The PracTEX Journal, 2006, No. 4
Article revision 2006/10/16

Avoid eqnarray!

Lars Madsen

Email daleif@imf.au.dk
Website http://home.imf.au.dk/daleif/
Address Department of Mathematical Sciences
Faculty of Science, University of Aarhus
Denmark

Abstract Whenever the eqnarray environment appears in a question or an example of a problem on comp.text.tex or the TEXhax mailing list there is a large chance that someone will tell the poster not to use eqnarray. This article will provide some examples of why many of us consider eqnarray to be harmful and why it should not be used.

Introduction

Whenever someone on comp.text.tex asks a question about the eqnarray environment or shows an example using it there will always be someone that instructs the poster to stop using eqnarray and use something better instead. This article provides an example based overview of some of the reasons why a lot of people consider eqnarray to be obsolete. This article can then be used as a reference when a poster asks for an explanation.

The prerequisites for this article are a basic knowledge of LATEX and knowledge of the syntax used by eqnarray. Experience with the environments from the amsmath package is a plus but is not necessary.

1 The basics

In plain LaTeX, the `eqnarray` environment is basically the only construction available for numbered multi-line equations. The `eqnarray` environment is similar to

```
\begin{array}{rcl}
  ...
\end{array}
```

The difference being that the first and last cell in each row is automatically typeset as display style mathematics, and not as text style math like it would have in the `array` environment, and `eqnarray`, also supports automatic equations numbers.

The principal `eqnarray` usage is similar to this example:

```
\begin{eqnarray}
  y &=& (x+1)^2 \\  &=& x^2+2x+1
\end{eqnarray}
```

Which results in (without the box):

$$y \;=\; (x+1)^2 \tag{1}$$
$$=\; x^2+2x+1 \tag{2}$$

In the examples that follow, we use the command `\dbx` instead of writing some meaningless arbitrary mathematical formula. `\dbx` is a simple macro, defined by the author, that writes a box to simulate some random mathematical material. Using an optional argument we can adjust the width of the box created by `\dbx`.

The reason for using simulated math instead of actually writing something is that removing the actual text makes the reader more aware of the actual problem, which is not the text but rather the construction/surroundings themselves. The example above will now appear like this instead:

```
\begin{eqnarray*}
  \dbx &=& \dbx[5cm] \\  &=& \dbx
\end{eqnarray*}
```

Resulting in:

$$\begin{aligned}\boxed{} \quad &= \quad \boxed{} \\ &= \quad \boxed{}\end{aligned}$$

2 Behold the problems

2.1 The major problem: Spacing inconsistency

The most common way `eqnarray`-users write their displayed equations is via `eqnarray` and `eqnarray*` mixed with `equation`, `\[...\]`, or `$$...$$` constructions. Some even mix it with environments from the `amsmath` package (though this is mostly seen when a document has been written by more than one author).

This poses the major problem concerning `eqnarray` – spacing inconsistency. In the following example we consider a single line equation versus a multi-line equation.

```
\[ \dbx = \dbx \]
whereas
\begin{eqnarray*}
  \dbx &=& \dbx[3cm] \\  &=& \dbx
\end{eqnarray*}
```

which results in

$$\boxed{} = \boxed{}$$

whereas

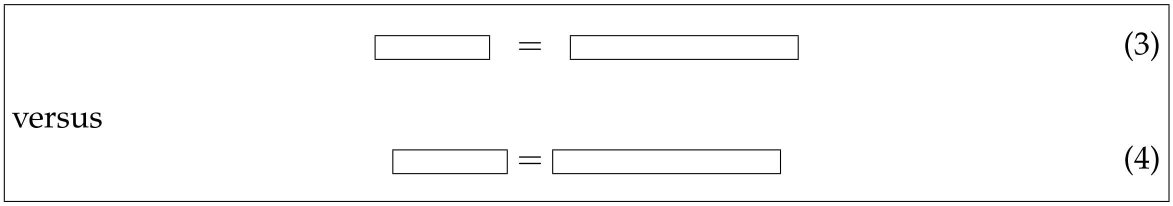
$$\begin{aligned}\boxed{} \quad &= \quad \boxed{} \\ &= \quad \boxed{}\end{aligned}$$

Can you spot the problem?

It is even more obvious when we place the same code in `eqnarray` and `equation` up against each other:

```
\begin{eqnarray} \dbx &=& \dbx[3cm] \end{eqnarray}
versus
\begin{equation} \dbx = \dbx[3cm] \end{equation}
```

which results in



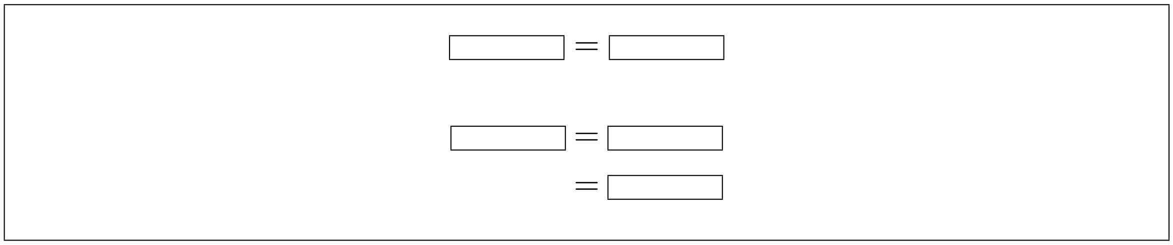
Can you see the difference?

We notice how the spacings around the ='s are inconsistent (not equal). Consistency being one of the key values in any good document design.[1]

Since `eqnarray` is (naively) build over the `array` environment we still have the `\arraycolsep` between columns, which then affects the spacing around the ='s in our case. We could change the value of `\arraycolsep`:

```
\setlength\arraycolsep{1.4pt}% some length
\[ \dbx = \dbx \]
\begin{eqnarray*}
  \dbx & = & \dbx  \\  &= & \dbx
\end{eqnarray*}
```

Resulting in:



Changing the value of `\arraycolsep`, however, will also change the appearance of any construction that might be using `array`, so that will not help either, see the following example.

[1] The spacing around the = signs should equal on both sides (not counting stretch), no matter which construction is used.

```
Before the change:
\begin{eqnarray*}
  A &=& \left(\begin{array}{cc}\dbx&\dbx\\\dbx&\dbx\end{array}\right)
\end{eqnarray*}
  after the change:
\setlength\arraycolsep{1.4pt}% some length
\begin{eqnarray*}
  A &=& \left(\begin{array}{cc}\dbx&\dbx\\\dbx&\dbx\end{array}\right)
\end{eqnarray*}
```

Resulting in:

Before the change:

$$A \quad = \quad \left(\begin{array}{cc}\boxed{} & \boxed{}\\ \boxed{} & \boxed{}\end{array}\right)$$

after the change:

$$A = \left(\begin{array}{cc}\boxed{}\boxed{}\\ \boxed{}\boxed{}\end{array}\right)$$

Some people argue that this larger spacing is a good thing, that it helps understanding the equations in question. For that to be true the author should do this with every single equation, whether the equation was written using `eqnarray` or not. Consistency above all. We can plainly see that `eqnarray` does not follow the spacing conventions Knuth set out in TEX, whereas both `equation` and `\[...\]` do.

Here is another example from a set of notes I have been editing (actual code from the original unedited notes).

```
\begin{eqnarray*}
  {\cal C}_{0} &\subseteq&  {\cal C}\subseteq
  \sigma ({\cal  C}_{0},{\cal N}) ,
\end{eqnarray*}
```

$$\mathcal{C}_0 \quad \subseteq \quad \mathcal{C} \subseteq \sigma(\mathcal{C}_0,\mathcal{N}),$$

Which makes one wonder if LaTeX authors don't even notice the difference in spacing, or do they just accept this as a fact of life?

Even though eqnarray might not be recommended for one-liners, they do still appear quite a lot in the 'wild'.

As eqnarray is the only multi-line construction for plain LaTeX, what should be used instead? Short answer: Use the environments from the amsmath package, in particular the align environment.

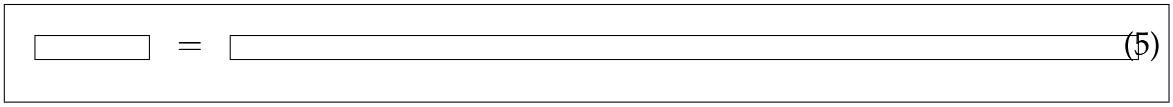
Longer answer: There are a few packages that can help including nath, mathenv and amsmath. Using amsmath is highly recommended since it is already an integrated part of every LaTeX installation.

For those not familiar with the amsmath package we present a few useful constructions in Appendix A.

2.2 Eqnarray might overwrite equation numbers

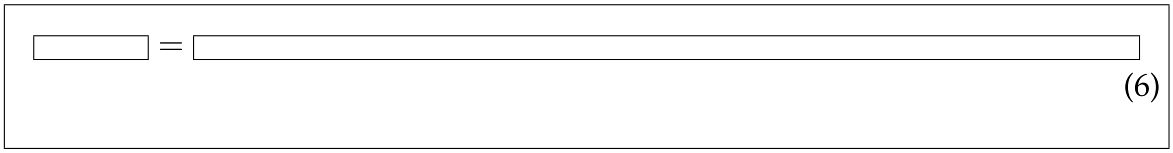
Given a long formula which happens to fit within one line if were it not for the equation number, eqnarray will happily just ignore the equation number without any warnings.

```
\begin{eqnarray}
  \dbx &=& \dbx[12cm]
\end{eqnarray}
```



At least the environments from the amsmath bundle take the equation number into consideration. Here is an example using align[2]:

```
\begin{align}
  \dbx &= \dbx[12.5cm]
\end{align}
```



[2] We need a longer box in this example to compensate for the better spacing around the =.

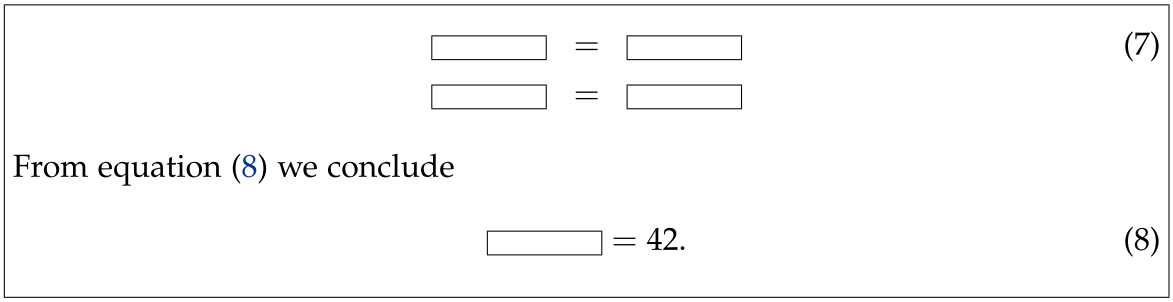
2.3 Silence of the labels

Part of my job is to process preprint series published by my department. This brings me into contact with a lot of different LaTeX writing/usage styles. One thing that I frequently do (as part of my visual improvement procedures) is to convert `eqnarray` environments into `align` environments (or similar). This is where one starts to find the hidden label errors. Most of these occur when two or more people have been writing/editing the same file.

Here is the first example:

```
\begin{eqnarray}
  \dbx & = & \dbx  \\ \dbx & = & \dbx \label{eq:2} \nonumber
\end{eqnarray}
From equation (\ref{eq:2}) we conclude
\begin{equation}
  \dbx=42.
\end{equation}
```

So the author had an equation which he or she no longer wanted to have numbered (`\nonumber`). Which is perfectly reasonable, but the author did not check if the now *dead* label (`eq:2`) was actually referred to elsewhere. The result is as follows:



$$\boxed{} = \boxed{} \tag{7}$$

$$\boxed{} = \boxed{}$$

From equation (8) we conclude

$$\boxed{} = 42. \tag{8}$$

Huh? This might end up as an interesting form of argumentation. It seems as if `eqnarray` actually steps up the equation counter at the start of every line (hence `\label` catches something) and when it encounters `\nonumber` it does not write any equation number and steps the equation counter one down again. On a side note, `equation` has the same problem if one mixes it with `\nonumber` (something which is *not* fixed by using `amsmath`).

The worst thing here is that `eqnarray` does this silently, without warnings, so if you do not know that this might happen you will never notice it unless someone actually reads the article.

As it so happens I recently received an article which showed exactly the same problem in eqnarray*. Here one just has to place a label inside a non-numbering eqnarray* (notice the use of \theequation to show the current value of the equation number):

```
Current equation number: \theequation
\begin{eqnarray*}
  \label{eq:4}
  \theequation & = & \dbx
\end{eqnarray*}
The reference is (\ref{eq:4}). Current equation number: \theequation
```

Resulting in:

Current equation number: 8

$$9 = \boxed{}$$

The reference is (9). Current equation number: 8

Who smells rat? So, even in eqnarray* the equation counter is stepped up, and later stepped down at the end of each line. As we have see above, this is not such a clever idea.

3 Solution

The best solution is to *not* use the environment at all. Use the environments from amsmath instead, but in case that will not do, the mathenv package reimplements eqnarray to get a better look. It also removes the restraint on the number of columns in an eqnarray.

But sadly we still see a lot of Journals and Publishing Houses who still recommend (or at least mention) the use of eqnarray in their guides for authors.

A The amsmath package

For more information about amsmath see [2], [1] and [3] in order of recommended reading. Here are a few interesting constructions.

Please note that all of the following examples require that the user adds

```
\usepackage{amsmath}
```

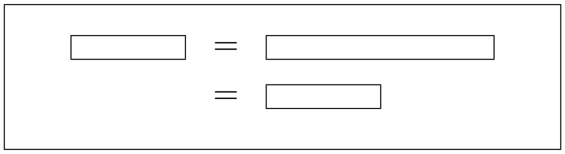
to their preamble.

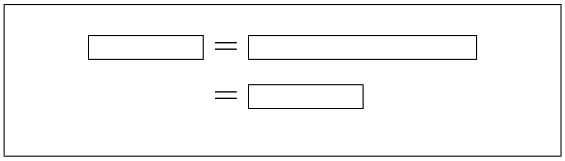
One thing to note about amsmath is that *every* environment from amsmath that provides equation numbers also has a $*$-version that does not. The package also includes an equation* environment which is missing from plain LaTeX.

Now the first thing we need is a replacement for eqnarray. We choose align, which has a slightly different syntax than eqnarray:

```
\begin{eqnarray*}
  \dbx &=& \dbx[3cm] \\
       &=& \dbx
\end{eqnarray*}
```

```
\begin{align*}
  \dbx &= \dbx[3cm] \\
       &= \dbx
\end{align*}
```



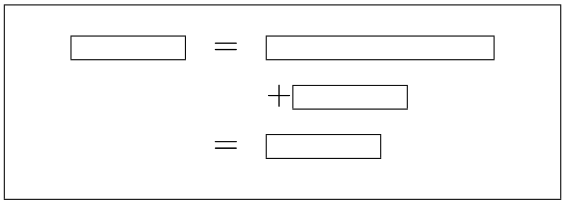


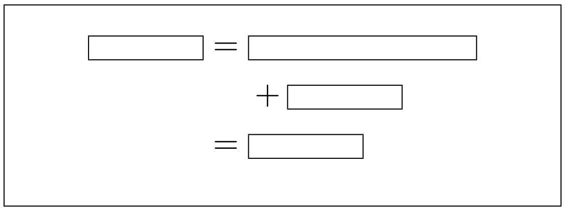
Note the reduced number of &'s.

Here is another common eqnarray construction and its align counterpart:

```
\begin{eqnarray*}
  \dbx &=& \dbx[3cm] \\
        && + \dbx    \\
       &=& \dbx
\end{eqnarray*}
```

```
\begin{align*}
  \dbx = {} & \dbx[3cm] \\
            & + \dbx    \\
       = {} & \dbx
\end{align*}
```





Notice the use of {} when the & is placed on the right of a relational symbol. Also note that the spacing around the + is correct in the align case but when using eqnarray.

One construction that is not easily achieved using plain LaTeX is a formula spread over several lines but with only one number for the entire formula. Again, this is easy using constructions from the amsmath package:

```
\begin{equation}
  \begin{split}
    \dbx & = \dbx[3cm] \\
         & = \dbx
  \end{split}
\end{equation}
```

$$\begin{split} \boxed{} &= \boxed{} \\ &= \boxed{} \end{split} \tag{9}$$

Notice how the equation number is placed vertically centred. The syntax for split is otherwise more or less the same as for align*.

References

[1] American Mathematical Society, *User's Guide for the* amsmath *Package*, 1999. Normally included in every LaTeX installation as amsldoc but also available via CTAN:macros/latex/required/amslatex/math/

[2] Michael Downes, *Short Math Guide*, 2002. Short introduction to the amsmath and amssymb packages. Available via ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf

[3] Herbert Voß, *Math mode*, 2006. Large summary describing various mathematical constructions, both with and without the amsmath package. Available via CTAN:info/math/voss/mathmode/Mathmode.pdf