be numbered, then the `array` environment suffices (nevertheless it is often easier to use the `equationarray*` environment than both an `equation` plus an `array` environment). If the equations aren't to be aligned with each other, then the `equation` environment is preferred.

The following is a new version of Tony Li's `equationarray` environment modified such that it is compatible with Frank Mittelbach's `array` environment, i.e. it should be possible to use all options of the `array` environment. If you find a bug (see below!) or if you make any improvements, I'd like to hear about them. When writing the `equationarray` environment I used the version v2.1b of the `array` environment. The original version of `equationarray` was written by Tony Li, University of Southern California, tli@sargas.usc.edu starting 6/15/88.

I know that my documentation of the `equationarray` environment is rather short – but maybe nobody else might like to use it and even this work was superfluous. So if you think that it would be useful to have a better documentation please let me know.

2 Example

We give a short example for the use of `equationarray`. The text is

```
\newcolumntype{e}{@{}}
\begin{equationarray}{rclel}
\phi & = & \sum \bigg( & \frac{xxxxxxxxx}{\phi}
\nonumber\\*[3ex]
```

```
& & & + \frac{yyyyyyyyyy}{\phi} \bigg) \\[4ex]
& = & \multicolumn{2}{l}{ \frac{zzzzzzzzzz}{\phi} }
\end{equationarray}
```

and we obtain (usually my formulas are much longer . . .)

$$\phi \;=\; \sum \left(\frac{xxxxxxxxxx}{\phi} \right.$$

$$\left. + \frac{yyyyyyyyyy}{\phi} \right) \tag{1}$$

$$=\; \frac{zzzzzzzzzz}{\phi} \tag{2}$$

One can avoid pagebreaks between two lines by using the star-version *. There is also the star-version equationarray* which has usually no linenumbers. But if for a particular line you still want a linenumber, you can use \yesnumber, (I find this easier than many lines with \nonumber). With the options [l] or [r] the equationarray will appear flushleft or flushright, with the options [c] the equationarray will be centered, e.g. \begin{equationarray}[l]{rclll}. The default will be centering without the fleqn option and left-justified, indented by \mathindent with the fleqn option. Thanks to Piet van Oostrum <piet@cs.ruu.nl> who added the code which is necessary for the fleqn option.

3   Bugs

The equationarray has a slightly modified version of \multicolumn because we must count the \spaned columns. Thus if you have \multicolumn within an array within the equationarray environment, then you might run into difficulties if you have a line with less &'s than columns defined in the preamble. Thus you can either fill up the end of a half empty line with extra &'s or you can put the original definition of \multicolumn within the definition of the array command.

4   The code

```
\typeout{equationarray \fileversion\space<\filedate>}
\typeout{English documentation\space\space<\docdate>}
```

Process the fleqn option.

```
\def\eqnarr@left{\@centering}
\let\eqnarr@opts\relax
\DeclareOption{fleqn}{
  \def\eqnarr@left{\mathindent}
  \def\eqnarr@opts{\displaywidth\linewidth
```

```
                              \advance\displaywidth-\mathindent} }
                \ProcessOptions
```

\equationarray
```
\def\equationarray{%
 \col@sep\arraycolsep
 \def\d@llarbegin{$\displaystyle}%
 \def\d@llarend{$}%
 \stepcounter{equation}%
 \let\@currentlabel=\theequation
 \set@eqnsw \global\@eqcnt\z@ \global\@eqargcnt\z@
 \let\@classz\@eqnclassz
 \def\multicolumn##1##2##3{\@eqnmulticolumn{##1}{##2}{##3}%
                              \global\advance\@eqcnt##1
                              \global\advance\@eqcnt\m@ne}%
 \def\@haligno{to\displaywidth}%
 \@ifnextchar[{\@equationarray}{\@equationarray[.]}}
```

\@eqnmulticolumn   \@eqnmulticolumn equals the original version of \multicolumn.
```
\let\@eqnmulticolumn=\multicolumn
```

\nonumber,\yesnumber   Note, that \nonumber is already defined in standard latex.tex
```
% \def\nonumber{\global\@eqnswfalse}
\def\yesnumber{\global\@eqnswtrue}
\let\set@eqnsw=\yesnumber
```

\@amper   We need a macro for & that expands at the right time.
```
\def\@amper{&}
```

\@eqargcnt   We must count the number of columns defined in the preamble so that we can
fill every line with exactly \@eqargcnt copies of & before we insert the equation
number.
```
\newcount\@eqargcnt   % counts number of columns
```

\@equationarray   The definition of \@equationarray follows the TeXbook, Exercise 22.9
```
\def\@equationarray[#1]#2{%
    \eqnarr@opts
    \@tempdima \ht \strutbox
    \advance \@tempdima by\extrarowheight
    \setbox\@arstrutbox=\hbox{\vrule
            \@height\arraystretch \@tempdima
            \@depth\arraystretch \dp \strutbox
            \@width\z@}%
    \gdef\advance@eqargcnt{\global\advance\@eqargcnt\@ne}%
    \begingroup
    \@mkpream{#2}%
    \xdef\@preamble{%
     \if #1l\tabskip\z@ \else\if #1r\tabskip\@centering
                        \else\if #1c\tabskip\@centering
```

3

```
                              \else\tabskip\eqnarr@left \fi\fi\fi
        \halign \@halignto
        \bgroup \tabskip\z@ \@arstrut \@preamble
        \if #1l\tabskip\@centering \else\if #1r\tabskip\z@
                              \else\tabskip\@centering \fi\fi
```

Here we need an extra column for the equation-number

```
          \@amper\llap{\@sharp}\tabskip\z@\cr}%
        \endgroup
        \gdef\advance@eqargcnt{}%
        \bgroup
        \let\@sharp## \let\protect\relax
        \m@th    \let\\=\@equationcr
        \let\par\@empty
        $$                             % $$ BRACE MATCHING HACK
        \lineskip \z@
        \baselineskip \z@
        \@preamble}
```

`\@eqnclassz` `\@eqnclassz` does the same thing as `\@classz` except that we add
`\advance@eqargcnt`

```
      \def\@eqnclassz{\@classx
        \@tempcnta \count@
        \advance@eqargcnt
        \prepnext@tok
        \@addtopreamble{%
          \global\advance\@eqcnt\@ne
          \ifcase \@chnum
          \hfil \d@llarbegin \insert@column \d@llarend\hfil \or
          \d@llarbegin \insert@column \d@llarend \hfil \or
          \hfil\kern\z@ \d@llarbegin \insert@column \d@llarend \or
          $\vcenter
          \@startpbox{\@nextchar}\insert@column \@endpbox $\or
          \vtop \@startpbox{\@nextchar}\insert@column \@endpbox \or
          \vbox \@startpbox{\@nextchar}\insert@column \@endpbox
          \fi}\prepnext@tok}
```

`\endequationarray`

```
        \def\endequationarray{\@zequationcr
          \egroup
          \global\advance\c@equation\m@ne $$  % $$ BRACE MATCHING HACK
          \egroup\global\@ignoretrue
          \gdef\@preamble{}}
```

`\@equationcr` If we have `\\*` the command `\@equationcr` avoids pagebreaks

```
        \def\@equationcr{${\ifnum0=`}\fi\@ifstar{\global\@eqpen\@M
            \@xequationcr}{\global\@eqpen\interdisplaylinepenalty
                            \@xequationcr}}
```

4

\@xequationcr

```
\def\@xequationcr{%
    \@ifnextchar[{\@argequationcr}{\ifnum0=`{\fi}${}%
    \@zequationcr}}
```

\@argequationcr

```
\def\@argequationcr[#1]{\ifnum0=`{\fi}${}\ifdim #1>\z@
    \@xargequationcr{#1}\else
    \@yargequationcr{#1}\fi}
```

\@xargequationcr

```
\def\@xargequationcr#1{\unskip
    \@tempdima #1\advance\@tempdima \dp \@arstrutbox
    \vrule \@depth\@tempdima \@width\z@
    \@zequationcr\noalign{\penalty\@eqpen}}
```

\@yargequationcr

```
\def\@yargequationcr#1{%
    \@zequationcr\noalign{\penalty\@eqpen\vskip #1}}
```

We add `&\omit` for those columns that will remain empty. Note that without `\omit` we already have `\advance\@eqcnt\@ne` in the preamble.

```
\def\@zequationcr{\@whilenum\@eqcnt <\@eqargcnt
    \do{\@amper\omit\global\advance\@eqcnt\@ne}%
```

We add an extra alignment tab for the equationnumber

```
\@amper
\if@eqnsw\@eqnnum\stepcounter{equation}\fi
\set@eqnsw\global\@eqcnt\z@\cr}
```

\equationarray*   Finally we define the `equationarray*` environment. It does exactly the same thing as `\equationarray` except that we `\let` the command `\set@eqnsw` equal `\nonumber`

```
\@namedef{equationarray*}{%
    \let\set@eqnsw=\nonumber \equationarray}
\@namedef{endequationarray*}{\endequationarray}
```