

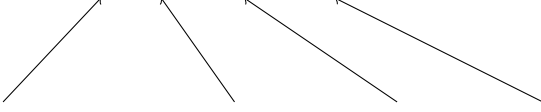
The wrapfig package*

Donald Arseneau†

Jan 31, 2003

Wrapfig.sty provides the environments `wrapfigure` and `wratable` for typesetting a narrow float at the edge of the text, and making the text wrap around it. The `wrapfigure` and `wratable` environments interact properly with the `\caption` command to produce proper numbering, but they are not regular floats like `figure` and `table`, so (beware!) they may be printed out of sequence with the regular floats. There are four parameters for `\begin{wrapfigure}`, two optional and two required, plus the text of the figure, with a caption perhaps:

```
\begin{wrapfigure}[12]{r}[34pt]{5cm} <figure> \end{wrapfigure}
```



[<number of narrow lines>] {<placement>} [<overhang>] {<width>}

Some idiosyncrasies:

- You must not specify a `wrapfigure` in any type of list environment or immediately before or immediately after one. It is OK to follow a list if there is a blank line (`\par`) in between.
- If you put a `wrapfigure` in a `parbox` or a `minipage`, or any other type of grouping, the text wrapping should end before the group does.
- It does work in two-column format, but are your figures that small?
- It may be out of sequence with regular floats.
- The hlines that may be printed above and below floats are ignored; you must insert them manually if desired.
- `\linewidth` is now adjusted within the wrapped text, but since it can only be set for whole paragraphs at a time, it will persist with the wrong value after the wrapping, until the paragraph is finished.

*This manual corresponds to `wrapfig.sty` v3.6, dated Jan 31, 2003.

†asnd@triumf.ca

New wrapping environments may be added when new float types are defined (using `memoir.cls`, `float.sty`, or `ccaption.sty`). Any wrapping environment, `wrapfigure`, `wraptable`, or something else may be invoked using the `wrapfloat` environment, as in `\begin{wrapfloat}{figure}{0}{5cm}`.

To use `float.sty` properly, load package `float` before `wrapfig`, and declare any new float types after loading both. Likewise for `ccaption.sty` and `\newfloatlist` and `memoir.cls` and its `\newfloat`.

1 Placement and Floating

Parameter #2 (required) is the figure placement code, but the valid codes are different from regular figures. They come in pairs: an uppercase version which allows the figure to float, and a lowercase version that puts the figure “exactly here”.

- `r` `R` – the right side of the text
- `l` `L` – the left side of the text
- `i` `I` – the inside edge—near the binding (if `[twoside]` document)
- `o` `O` – the outside edge—far from the binding

You should specify one code only, not a list. The figure or table must be on one side or the other; it cannot be in the middle with text on both sides. The `i` and `o` options refer to the inside and outside of the whole page, not individual columns.

The ability to float is somewhat restricted, and you will get best results by giving exact manual placement, but floating is more convenient while revising the document. Any changes to the formatting can ruin your manual positioning so you should adjust the placement just before printing a final copy. Here are some tips for good placement:

- The environment should be placed so as to not run over a page break.
- The environment must not be placed in special places like lists.
- For esthetic reasons, only plain text should wrap around the figure. Section titles and big equations look bad; lists are bad if the figure is on the left. (All these function properly, they just don’t look very good.) Small equations look fine.
- It is convenient to begin the environment between paragraphs, but if you want placement in the middle of a paragraph, you must put the environment between two words where there is a natural line break.

When floating, \LaTeX tries to apply these rules. More specifically, a floated wrapping environment will only begin...

- at the beginning of a paragraph,
- when there is enough room on the page, or it is possible to go on the next page,
- if the ‘paragraph’ is not in a section title or a list,

- if the paragraph is not wrapping around another figure,
- in the main text (not in a `minipage` etc.)

It is possible that a non-floating `wrapfigure` will be forced to float when an earlier one is still being processed. A warning will be written in that case. You can have more information about the floating process written to the log file by specifying `\usepackage[verbose]{wrapfig}`.

If there is a lot of flexibility on a page, a floating `wrapfigure` may be placed badly; you must turn to manual placement. A rare problem is that floats and footnotes specified within the wrapping text can also cause poor placement and bad formatting.

2 Sizing and optional overhang

Parameter #4 (the second required parameter) is the width of the figure or table. Given the way that \LaTeX puts just about everything into boxes with the current line-width, the width parameter will take precedence over whatever natural width the figure has. In particular, the caption is always typeset with the specified width. If the figure is wider than the space allotted, you will get an “overfull box” warning.

However, if you specify a width of *zero* (`0pt`), the actual width of the figure will determine the wrapping width. A following `\caption` should have the same width as the figure, but it might fail badly; it is safer to specify a width when you use a caption.

\LaTeX will wrap surrounding text around the figure, leaving a gap of `\intextsep` at the top and bottom, and `\columnsep` at the side, by producing a series of shortened text lines beside the figure. The indentation (shortening) of the text is the figure width plus `\columnsep` minus overhang (if any; see below).

\LaTeX calculates the number of short lines needed based on the height of the figure and the length `\intextsep`. You can override this guess by giving the first optional argument (parameter #1) specifying the number of shortened lines (counting each displayed equation as 3 lines). This is particularly useful when the surrounding text contains extra vertical spacing that is not accounted for automatically.

The second optional parameter (#3) tells how much the figure should hang out into the margin. The default overhang is given by the length `\wrapoverhang`, which is `0pt` normally but can be changed using `\setlength`. For example, to have all wrapfigures use the space reserved for marginal notes,

```
\setlength{\wrapoverhang}{\marginparwidth}
\addtolength{\wrapoverhang}{\marginparsep}
```

When you do specify the overhang explicitly for a particular figure, you can use a special unit called `\width` meaning the width of the figure. For example, `[0.5\width]` makes the center of the figure sit on the edge of the text, and `[\width]` puts the figure entirely in the margin (and the adjacent text is indented by just `\columnsep`). This `\width` is the actual width of the `wrapfigure`, which may be greater than the declared width.

3 Some Random Implementation Notes

Unfortunately, L^AT_EX's system of setting `\everypar` and `\par` is unable to coexist peacefully with a wrapping environment, so I was forced to subvert the `\@setpar` mechanism and `\everypar`. (`\everypar` is already subverted once by NFSS.)

When checking the room left on the page, remember that if there is less than `\baselineskip` the new paragraph will begin on the next page, even if there is no page stretch. If non-floating, I force a bad page break rather than have the figure hang into the bottom margin.

Here are notes on various variables and some macros; what info they store and how they are used.

`\WF@wli` number-of-wrapped-lines parameter, saved for start of wrapping. Set globally by `\WF@wr` (set empty if no optional parameter given). The floating mechanism ignores this and uses the real size.

`\WF@ovh` margin overhang set globally by `\WF@rapt`, saved until placing figure (but not reset). Actually, the setting is very tricky so that the expected values are used when a figure floats. First, the expression is saved without evaluation by `\WF@rapt` (`\begin{wrapfigure}`) because `\width` is still unknown. Soon after that, `\endwrapfigure` executes `\WF@ovh` to evaluate the overhang and save the result (so that changes to `\wrapoverhang` while this figure is floating won't affect this figure). Finally, it is used by `\WF@putfigmaybe` when printing the fig.

`\WF@place` a macro that is used as a number, giving the placement code. It might start out as ``I` and later be converted to `114 (r)`.

`\WF@box` tested for void at `\begin{wrapfigure}`, to avoid collisions, by `\everypar` to do floating, and by `\WF@finale` before resetting `\everypar`. Voided globally when used by `\WF@putfigmaybe` (or by `\WF@wr` if an old figure must be dumped prematurely).

`\par` test if it is `\@@par` by `\begin{wrapfigure}` and `\WF@floathand` to float past special environments. It is set to `\@par` (`\WF@mypar`) by `\WF@startwrapping`, and restored by an end-group (bad!) or by `\WF@finale` (good). It is protected from change by redefining `\@setpar`.

`\parshape` let to `\WF@fudgeparshape` by `\WF@startwrapping`, so lists will continue wrapping; `\@@parshape` preserves the real `\parshape` command, and it is restored by `\WF@finale` or `\@parboxrestore`. `\WF@floathand` and `\WF@wr` test if old wrapping is still in progress with `\ifx\parshape\WF@fudgeparshape`. The value of `\@@parshape` is also tested to float past lists and other wrapping environments.

`\hangindent` tested to float past section titles etc.

`\c@WF@wrappedlines` the number of shortened lines + 1; set globally by `\WF@startwrapping` and decremented by `\par` (`\WF@mypar`). It is `> 1` only when wrapping is

incomplete. `\WF@wrapand`, `\WF@fudgeparshape`, and `\WF@mypar` test the number for calling `\WF@finale`. It may get stuck at some high value if `\par` is restored by an end-group, (and wrapping is terminated prematurely) so it is unwise to use this as a test for wrapping-complete.

`\pagetotal` one of many parameters used to compute floating. When putting a `wrapfigure` in a `parbox`, I assign `\let\pagetotal\maxdimen` (locally!) to signal not-top-of-page and no floating.

`\WF@pspars` the `\parshape` parameters as L^AT_EX sets them for lists (`\WF@fudgeparshape`); when wrapping I test it and use it to modify my own real params for the paragraph. They are also used when `\parshape` is restored after wrapping.

`\WF@finale` is performed by `\par` when wrapping should end. However, that might happen inside a group (a list especially), so the subverted versions of `\par`, `\parshape` etc. will be reinstated when the group ends. Thus, they must themselves test `\c@WF@wrappedlines < 2` to see when wrapping is over, and if so, they should just do `\WF@finale` again.

These are the tests to see if a floating `wrapfigure` will fit at a particular spot. These tests are performed at the beginning of every paragraph after the figure, except in lists etc. (`\pagegoal - \pagetotal` is the room left on the page.)

```
>
room_left := \pagegoal - \pagetotal
if room_left < 0 then page overfull already:
  put figure (on next page)
else
  if figure_size > room_left then does not fit
    if max(min_stretch, \pagestretch) + extra > room_left
      then page can stretch until full:
        put figure (at top of next page)
    fi
  else figure fits: put figure
fi fi
<
```

Even if a `wrapfigure` is not floating, it will go through the same logic to generate a `\pagebreak`, and maybe an underfull page, when the current page can stretch until full. The `min_stretch` depends on whether it is floating or not: `.5\baselineskip` (floating) `2\baselineskip` (not). The `extra` is `.5\baselineskip` in either case. These values can be adjusted.

There are some other ‘magic numbers’ for floating that aren’t really so special, but you must change them together if you change them at all. To make floating wrapfigures float less and fit on pages more frequently, but not change the number of wrapped lines, decrease the `1.5` in `\global\advance\WF@size1.5\baselineskip` and increase the `1.1`

in `\advance\WF@size1.1\baselineskip` by the same amount (and vice versa). To make more (or fewer) wrapped lines for the same size figure, without changing the floating, change 1.1 in `\advance\WF@size1.1\baselineskip` unilaterally.