The `empheq` package*
Emphasizing equations in LaTeX 2ε†

Morten Høgholm

2003/05/11

Abstract

The `empheq` package automatically detects several `amsmath` environments and the size of the displayed math material. The user interface makes it easy to add various kinds of visual markup to these equations.

Contents

1 Introduction

Users who have wanted to put a system of equations inside boxes has hitherto been forced to use the features of `fancybox` or the `\boxed` command of `amsmath`. Both alternatives have serious limitations though. `fancybox` allows only `eqnarray`-style equations and at the end of the day they aren't all that pretty. Most mathematical typesetting in LaTeX is done with the aid of `amsmath` anyway, but it only offers the single line quick-fix `\boxed`. What we really want is something that will enable us to do something along the likes of this:

$$\boxed{\tilde{S} = 1 \Rightarrow \begin{cases} a = \displaystyle\int_{-2}^{3} tb^t \, dt \quad \text{and} & \text{(1a)} \\ c = d - a & \text{(1b)} \end{cases}}$$

*This file has version number v0.7, last revised 2003/05/11.

†Thanks to Lars Madsen for asking for the subtle feature that evolved into this package.

As you can see, empheq can do all the tricks the cases package by Donald Arseneau can do and more. It even supports subequations—as shown in equations (1a) and (1b)—from amsmath without complaining.

In order to combine the best of two worlds the empheq package tries to take advantage of the widespread features of amsmath. As it should be well known if you read this, amsmath has amongst its arsenal of structures align, gather, alignat and multline. empheq works with these as well as their starred variants.

"But what about equation?" you say. equation is (in my opinion) merely a poor man's gather. If you really want to use empheq's features on a one-liner (no pun intended) go with gather. Anyway the real goal of this package is to do nifty tricks with multi line equations ...

2 Examples

Any options given to empheq is passed on to amsmath, thus the line

\usepackage[⟨*options*⟩]{empheq}

will load amsmath with exactly those options. empheq doesn't redefine anything so to use it in an existing document you need only replace amsmath with empheq—no harm done.

empheq is *really* easy to use; you simply put an empheq environment around your amsmath environment:

```
\begin{empheq}
  \begin{align}
    E&=mc^2 \\
    Y&= \sum_{n=1}^\infty \frac{1}{n^2}
  \end{align}
\end{empheq}
```

$$E = mc^2 \tag{2}$$

$$Y = \sum_{n=1}^{\infty} \frac{1}{n^2} \tag{3}$$

Impressed? No? Well then I guess it's about time I told you about the *optional* argument of the empheq environment. It allows you control what material to put on either side of the math and the sort of box to go around it all. That means that we can say

```
\begin{empheq}[boxtype=\fbox]
  \begin{align}
    E&=mc^2 \\
    Y&= \sum_{n=1}^\infty \frac{1}{n^2}
  \end{align}
\end{empheq}
```

to obtain the display

$$\boxed{\begin{aligned} E &= mc^2 \\ Y &= \sum_{n=1}^{\infty} \frac{1}{n^2} \end{aligned}} \tag{4--5}$$

This requires the use of the `keyval` package from the tools bundle, which is undoubtedly installed on your system. When using the `keyval` package there are a few things we need to keep in mind. In mathematical typesetting '`=`' and '`,`' are quite frequently used, thus requiring the user to enclose them in braces:

```
\begin{empheq}[boxtype=\fbox,
               Left={(a,b)=(c,i) \Rightarrow\empheqlbrace}]
\begin{alignat}{2}
(a,b)&= (\cos^a c, \tan^b i)&& \quad\text{for $i>1$}\\
(a,b)&= (\arccos x^a, \arctan i^b)&& \quad\text{for $i\leq1$}
\end{alignat}
\end{empheq}
```

$$\boxed{(a,b) = (c,i) \Rightarrow \begin{cases} (a,b) = (\cos^a c, \tan^b i) & \text{for } i > 1 \quad (6) \\ (a,b) = (\arccos c^a, \arctan i^b) & \text{for } i \leq 1 \quad (7) \end{cases}}$$

The same with `cases` from `amsmath` for comparison:

$$\boxed{(a,b) = (c,i) \Rightarrow \begin{cases} (a,b) = (\cos^a c, \tan^b i) & \text{for } i > 1 \\ (a,b) = (\arccos x^a, \arctan i^b) & \text{for } i \leq 1 \end{cases}}$$

Notice that `cases` uses an array for the conditions, so you have to force `\displaystyle` yourself. This is not needed with `empheq` as you're already using a `\displaystyle` environment such as `gather` etc.

Observe what can be done if we replace `\fbox` with another framed box and add some space on all sides:

```
\definecolor{lightblue}{rgb}{.8, .8, 1}
\begin{empheq}[boxtype={\setlength{\fboxsep}{10pt}%
                        \colorbox{lightblue}},
               Right={\empheqrbrace \beta}]
    .
    .
    .
\end{empheq}
```

$$\left.\begin{aligned} a &= \int_{-2}^{3} tb^t\,dt \quad \text{and} \quad (8) \\ c &= d - a \quad (9) \end{aligned}\right\} \beta$$

As a convenience for the user, it is also possible to declare delimiters with the commands `\DeclareRightDelimiter{`⟨*delimiter*⟩`}` and its companion `\DeclareLeftDelimiter{`⟨*delimiter*⟩`}`. For instance the commands `\DeclareRightDelimiter{\rangle}` defines `\empheqrangle` and `\DeclareLeftDelimiter{\langle}` defines `\empheqlangle`. The usage is simple.

```
\begin{empheq}[Right=\empheqrangle,
               Left=\empheqlangle]
```

```
  \begin{gather}
    a=b \\
    c=d
  \end{gather}
\end{empheq}
```

$$\left\langle \begin{gathered} a = b \\ c = d \end{gathered} \right. \tag{10}$$
$$\tag{11}$$

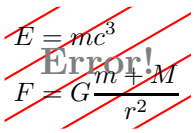
`\DeclareRightDelimiter` and its companion have an optional argument which controls spacing (default is a negative thinspace), but beware: It will simply overwrite the original definition and it might not look all that pretty as the following example shows:

```
\DeclareRightDelimiter[\>]{\rangle}
\DeclareLeftDelimiter[\mkern-10mu]{\langle}
```

$$\left\langle \begin{gathered} a = b \\ c = d \end{gathered} \right\rangle \tag{12}$$
$$\tag{13}$$

These examples will have given you a glimpse of the possibilities `empheq` offers. In particular the commands `\empheqlbrace` and `\empheqrbrace` produce braces just tall enough to encompass the display. You can of course use `\big...` delimiters if you want.

If you want you can even create your own box and use it with `empheq`. In the following silly example I have created `\ErrorBox` with `PSTricks` to achieve a disturbing effect:



Remember one thing when creating your own box: Make it symmetrical.

3 Implementation

3.1 Strategy

Whenever `amsmath` is typesetting a display environment it actually does it twice; the first for the "measuring" phase and the second for the "production" phase. The `empheq` package uses a similar approach:

- Collects the input in a token register for later use.
- Typesets the input in the background and checks certain values to determine the kind of math structure you just typed.
- Based on the given information, alterations are made to various lengths.
- Then the equations are typeset again and stored in a box for easy retrieval.

- Finally the saved box is put into a box of the user's liking. Any material to the right or left of the math structure (added with the `Left` and `Right` keys) are put into the box as well.

Assuming we have found out which `amsmath` structure the user typed, there are still a few things we need to sort out before we proceed.

First of all the equation number should still be aligned with those of the ordinary math structures, which is a problem as soon as the additions we make to the left and the right differ—presumably they will. And when we are in `fleqn` mode, we should make sure that the structure is shifted just enough to the right so that all equations still are right adjusted. Therefore we need to make clever changes to the line width to make sure the margins come out right.

Then assuming we have overcome all of this there's still the final step: Placing the math structure in the appropriate box at the appropriate position. Once again we need to do some manipulation with spacing to make it just right.

3.2 Preliminary actions

To avoid clashes with other packages, all private commands will contain the sequence `@emphEQ@`.

We will need `amsmath` of course and it would be nice if `empheq` would recognize all the `amsmath` options so we make that happen. To make sure the user has an updated system the newest release of `amsmath` is required.

`calc` is used to increase readability of the code (thus making it *far* easier to write too) and `keyval` is used for user friendliness. They should all be part of any LaTeX system.

```
%<*package>
\RequirePackageWithOptions{amsmath}[2000/07/18]
\RequirePackage{calc,keyval}
```

`\if@emphEQ@FLL` `\if@emphEQ@FLR` `\if@emphEQ@L` `\if@emphEQ@R`

There are four available layouts for mathematical displays: `[fleqn,leqno]` (FLL), `[fleqn,reqno]` (FLR), `[leqno]` (L) and `[reqno]` (R). We create four TeX conditionals to make the algorithm easier to read as well as to speed up things a bit.

```
\newif\if@emphEQ@FLL
\newif\if@emphEQ@FLR
\newif\if@emphEQ@L
\newif\if@emphEQ@R
```

We then perform a simple test to determine the current mode.

```
\if@fleqn
    \iftagsleft@
        \@emphEQ@FLLtrue
    \else
        \@emphEQ@FLRtrue
    \fi
\else
    \iftagsleft@
        \@emphEQ@Ltrue
    \else
        \@emphEQ@Rtrue
    \fi
```

```
\fi
```

\if@emphEQ@gather
\if@emphEQ@align
\if@emphEQ@alignat
\if@emphEQ@multline
\if@emphEQ@NoNumEq

Since we're going to determinate which math structure the user has input, we create some more conditionals to improve speed and readability. Furthermore it will show crucial to know whether or not the equation counter has been stepped since multline∗ and multline don't have exactly the same structure.

```
\newif\if@emphEQ@gather
\newif\if@emphEQ@align
\newif\if@emphEQ@alignat
\newif\if@emphEQ@multline
\newif\if@emphEQ@NoNumEq
```

\@emphEQ@Box
\@emphEQ@BoxWidth
\@emphEQ@displaywidth
\@emphEQ@mathbody

We need a single box to hold the math structure, a length so we can remember the size of the displayed math for later use and a token register for the math environment.

```
\newsavebox{\@emphEQ@Box}
\newlength{\@emphEQ@BoxWidth}
\newlength{\@emphEQ@displaywidth}
\newtoks\@emphEQ@mathbody
```

\@emphEQ@Left
\@emphEQ@Right
\@emphEQ@lwd
\@emphEQ@rwd

Here are some vital parts: The additional informations the user want to display on either side of the math structure are saved by `\@emphEQ@Left` and `\@emphEQ@Right` and `\@emphEQ@lwd` and `\@emphEQ@rwd` measure the width of them for later use.

```
\newcommand{\@emphEQ@Left}{}
\newcommand{\@emphEQ@Right}{}
\newlength{\@emphEQ@lwd}
\newlength{\@emphEQ@rwd}
```

\@emphEQ@boxtype
\@emphEQ@boxtypelength
\@emphEQ@shadowbox
\if@emphEQ@shadowbox

To make sure spacing is right when putting the math structure into a box, we need to know what kind of box it is: The width is extremely important. Often we will have to subtract or add half of an empty (but *not* zero-width) box' width to certain lengths. When using an `\fbox` or `\colorbox` this is just `\fboxrule`+`\fboxsep`, but when using say, `\doublebox` of `fancybox` fame the necessity arises. `fancybox` also defines `\shadowbox` which needs extra taking care of, since it's asymmetrical unlike most other framed boxes.

Thus we test whether or not `fancybox` has been loaded at the beginning of the document. If it has, we don't lift a finger; if it hasn't, we let `\fbox` be an alias for `\shadowbox`. Then we create a private version of `\shadowbox` for testing purposes and a conditional for the algorithm.

```
\newcommand{\@emphEQ@boxtype}{}
\newlength{\@emphEQ@boxtypelength}
\AtBeginDocument{%
    \@ifpackageloaded{fancybox}{}{%
        \let\shadowbox=\fbox}}
\newcommand{\@emphEQ@shadowbox}{\shadowbox}
\newif\if@emphEQ@shadowbox
```

\@emphEQ@templength
@emphEQ@EqCtr

We are almost ready to go. We only need two temporary little helpers: one length and one counter.

```
\newlength{\@emphEQ@templength}
\newcounter{@emphEQ@EqCtr}
```

3.2.1 Faraway, so close

\@emphEQ@CloseEnough When determining the display type we will occasionally have to check if an equation of the form

$$\texttt{\totwidth@} - \texttt{\tagshift@} = \texttt{\linewidth}$$

is true. However small rounding errors *can* occur in alignat so we would rather see if the equation

$$-\delta < \texttt{\totwidth@} - \texttt{\tagshift@} - \texttt{\linewidth} < \delta$$

holds for some sufficiently small $\delta > 0$ pt. I have used $\delta = 5$ sp.

```
\newcommand{\@emphEQ@CloseEnough}[4]{%
    \setlength{\@tempdima}{#1-#2}%
    \ifdim\@tempdima>-5sp
        \ifdim\@tempdima<5sp
            #3
        \else
            #4
        \fi
    \else
        #4
    \fi
}
```

3.3 Saving the input

\@emphEQ@MainEnv
\end@emphEQ@MainEnv
\@emphEQ@get@body
\@emphEQ@findend
\@emphEQ@EnvName

We start out by saving the input in the token register \@emphEQ@mathbody. The method is heavily inspired by tabularx ... No need to copy the words of David Carlisle here—his code will do!

```
\def\@emphEQ@MainEnv{%
    \edef\@emphEQ@EnvName{\@currenvir}%
    {\ifnum0=`}\fi
    \@emphEQ@mathbody{}\@emphEQ@get@body
}
\let\end@emphEQ@MainEnv\relax
\long\def\@emphEQ@get@body#1\end{%
    \@emphEQ@mathbody\expandafter{\the\@emphEQ@mathbody #1}%
    \@emphEQ@findend
}
\def\@emphEQ@findend#1{%
    \def\@tempa{#1}%
    \ifx\@tempa\@emphEQ@EnvName
        \expandafter\@emphEQ@PostProcess
    \else
        \@emphEQ@mathbody\expandafter{\the\@emphEQ@mathbody\end{#1}}%
        \expandafter\@emphEQ@get@body
    \fi
}
\def\@emphEQ@EnvName{@emphEQ@MainEnv}
```

@emphEQ@PostProcess Now we can go on with the post processing, which is the real workhorse in this package. The first job is to test if we're using a \shadowbox and then we set some

amsmath dimensions to zero—amsmath doesn't do this by itself, and for that we can be thankful. Otherwise you wouldn't be reading this now ;-)

```
\newcommand*{\@emphEQ@PostProcess}{%
    \ifx\@emphEQ@boxtype\@emphEQ@shadowbox
        \@emphEQ@shadowboxtrue
    \else
        \@emphEQ@shadowboxfalse
    \fi
    \tagshift@=0pt \tagwidth@=0pt \totwidth@=0pt
    \eqnshift@=0pt \alignsep@=0pt
```

3.4 The first pass

We then run the first pass to find out which environment we've got. However we have to store the equation counter because it is stepped in this pass, and we certainly don't want that to happen. As a side effect this will also tell us if the equations are numbered.

If we are to put a brace to the left of the display, it would be rather pleasant to know just how high the structure is, since this will affect the height and more importantly the width the brace: we will need to know the *exact* width of such material. Hence we put it in a box which is easy to measure and fiddle around with. To achieve this we put the contents of the token register `\@emphEQ@mathbody` in a minipage inside a lrbox. Alas, lrbox won't let us escape into display math mode, so we play a dirty trick on it and use a minipage, which will let us have our way with it.

The minipage starts in horizontal mode and then goes straight to display math mode, thus adding some space before the display math. However there was nothing on the first line in the minipage thus leading TEX to add a `\abovedisplayskip` before the display. We subtract this as fast as we can! Furthermore it showed in test runs, that an additional 1 pt had to be subtracted in the top—but not in the bottom. It has nothing to do with any of the standard page dimensions or parameters. Strange indeed.

```
    \setcounter{@emphEQ@EqCtr}{\value{equation}}%
    \begin{lrbox}{\@emphEQ@Box}%
        \begin{minipage}{\linewidth}%
        \vskip-\abovedisplayskip\relax\vskip-1pt\relax
        \the\@emphEQ@mathbody
        \end{minipage}%
    \end{lrbox}%
```

We then look at the equation counter to see if we had any numbered equations and reset it if that was the case. If we used a multline∗ we can detect this by measuring `\tagwidth@`.

```
    \ifnum\value{equation}=\value{@emphEQ@EqCtr}%
        \ifdim\tagwidth@>0pt
        \else
            \@emphEQ@NoNumEqtrue
        \fi
    \else
        \setcounter{equation}{\value{@emphEQ@EqCtr}}%
    \fi
```

Now we can determine the various math environments. Firstly we measure an empty `\@emphEQ@boxtype`—we will need this information when the math structure is finally put into a box of this type, but we have to take it into account when calculating the allowable width of multline displays because we want to use the margins already set for this environment.

```
\settowidth{\@emphEQ@boxtypelength}{\@emphEQ@boxtype{}}%
```

The idea is that each of the amsmath multiple line environments can be singled out because it uses a certain dimension in a way none of the others use it.

align: Uses `\alignsep@` and sets it to something positive. None of the others use it.

multline: Doesn't use `\eqnshift@`.

gather: Doesn't use `\tagshift@`.

alignat: Can be determined because some dimensions will have special relations e.g., when in `[fleqn,reqno]` mode the equation

$$\texttt{\totwidth@} - \texttt{\tagshift@} = \texttt{\linewidth}$$

holds[1] and similarly for the other three display modes. See the code for these equations.

3.4.1 [fleqn,leqno]

```
\if@emphEQ@FLL % fleqn and leqno
    \ifdim\alignsep@>0pt % => align
        \@emphEQ@aligntrue
        \setlength{\@emphEQ@displaywidth}{\tagshift@-\@mathmargin}%
    \else
        \ifdim\eqnshift@=0pt % => multline %tagwidth@>
            \@emphEQ@multlinetrue
            \if@emphEQ@NoNumEq
                \setlength{\@emphEQ@displaywidth}{%
                    \linewidth-\multlinegap-\@mathmargin
                    -\@emphEQ@boxtypelength}%
            \else
                \setlength{\@emphEQ@displaywidth}{%
                    \linewidth-\tagwidth@-\multlinegap
                    -\multlinetaggap-\@emphEQ@boxtypelength}%
            \fi
        \else
            \ifdim\tagshift@=0pt % => gather
                \@emphEQ@gathertrue
                \setlength{\@emphEQ@displaywidth}{%
                    \totwidth@-\@mathmargin}%
            \else
                \ifdim\tagshift@=\totwidth@ % => alignat
                    \@emphEQ@alignattrue
                    \setlength{\@emphEQ@displaywidth}{%
```

[1] Well almost—they may be off by a few scaled points. See section 3.2.1 for further details.

```
                            \totwidth@-\@mathmargin}%
                    \else
                        %Nothing thus far...
                    \fi
                \fi
            \fi
        \fi
    \fi
```

3.4.2 [fleqn,reqno]

```
    \if@emphEQ@FLR % fleqn and reqno
        \ifdim\alignsep@>0pt % => align
            \@emphEQ@aligntrue
            \setlength{\@emphEQ@displaywidth}{%
                \linewidth-\@mathmargin+\tagshift@}%
        \else
            \ifdim\eqnshift@=0pt % => multline
                \@emphEQ@multlinetrue
                \if@emphEQ@NoNumEq
                    \setlength{\@emphEQ@displaywidth}{%
                        \linewidth-\multlinegap-\@mathmargin
                        -\@emphEQ@boxtypelength}%
                \else
                    \setlength{\@emphEQ@displaywidth}{%
                        \linewidth-\tagwidth@-\@mathmargin
                        -\multlinetaggap-\@emphEQ@boxtypelength}%
                \fi
            \else
                \ifdim\tagshift@=0pt % => gather
                    \@emphEQ@gathertrue
                    \setlength{\@emphEQ@displaywidth}{%
                        \totwidth@-\@mathmargin}%
                \else
                    \@emphEQ@CloseEnough{\totwidth@-\tagshift@}%
                    {\linewidth}%
                    {\@emphEQ@alignattrue
                    \setlength{\@emphEQ@displaywidth}%
                    {\totwidth@-\eqnshift@}}{}%
                \fi
            \fi
        \fi
    \fi
```

3.4.3 [reqno]

```
    \if@emphEQ@R % reqno
        \ifdim\alignsep@>0pt % => align
            \@emphEQ@aligntrue
            \setlength{\@emphEQ@displaywidth}{\totwidth@}%
        \else
```

```
            \ifdim\eqnshift@=0pt % => multline
                \@emphEQ@multlinetrue
                \if@emphEQ@NoNumEq
                    \setlength{\@emphEQ@displaywidth}{%
                        \linewidth-2\multlinegap
                        -\@emphEQ@boxtypelength}%
                \else
                    \setlength{\@emphEQ@displaywidth}{%
                        \linewidth-\tagwidth@-\multlinegap
                        -\multlinetaggap-\@emphEQ@boxtypelength}%
                \fi
            \else
                \ifdim\tagshift@=0pt % => gather
                    \@emphEQ@gathertrue
                    \setlength{\@emphEQ@displaywidth}{\totwidth@}%
                \else
                    \@emphEQ@CloseEnough{\tagshift@}{-\eqnshift@}%
                    {\@emphEQ@alignattrue
                    \setlength{\@emphEQ@displaywidth}{\totwidth@}}{}%
                \fi
            \fi
        \fi
    \fi
```

3.4.4 [leqno]

```
    \if@emphEQ@L % leqno
        \ifdim\alignsep@>0pt % => align
            \@emphEQ@aligntrue
            \setlength{\@emphEQ@displaywidth}{\totwidth@}%
        \else
            \ifdim\eqnshift@=0pt % => multline
                \@emphEQ@multlinetrue
                \if@emphEQ@NoNumEq
                    \setlength{\@emphEQ@displaywidth}{%
                        \linewidth-2\multlinegap
                        -\@emphEQ@boxtypelength}%
                \else
                    \setlength{\@emphEQ@displaywidth}{%
                        \linewidth-\tagwidth@-\multlinegap
                        -\multlinetaggap-\@emphEQ@boxtypelength}%
                \fi
            \else
                \ifdim\tagshift@=0pt % => gather
                    \@emphEQ@gathertrue
                    \setlength{\@emphEQ@displaywidth}{\totwidth@}%
                \else
                    \@emphEQ@CloseEnough{%
                        \linewidth-\eqnshift@}{\tagshift@}{%
                        \@emphEQ@alignattrue
                        \setlength{\@emphEQ@displaywidth}{\totwidth@}}%
                        {}%
                \fi
```

```
            \fi
        \fi
    \fi
```

Phew! That was tedious, but necessary. I wish I could say it got better further down the road, but it doesn't. Now we can measure the width of the additional material on both sides of the math structure. We add braces here and there to avoid any `\thinmuskip`'s.

```
    \settowidth{\@emphEQ@lwd}{${\@emphEQ@Left}%
                              {\vphantom{\usebox{\@emphEQ@Box}}}$}%
    \settowidth{\@emphEQ@rwd}{${\vphantom{\usebox{\@emphEQ@Box}}}%
                              {\@emphEQ@Right}$}%
```

3.5 The second pass

When we try to create an effect as in the preliminary example there are a few things we need to tell LaTeX for the overall result to be acceptable. If the display math is centered, then the new one should also be centered. This means that when typesetting the math structure for the second time, we need to make adjustments to `\linewidth`, `\@mathmargin` etc. for the equation tag to be placed correctly.

That process is extremely tedious, but there are only few shortcuts, so we just go through each of the four environments, check each of the four display modes and alter dimensions a we trudge along. First we check if we are trying to put something into an asymmetrical box and then make it symmetrical be removing the size of the shadow.

```
    \begin{lrbox}{\@emphEQ@Box}%
    \if@emphEQ@shadowbox
        \addtolength{\@emphEQ@boxtypelength}{-\shadowsize}%
    \fi
```

3.5.1 align

```
    \if@emphEQ@align
        \if@emphEQ@FLL
            \begin{minipage}{\linewidth+\@emphEQ@lwd
                +.5\@emphEQ@boxtypelength}%
                \addtolength{\@mathmargin}{\@emphEQ@lwd
                    +.5\@emphEQ@boxtypelength}%%
        \fi
        \if@emphEQ@FLR
            \begin{minipage}{\linewidth}%
                \addtolength{\@mathmargin}{\@emphEQ@lwd
                    +.5\@emphEQ@boxtypelength}%%
        \fi
        \if@emphEQ@L
            \begin{minipage}{\linewidth+\@emphEQ@lwd-\@emphEQ@rwd}%
        \fi
        \if@emphEQ@R
            \begin{minipage}{\linewidth-\@emphEQ@lwd+\@emphEQ@rwd}%
        \fi
    \fi
```

3.5.2 gather

```
\if@emphEQ@gather
    \if@emphEQ@FLL
        \begin{minipage}{\linewidth+\@emphEQ@lwd
            +.5\@emphEQ@boxtypelength}%
            \addtolength{\@mathmargin}{\@emphEQ@lwd
                +.5\@emphEQ@boxtypelength}%%
    \fi
    \if@emphEQ@FLR
        \begin{minipage}{\linewidth}%
            \addtolength{\@mathmargin}{\@emphEQ@lwd
                +.5\@emphEQ@boxtypelength}%%
    \fi
    \if@emphEQ@R
        \setlength{\@emphEQ@templength}{\linewidth-\totwidth@
                  -\eqnshift@}%
        \begin{minipage}{\linewidth-\@emphEQ@lwd+\@emphEQ@rwd
                         +\@emphEQ@templength-\eqnshift@}%
    \fi
    \if@emphEQ@L
        \setlength{\@emphEQ@templength}{\linewidth-\totwidth@
                  -\eqnshift@}%
        \begin{minipage}{\linewidth+\@emphEQ@lwd-\@emphEQ@rwd
                         -\@emphEQ@templength+\eqnshift@}%
    \fi
\fi
```

3.5.3 alignat

```
\if@emphEQ@alignat
    \if@emphEQ@FLL
        \begin{minipage}{\linewidth+\@emphEQ@lwd
            +.5\@emphEQ@boxtypelength}%
            \addtolength{\@mathmargin}{\@emphEQ@lwd
                +.5\@emphEQ@boxtypelength}%%
    \fi
    \if@emphEQ@FLR
        \begin{minipage}{\linewidth}%
            \addtolength{\@mathmargin}{\@emphEQ@lwd
                +.5\@emphEQ@boxtypelength}%%
    \fi
    \if@emphEQ@L
        \begin{minipage}{\linewidth+\@emphEQ@lwd-\@emphEQ@rwd}%
    \fi
    \if@emphEQ@R
        \begin{minipage}{\linewidth-\@emphEQ@lwd+\@emphEQ@rwd}%
    \fi
\fi
```

3.5.4 multline

```
\if@emphEQ@multline
    \if@emphEQ@FLL
        \begin{minipage}{\linewidth}%
            \if@emphEQ@NoNumEq
                \addtolength{\@mathmargin}{\@emphEQ@lwd
                    +.5\@emphEQ@boxtypelength}%
                \if@emphEQ@shadowbox
                    \addtolength{\@mathmargin}{\shadowsize}
                \fi%
            \else
                \addtolength{\multlinetaggap}{\@emphEQ@lwd
                    +.5\@emphEQ@boxtypelength}%
                \if@emphEQ@shadowbox
                    \addtolength{\multlinegap}{\shadowsize}%
                \fi
            \fi
            \addtolength{\multlinegap}{\@emphEQ@rwd
                +.5\@emphEQ@boxtypelength}%
    \fi
    \if@emphEQ@FLR
        \begin{minipage}{\linewidth}%
            \if@emphEQ@NoNumEq
                \addtolength{\multlinegap}{\@emphEQ@rwd
                    +.5\@emphEQ@boxtypelength}%
                \if@emphEQ@shadowbox
                    \addtolength{\multlinegap}{\shadowsize}%
                \fi
            \else
                \addtolength{\multlinetaggap}{\@emphEQ@rwd
                    +.5\@emphEQ@boxtypelength}%
                \if@emphEQ@shadowbox
                    \addtolength{\multlinetaggap}{\shadowsize}%
                \fi
            \fi
            \addtolength{\@mathmargin}{\@emphEQ@lwd
                +.5\@emphEQ@boxtypelength}%
    \fi
    \if@emphEQ@L
        \begin{minipage}{\linewidth}%
            \if@emphEQ@NoNumEq
                \addtolength{\multlinegap}{.5\@emphEQ@rwd
                    +.5\@emphEQ@lwd+.5\@emphEQ@boxtypelength}%
                \if@emphEQ@shadowbox
                    \addtolength{\multlinegap}{.5\shadowsize}%
                \fi
            \else
                \addtolength{\multlinetaggap}{\@emphEQ@lwd
                    +.5\@emphEQ@boxtypelength}%
                \addtolength{\multlinegap}{\@emphEQ@rwd
                    +.5\@emphEQ@boxtypelength}%
                \if@emphEQ@shadowbox
```

```
                        \addtolength{\multlinegap}{\shadowsize}%
                    \fi
                \fi
        \fi
        \if@emphEQ@R
            \begin{minipage}{\linewidth}%
                \if@emphEQ@NoNumEq
                    \addtolength{\multlinegap}{.5\@emphEQ@rwd
                        +.5\@emphEQ@lwd+.5\@emphEQ@boxtypelength}%
                    \if@emphEQ@shadowbox
                        \addtolength{\multlinegap}{.5\shadowsize}
                    \fi
                \else
                    \addtolength{\multlinetaggap}{\@emphEQ@rwd
                        +.5\@emphEQ@boxtypelength}%
                    \addtolength{\multlinegap}{\@emphEQ@lwd
                        +.5\@emphEQ@boxtypelength}%
                    \if@emphEQ@shadowbox
                        \addtolength{\multlinetaggap}{\shadowsize}
                    \fi
                \fi
        \fi
    \fi
```

Finally we get to typeset the contents of `\@emphEQ@mathbody` for the second time...

```
    \vskip-\abovedisplayskip\relax\vskip-1pt\relax
    \the\@emphEQ@mathbody
    \end{minipage}%
    \end{lrbox}%
```

3.6 Putting it into the box

Because we were clever to start with—by measuring the width of the display math and storing it in `\@emphEQ@displaywidth`—we can now put it into a box of width `\@emphEQ@displaywidth+\@emphEQ@rwd+\@emphEQ@lwd`. This doesn't apply to multline though: it has to be set in a box of width `\@emphEQ@displaywidth`.

```
\setlength{\@emphEQ@BoxWidth}{\@emphEQ@displaywidth+\@emphEQ@rwd
                              +\@emphEQ@lwd}%
```

3.6.1 align

```
\if@emphEQ@align
    \if@emphEQ@FLL
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength%
            \kern-\@mathmargin
            \kern-\@emphEQ@lwd
            \usebox{\@emphEQ@Box}%
            \kern-\@emphEQ@displaywidth
```

```
            \kern-\@emphEQ@rwd
            \kern-\@emphEQ@lwd
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@FLR
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength%
            \kern-\@mathmargin
            \kern-\@emphEQ@lwd
            \usebox{\@emphEQ@Box}%
            \kern-\alignsep@
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@L
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][c]{%
            %\kern\eqnshift@
            %\kern-\alignsep@
            $\@emphEQ@Left$%
            \kern-\eqnshift@
            %\kern-\alignsep@
            \usebox{\@emphEQ@Box}%
            %\kern-\eqnshift@
            \kern-\alignsep@
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@R
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][c]{%
            $\@emphEQ@Left$%
            \kern-\eqnshift@
            \usebox{\@emphEQ@Box}%
            \kern-\eqnshift@
            $\@emphEQ@Right$}}%
    \fi
\fi
```

3.6.2 alignat

```
\if@emphEQ@alignat
    \if@emphEQ@FLL
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@mathmargin
            \kern-\@emphEQ@lwd
            \usebox{\@emphEQ@Box}%
            \kern\@mathmargin
            \kern\@emphEQ@displaywidth
            \kern-\linewidth
            $\@emphEQ@Right$}}%
    \fi
```

```
    \if@emphEQ@FLR
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@mathmargin
            \kern-\@emphEQ@lwd
            \usebox{\@emphEQ@Box}%
            \kern\@mathmargin
            \kern\@emphEQ@displaywidth
            \kern-\linewidth
            \kern\@emphEQ@lwd
            \kern.5\@emphEQ@boxtypelength
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@L
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][c]{%
            $\@emphEQ@Left$%
            \kern-\eqnshift@
            \usebox{\@emphEQ@Box}%
            \kern-\eqnshift@
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@R
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][c]{%
            $\@emphEQ@Left$%
            \kern-\eqnshift@
            \usebox{\@emphEQ@Box}%
            \kern-\eqnshift@
            $\@emphEQ@Right$}}%
    \fi
\fi
```

3.6.3 gather

```
\if@emphEQ@gather
    \if@emphEQ@FLL
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@mathmargin
            \kern-\@emphEQ@lwd
            \usebox{\@emphEQ@Box}%
            \kern-\linewidth
            \kern\@mathmargin
            \kern\@emphEQ@displaywidth
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@FLR
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize
            \fi
```

```
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@mathmargin
            \kern-\@emphEQ@lwd
            \usebox{\@emphEQ@Box}%
            \kern-\linewidth
            \kern.5\@emphEQ@boxtypelength
            \kern\@mathmargin
            \kern\@emphEQ@displaywidth
            \kern\@emphEQ@lwd
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@L
        \setlength{\tagwidth@}{\eqnshift@+\@emphEQ@rwd}%
        \setlength{\@emphEQ@templength}{\linewidth-\totwidth@
                                        -\eqnshift@}%
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][c]{%
            \ifdim\@emphEQ@templength<\tagwidth@
                \setlength{\tagwidth@}{\@emphEQ@templength
                                       +\@emphEQ@lwd}%
                \ifdim\tagwidth@<\eqnshift@
                    \kern-\@emphEQ@templength
                    \kern\eqnshift@
                \fi
            \else
                \kern-\@emphEQ@lwd
                \kern\@emphEQ@rwd
                \kern-\@emphEQ@templength
                \kern\eqnshift@
            \fi
            $\@emphEQ@Left$%
            \kern-\eqnshift@
            \usebox{\@emphEQ@Box}%
            %\kern-\eqnshift@
            \kern-\@emphEQ@lwd
            \kern\@emphEQ@rwd
            \kern-\@emphEQ@templength
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@R
        \setlength{\tagwidth@}{\eqnshift@+\@emphEQ@rwd}%
        \setlength{\@emphEQ@templength}{\linewidth-\totwidth@
                                        -\eqnshift@}%
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@BoxWidth][c]{%
            \kern-\@emphEQ@lwd
            \kern\@emphEQ@rwd
            \kern\@emphEQ@templength
            \kern-\eqnshift@
            $\@emphEQ@Left$%
            \ifdim\@emphEQ@templength>\tagwidth@
                \kern-\eqnshift@
            \else
                \kern\@emphEQ@lwd
                \kern-\@emphEQ@rwd
```

```
                \kern-\@emphEQ@templength
            \fi
            \usebox{\@emphEQ@Box}%
            \kern-\eqnshift@
            $\@emphEQ@Right$}}%
    \fi
\fi
```

3.6.4 multline

multline is somewhat more difficult because multline∗ differs on several accounts. Thus we make use of our little test to see if the math structure was numbered to begin with (\if@emphEQ@NoNumEq).

```
\if@emphEQ@multline
    \if@emphEQ@FLL
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@displaywidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@emphEQ@lwd
            \if@emphEQ@NoNumEq
                \if@emphEQ@shadowbox \kern-\shadowsize \fi
                \kern-\@mathmargin
            \else
                \kern-\tagwidth@
                \kern-\multlinetaggap
            \fi
            \usebox{\@emphEQ@Box}%
            \kern-\@emphEQ@boxtypelength
            \if@emphEQ@NoNumEq
                \kern-\multlinegap
            \else
                \kern\multlinegap
            \fi
            \kern-\@emphEQ@rwd
            \kern-\@emphEQ@lwd
            \kern-\@mathmargin
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@FLR
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@displaywidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@emphEQ@lwd
            \kern-\@mathmargin
            \usebox{\@emphEQ@Box}%
            \kern-\@emphEQ@boxtypelength
            \kern-\@emphEQ@rwd
            \kern-\@emphEQ@lwd
            \if@emphEQ@NoNumEq
                \kern-\multlinegap
            \else
                \kern-\multlinetaggap
```

```
            \fi
            \kern-\@mathmargin
            \kern-\tagwidth@
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@L
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@displaywidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@emphEQ@lwd
            \if@emphEQ@NoNumEq
                \kern-\multlinegap
                \kern.5\@emphEQ@lwd
                \kern-.5\@emphEQ@rwd
                \if@emphEQ@shadowbox \kern-.5\shadowsize \fi
            \else
                \kern-\multlinetaggap
                \kern-\tagwidth@
            \fi
            \usebox{\@emphEQ@Box}%
            \kern-.5\@emphEQ@boxtypelength
            \if@emphEQ@NoNumEq
                \kern.5\@emphEQ@rwd
                \kern-.5\@emphEQ@lwd
            \fi
            \kern-\@emphEQ@rwd
            \kern-\multlinegap
            $\@emphEQ@Right$}}%
    \fi
    \if@emphEQ@R
        \savebox{\@emphEQ@Box}{\makebox[\@emphEQ@displaywidth][l]{%
            \if@emphEQ@shadowbox \kern.5\shadowsize \fi
            $\@emphEQ@Left$%
            \kern-.5\@emphEQ@boxtypelength
            \if@emphEQ@NoNumEq
                \if@emphEQ@shadowbox \kern-.5\shadowsize \fi
                \kern-.5\@emphEQ@rwd
                \kern.5\@emphEQ@lwd
            \fi
            \kern-\@emphEQ@lwd
            \kern-\multlinegap
            \usebox{\@emphEQ@Box}%
            \kern-.5\@emphEQ@boxtypelength
            \kern-\@emphEQ@rwd
            \if@emphEQ@NoNumEq
                \kern-\multlinegap
                \kern-.5\@emphEQ@lwd
                \kern.5\@emphEQ@rwd
            \else
                \kern-\multlinetaggap
            \fi
            \kern-\tagwidth@
            $\@emphEQ@Right$}}%
```

```
    \fi
\fi
```

3.6.5 The final step

At last here comes the good part! We are now ready to place the box containing all of our math material. If we have used a multline we have to do a few kerning tricks but at least we would seem to know the drill by now... To make sure vertical spacing is ok, we put it all into a simple math environment `\[...\]`.

```
\[
    \if@emphEQ@align
        \if@emphEQ@L
            \kern-\alignsep@
            \kern\eqnshift@
        \fi
    \fi
    \if@emphEQ@gather
        \if@emphEQ@L
            \kern-\@emphEQ@templength
            \kern\eqnshift@
            \kern\@emphEQ@rwd
            \kern-\@emphEQ@lwd
        \fi
        \if@emphEQ@R
            \setlength{\tagwidth@}{\eqnshift@+\@emphEQ@rwd}%
            \ifdim\@emphEQ@templength>\tagwidth@
                \kern-\@emphEQ@templength
                \kern\eqnshift@
                \kern\@emphEQ@lwd
            \else
                \kern\@emphEQ@templength
                \kern-\eqnshift@
                \kern\@emphEQ@rwd
                \kern-\@emphEQ@lwd
            \fi
        \fi
    \fi
    \if@emphEQ@multline
        \if@emphEQ@NoNumEq
            % naught
        \else
            \if@emphEQ@FLL
                \kern-\@mathmargin
                \kern\tagwidth@
                \kern\multlinetaggap
            \fi
            \if@emphEQ@L
                \kern\tagwidth@
                \kern\multlinetaggap
                \kern-\multlinegap
            \fi
            \if@emphEQ@R
                \kern-\tagwidth@
                \kern-\multlinetaggap
```

```
                \kern\multlinegap
            \fi
        \fi
    \else
```

If we are using a `\shadowbox` in centered display mode, I would rather that the shadowsize is ignored when placing the box. Reason: I think it looks better. Judge for yourself:

$$a = b \tag{14}$$

$$\Phi(x) = \int_{-\infty}^{x} e^{-t^2/2}\,dt \tag{15}$$

$$\boxed{\begin{aligned} a &= b \qquad (16)\\ \Phi(x) &= \int_{-\infty}^{x} e^{-t^2/2}\,dt \qquad (17)\end{aligned}}$$

```
        \if@emphEQ@shadowbox
            \if@emphEQ@R
                \kern\shadowsize
            \fi
            \if@emphEQ@L
                \kern\shadowsize
            \fi
        \fi
    \fi
        \@emphEQ@boxtype{\usebox{\@emphEQ@Box}}%
```

After all is done, we empty out the added material and the boxtype.

```
        \gdef\@emphEQ@Left{}\gdef\@emphEQ@Right{}%
        \gdef\@emphEQ@boxtype{}%
    \]%
    \ifnum0=`{\fi}%
    \expandafter\end\expandafter{\@emphEQ@EnvName}%
}
```

3.7 Going public...

boxtype
Left
Right
empheq

In order to simplify the user interface, we make use of `keyval`. We define three keys to be used along with a more accessible version of @emphEQ@MainEnv.

```
\define@key{empheq}{boxtype}{\renewcommand\@emphEQ@boxtype{#1}}
\define@key{empheq}{Left}{\def\@emphEQ@Left{#1}}
\define@key{empheq}{Right}{\def\@emphEQ@Right{#1}}
\newenvironment{empheq}[1][]{\setkeys{empheq}{#1}\@emphEQ@MainEnv}%
                            {\end@emphEQ@MainEnv\ignorespacesafterend}%
```

\DeclareLeftDelimiter
\DeclareRightDelimiter

It might come in handy to have delimiters of the appropriate size available to cover the math display as in (1). Therefore we define two commands that will create such delimiters once and for all. Use this as an inspiration for further kinds of delimiters.

```
\newcommand{\DeclareLeftDelimiter}[2][\!]{%
    \@namedef{empheq\expandafter\@gobble\string#2}{
```

```
    \left#2
    \vphantom{\usebox{\@emphEQ@Box}}%
    \right.#1}}
\newcommand{\DeclareRightDelimiter}[2][\!]{%
    \@namedef{empheq\expandafter\@gobble\string#2}{%
        #1\left.
        \vphantom{\usebox{\@emphEQ@Box}}%
        \right#2}}%
\DeclareRightDelimiter{\rbrace}
\DeclareLeftDelimiter{\lbrace}
⟨/package⟩
```

4 Bugs and shortcomings

If you are to use align to create two alignment columns, then don't use it with empheq yet. Try with an alignat instead and all will be well. This will be fixed in upcoming versions as well as support for flalign though this is probably not the most sought after. `\intertext` shouldn't be used either.

Should you have any suggestions or bug reports then feel free to contact me at `moho01ab@student.cbs.dk`. Please put 'empheq' in the subject line.

Change History