LaTeX package eqnalign

— — making eqnarray(*) look and work like align(*) — —

by Tom Hejda, tohecz@gmail.com

1 The goal

The goal of this package is to allow easy conversion from the insanely-looking eqnarray to the look and behaviour of align from amsmath. It is inspired by a TeX.StackExchange question http://tex.stackexchange.com/q/96210/11002 by a user called "Werner", and by an answer of mine to the question.

2 The behaviour

The package is activated by simply loading it, and it does not have any package options. It just redefines eqnarray and eqnarray*, and then it makes amsthm aware of this redefinition so that \qedhere works inside these environments.

3 License and remarks

(1) The package is licensed under the LaTeX Project Public License version 1.3c (LPPL v1.3c) or higher. The latest version of this license is in http://www.latex-project.org/lppl.txt.

(2) All bugs shall be reported to the GitHub page http://github.com/tohecz/eqnalign.

(3) Note that actually, the usage of this package is discouraged, in favour of converting the code into proper "amsmath code", using the true align and align* environments. It is intended for cases where a lot of already existing code needs to be converted and there is no capacity for doing it the right but time-consuming way.

4 Implementation

```
1 ⟨*package⟩
```

Package header.

```
2 \ProvidesPackage{eqnalign}[2016/06/09 v1.0 Make eqnarray(*) behave like align(*).]
```

The only necessary package is amsmath so that align and align* are defined.

```
3 \RequirePackage{amsmath}
```

The package does some catcode mysteries that shouldn't propagate out, so we make everything in a group and use \gdef everywhere.

```
4 \begingroup
```

\eqna@origamp We store a catcode-4 (tab alignment char) & in a macro. We need a catcode-13 (active) & througout the rest of the package.

```
5 \catcode`\&=4
6 \gdef\eqna@tab@amp{&}
7 \catcode`\&=13
```

\eqna@newamp This will be the replacement of & inside eqnalign. We use \eqna@doamp if the innermost environment is eqnalign and \eqna@origamp otherwise; this is to allow things like arrays and matrices inside eqnalign.

```
\gdef\eqna@newamp{%
   \ifx\@currenvir\eqna@currenvir
      \eqna@doamp
   \else
      \eqna@tab@amp
   \fi
}
```

\eqna@amp@i
\eqna@amp@ii
\eqna@amp@iii Three macros that are "rotated", after the first, the second shall be used, then the third. The third one ends a group since it ends a table cell, therefore after that the first one is again in action. The first & on a line is kept, the second is ignored, the third is kept.

```
\gdef\eqna@amp@i{\eqna@origamp\let\eqna@doamp\eqna@amp@ii}
\gdef\eqna@amp@ii{\let\eqna@doamp\eqna@amp@iii}
\gdef\eqna@amp@iii{\eqna@origamp}
```

\eqna@doamp The default is \eqna@amp@i.

```
\global\let\eqna@doamp\eqna@amp@i
```

\eqna@hook The default meaning of & is \eqna@amp@i. We store the current environment, which is either eqnarray or eqnarray*; it is used in \eqna@newamp for the test for inner environments. Then we activate & and make its meaning \eqna@newamp.

```
\gdef\eqna@hook{%
   \let\eqna@currenvir\@currenvir
   \catcode`\&=\active
   \let&\eqna@newamp
}
```

Now we will be defining environments containing * in name, so we make it a letter.

```
\catcode`\*=11
```

\eqna@def@env
eqnarray
eqnarray* We define a macro \eqna@def@env that contains the redefinitions of eqnarray (and eqnarray*). The environments themselves are just like align, just hooked using \eqna@hook. We then call this macro immediately to define the environments. (All this "double way" is to correct things in case hyperref is loaded later.

```
\gdef\eqna@def@env{%
   \gdef\eqnarray{\eqna@hook\align}%
   \gdef\eqnarray*{\eqna@hook\align*}%
   \global\let\endeqnarray\endalign
   \global\let\endeqnarray*\endalign*
}
\eqna@def@env
```

\eqnarray@qed To make \qedhere work in eqnarray, we need to "hint" amsthm that the two environments exist.
\eqnarray*@qed

```
\global\let\eqnarray@qed\align@qed
\global\let\eqnarray*@qed\align*@qed
```

End the group we began at the very beginning.

```
\endgroup
```

EQNarray Just of sentiment, we keep the original eqnarray as EQNarray.

```
\def\EQNarray{%
   \stepcounter{equation}%
   \def\@currentlabel{\p@equation\theequation}%
   \global\@eqnswtrue
   \m@th
   \global\@eqcnt\z@
   \tabskip\@centering
```

```
    \let\\\@eqncr
    $$\everycr{}\halign to\displaywidth\bgroup
        \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnsel
       &\global\@eqcnt\@ne\hskip \tw@\arraycolsep \hfil${##}$\hfil
       &\global\@eqcnt\tw@ \hskip \tw@\arraycolsep
          $\displaystyle{##}$\hfil\tabskip\@centering
       &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
          \tabskip\z@skip
       \cr
}
\def\endEQNarray{%
       \@@eqncr
       \egroup
       \global\advance\c@equation\m@ne
    $$\@ignoretrue
}
\@namedef{EQNarray*}{\def\@eqncr{\nonumber\@seqncr}\EQNarray}
\@namedef{endEQNarray*}{\nonumber\endEQNarray}
```

Last but not least, if `hyperref` is loaded after `eqnalign` (and only in that case), we issue a warning since `hyperref` is doing bad things to `eqnarray`, and we redefine `eqnalign` once more.

```
\@ifpackageloaded{hyperref}{}{
    \AtBeginDocument{
       \@ifpackageloaded{hyperref}{
          \@latex@warning{Package ‘eqnalign’ should be loaded after ‘hyperref’.\MessageBreak
          Redefining ‘eqnarray’ and ‘eqnarray*’ at this point \MessageBreak and crossing fin
          \eqna@def@env
       }{}
    }
}
```

That's all.

```
\endinput
```

⟨/package⟩