Using EPS Graphics in LaTeX 2ε Documents Part 2: Floating figures, boxed figures, captions, and math in figures

Keith Reckdahl

Abstract

This is the second of two papers that explain how to use Encapsulated PostScript (EPS) files in LaTeX 2ε documents.

The first paper in the series, which appeared in *TUGboat* **17** (1), covered

- the graphics and graphicx packages, which provide commands to insert, scale, and rotate EPS graphics,
- commands which are commonly used in conjunction with EPS graphics,
- use of `dvips` to insert compressed EPS files and non-EPS graphic formats (TIFF, GIF, JPEG, PICT, etc.)
- software for decompression or graphics conversion capabilities, which must be provided by the user.

The present paper covers

- floating figures in various configurations (such as more than one figure in a single float), and the use of the subfigure package,
- creation of boxed figures, by use of the `\fbox` command, or of the facilities of the fancybox package,
- manipulation of the caption of a figure, including use of the caption2 package, and
- modifying the text within an EPS file by using the PSfrag system, for example to include mathematical symbols or equations.

1 The `figure` Environment

Graphics can be inserted as part of a LaTeX `figure` environment, which allows the graphics to float for better formatting, especially for large graphics. The `figure` environment also makes it easy to reference the graphic. The commands

```
\begin{figure}[htb]
  \centering
  \includegraphics[totalheight=2in]{graph.eps}
  \caption{This is an inserted \EPS{} graphic}
  \label{fig:graph}
\end{figure}

The graph in Figure~\ref{fig:graph} is
from an \EPS{} file generated by gnuplot.
```

insert the graphic in a figure and place a caption under the graphic. The optional `\label` command specifies a label which is used by the `\ref` command to reference the figure (the `\label` command must be *after* the `\caption` command). Note that the figure environment can only be used in *outer paragraph mode* and thus cannot be used inside any box (such as `\parbox` or `minipage`).

1.1 Caption Vertical Spacing

While the figure caption is usually placed below the graphic, it can be placed above the graphic simply by placing the `\caption` command before the graphics-inclusion command. For example, the commands

```
\begin{figure}[htb]
  \centering
  \caption{Caption Above Graphic}
  \includegraphics[width=1in]{box.eps}
\end{figure}
```

produce Figure 1.

Figure 1: Caption Above Graphic

Since captions are generally placed below the graphic, LaTeX places more vertical spacing above the caption than below it. As a result, the caption

in Figure 1 is placed quite close to the graphic. The spacing above and below the caption is controlled by the two lengths `\abovecaptionskip` (which is `10pt` by default) and `\belowcaptionskip` (which is zero by default). The standard LaTeX commands `\setlength` and `\addtolength` are used to modify these lengths. The commands

```
\setlength{\abovecaptionskip}{5pt}
\setlength{\belowcaptionskip}{0.5cm}
```

provides a 5 point spacing above the caption and a 0.5 centimeter spacing below the caption. The commands

```
\addtolength{\abovecaptionskip}{5pt}
\addtolength{\belowcaptionskip}{-5pt}
```

increases the spacing above the caption by 5 points and decreases the spacing below the caption by 5 points. For example, the commands

```
\begin{figure}[htb]
  \setlength{\belowcaptionskip}{10pt}
  \centering
  \caption{Caption Above Graphic}
  \includegraphics[width=1in]{box.eps}
\end{figure}
```

produce Figure 2.

Figure 2: Caption Above Graphic

1.2 Figure Placement Options

LaTeX figures are "floats" whose placement is decided by LaTeX. Since your taste in figure-placement may differ from that of LaTeX, the `figure` environment has placement options

`h` *Here:* Place the figure in the text where the figure command is located.

`t` *Top:* Place the figure at the top of a page.

`b` *Bottom:* Place the figure at the bottom of a page.

`p` *Page of Floats:* Place the figure on a separate page which contains only floats.

The placement options in the above example are `[htb]` which means that LaTeX first tries to place the figure at that location, then tries to place the figure at the top of a page, and finally tries to place the figure at the bottom of a page. When LaTeX "tries" to place a figure, it checks how many figures are already on the page and other esthetic concerns. If LaTeX determines that the figure wouldn't look good, it tries the next placement option.

The order in which the placement options are specified does *not* make any difference. The placement options are attempted in the order `h-t-b-p` regardless of the order in which the options are specified. Thus `[hb]` and `[bh]` are both attempted as `h-b`.

To make LaTeX "try really hard" in its float placement, put an exclamation point in the placement options (e.g., `\begin{figure}[!ht]`) which makes LaTeX suspend its esthetic rules and do its best to make the requested placement. Even with the `!` option, LaTeX has the final say in the placement and reserves the right to override the request. For example, if the commands

```
\begin{figure}[!ht]
  \includegraphics[totalheight=4in]{graph.eps}
\end{figure}
```

occur 3 inches from the bottom of the page, LaTeX objects to leaving 3 inches of whitespace at the bottom of the page and overrides the `[!h]`, filling the bottom 3 inches of the page with the text which is after the figure in the .tex file.

If you feel LaTeX is making poor float placement decisions, you may need to tweak its placement algorithm by modifying the float parameters (see [1, pages 199-200], [2, pages 141-143], or [3, pages 174-175]).

1.2.1 The float Package's `[H]` Placement Option

The float package adds an `[H]` option to the `figure` environment which *always* places the float "here". However, this option should normally be avoided, as the `[!ht]` option is a better way of producing the desired behavior.

To use the `[H]` option, include

```
\usepackage{float}
```

in the preamble and put a `\restylefloat{figure}` command *before* the `\begin{figure}[H]` command is used. (See [2, page 149].) When using the `[H]` option, the user is responsible for managing the document to avoid large sections of whitespace.

While the figure environment defined by the float package allows the `[H]` option, it also places the figure caption below the figure environment. While this does not affect simple figures, it prevents captions above graphics as in Figure 1 or the construction of side-by-side and other complex figure arrangements.

2 Landscape Figures

In a document with portrait orientation, there are three methods for producing figures with landscape orientation.

1. The lscape package provides a `landscape` environment, which treats the left edge of the paper as the top of the page, causing any text, tables, or figures in the `landscape` environment to have landscape orientation.
2. The rotating package has a `sidewaysfigure` environment which is similar to the `figure` environment except that the figures have landscape orientation.
3. The rotating package provides a `\rotcaption` command which is like the `\caption` command except that the caption has landscape orientation.

The differences between methods are as follows:

- Both options 1 and 2 place the rotated figure on a separate page. Option 3 produces an individual float which need not be on its own page.
- The full-page figure produced by Option 2 will float to provide better document formatting. Since the figure(s) produced by Option 1 can only float within the landscape pages, this may result in a partially-empty page before the figure.
- The `landscape` environment in Option 1 can be used to produce landscape pages containing any combination of text, tables, and figures. Option 2 produces only rotated figures or tables.

2.1 The `landscape` Environment

The lscape package (part of the standard "graphics bundle" distributed with LaTeX) defines the `landscape` environment; this lets you place landscape pages in a portrait document. The landscape pages are rotated such that the left edge of the portrait page is the top edge of the landscape page.

Entering `\begin{landscape}` generates a `\clearpage` command which prints all unprocessed portrait floats, before switching to landscape orientation. Likewise, `\end{landscape}` prints all unprocessed landscape floats before switching back to portrait orientation.

The entire contents of the `landscape` environment is typeset with landscape orientation. This may include any mixture of text, figures, and tables. If the landscape environment contains only a figure environment

```
\begin{landscape}
 \begin{figure}
  \centering
  \includegraphics[width=4in]{box.eps}
  \caption{Landscape Figure}
 \end{figure}
\end{landscape}
```

the `landscape` environment produces a landscape figure. Note that since the `landscape` environment starts a new page, it may result in a partially-blank page.

2.2 The `sidewaysfigure` Environment

The rotating package provides the `sidewaysfigure` environment which produces figures with landscape orientation. For example

```
\begin{sidewaysfigure}
 \centering
 \includegraphics[width=4in]{box.eps}
 \caption{Sidewaysfigure Figure}
\end{sidewaysfigure}
```

produces Figure 3.

Unlike the `landscape` environment, the figure produced by `sidewaysfigure` can float within the portrait pages to avoid the partially-blank page that the `landscape` environment may produce. However, the `landscape` environment is much more flexible, allowing the landscape pages to consist of a mixture of text, tables, and figures. The rotating package also provides a `sidewaystable` environment for producing tables with landscape orientation. Unlike the `landscape` environments, the `sidewaystable` and `sidewaysfigure` environments cannot contain a mixture of text, figures, and tables.

The default orientation of the figures produced by `sidewaysfigure` depends on whether the document is processed with the `oneside` or `twoside` documentclass option

- When the `oneside` option is chosen, the bottom of the graphic is towards the right edge of the portrait page.
- When the `twoside` option is chosen, the bottom of the graphic is towards the outside edge of the portrait page.

This default behavior can be overridden by options to the `\usepackage{rotating}` command.

```
\usepackage[rotateleft]{rotating}
```

causes the bottom of the `sidewaysfigure` graphics to be towards the left edge of the portrait page (regardless of `oneside` or `twoside` options). Similarly,

```
\usepackage[rotateright]{rotating}
```

causes the bottom of the `sidewaysfigure` graphics to be towards the right edge of the portrait page.

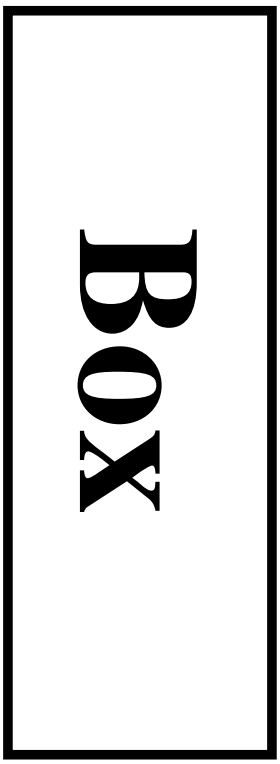

Figure 3: Sidewaysfigure Figure

2.3 The \rotcaption command

The methods in Sections 2.1 and 2.2 both produce full-page landscape figures, which may not be necessary for smaller landscape figures. The rotating package's `\rotcaption` command can be used to construct smaller landscape figures. For example

```
\begin{figure}
 \centering
 \begin{minipage}[c]{1in}
  \includegraphics[angle=90,
   width=\textwidth]{box.eps}
 \end{minipage}
 \begin{minipage}[c]{0.5in}
  \rotcaption{Rotcaption Caption}
  \label{fig:rotcaption}
 \end{minipage}
\end{figure}
```

produces Figure 4.

The caption produced by `\rotcaption` is always rotated such that its bottom is towards the right edge of the paper. Unlike the methods in Sections 2.1 and 2.2, the `\rotcaption` command does not rotate the graphics. We therefore added the `angle=90` option in the above example.

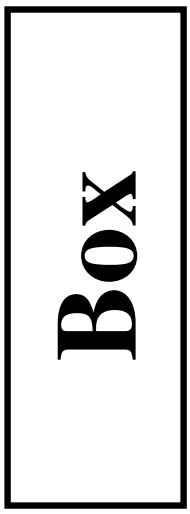

Figure 4: Rotcaption Caption

3 Side-by-Side Graphics

The commands necessary for side-by-side graphics depend on how the user wants the graphics organized. This section covers three common methods of organizing side-by-side graphics

1. The side-by-side graphics are combined into a single figure.
2. The side-by-side graphics each form their own figure (e.g., Figure 12, Figure 13, etc.)

3. The side-by-side graphics each form a subfigure (e.g., Figure 12a, Figure 12b, etc.) of a single figure (Figure 12).

While this section specifically discusses side-by-side graphics, most of the information is also valid for vertically-stacked graphics and complex figures such as Figures 28-34 on Page 297.

3.1 Side-by-Side Graphics in a Single Figure

The two most common methods for placing side-by-side graphics in a figure are

1. Multiple `\includegraphics` commands
2. Multiple `minipage` environments, each of which contains an `\includegraphics` command

3.1.1 Side-by-Side `\includegraphics` Commands

While spacing side-by-side graphics in a figure can be as simple as

```
\begin{figure}
  \centering
  \includegraphics[width=1in]{file1.eps}
  \includegraphics[width=2in]{file2.eps}
  \caption{Two Graphics in One Figure}
\end{figure}
```

there are usually horizontal-spacing commands such as `\hspace{1in}` or `\hfill` between the `\includegraphics` commands. For example,

```
\begin{figure}
 \centering
 \includegraphics[width=1in]{box.eps}%
 \hspace{1in}%
 \includegraphics[width=2in]{box.eps}
 \caption{Two Graphics in One Figure}
\end{figure}
```

produces Figure 5 which is 4 inches wide (1 inch for `file1.eps`, 1 inch for the `\hspace`, and 2 inches for `file2.eps`). This 4-inch-wide figure is centered on the page. If `\hfill` is used instead of `\hspace`, the graphics are pushed to the margins.

3.1.2 Side-by-Side `minipage` Environments

Greater control over the graphics' horizontal and vertical placement can be obtained by placing the commands inside `minipage` environments. For example,

```
\begin{figure}
 \centering
 \begin{minipage}[c]{0.5\textwidth}
  \centering
  \includegraphics[width=1in]{box.eps}
 \end{minipage}%
 \begin{minipage}[c]{0.5\textwidth}
  \centering
  \includegraphics[width=2in]{box.eps}
 \end{minipage}
 \caption{Centers Aligned Vertically}
\end{figure}
```

produces Figure 6.
Some notes on this example:

- Like any other LaTeX object, minipages are positioned such that their baseline is aligned with the current baseline. The minipage `[c]` option defines the minipage's baseline as its centerline. The `[b]` option defines the minipage's baseline as the baseline of the bottom line of the minipage (which is not necessarily the bottom of the minipage). The `[t]` option defines the minipage's baseline as the baseline of the top line of the minipage (which is not necessarily the top of the minipage). See section 4 for information on the `minipage` environment and its placement options.
- The `%` after the first `\end{minipage}` command prevents a space from being inserted between the `minipage` boxes. Such a space would use some horizontal space, preventing both minipages from fitting on the same line.

When the widths of the minipages do not add up to `1.0\textwidth`, the `\hspace` or `\hfill` commands can be used to specify to horizontal spacing. For example,

```
\begin{figure}
 \centering
 \begin{minipage}[c]{1in}
   \centering
   \includegraphics[width=\textwidth]{box.eps}
 \end{minipage}%
 \hspace{1in}%
 \begin{minipage}[c]{2in}
   \centering
   \includegraphics[width=\textwidth]{box.eps}
 \end{minipage}
 \caption{Centers Aligned Vertically}
\end{figure}
```

produces a figure with the same horizontal spacing as Figure 5, but the centers of the boxes are aligned vertically.

3.2 Side-by-Side Figures

In the previous section, multiple `minipage` environments were used inside a `figure` environment to produce a single figure consisting of multiple graphics. Placing `\caption` statements inside the minipages makes the minipages themselves become figures. For example,

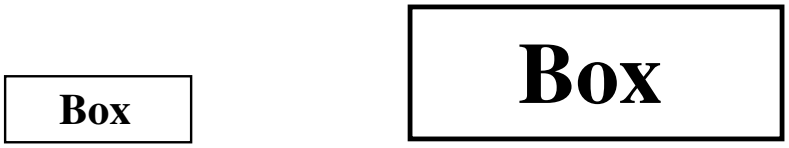

Figure 5: Two Graphics in One Figure

Figure 6: Centers Aligned Vertically

```
\begin{figure}
 \begin{minipage}[b]{0.5\linewidth}
  \centering
  \includegraphics[width=1in]{box.eps}
  \caption{Small Box}  \label{fig:side:a}
 \end{minipage}%
 \begin{minipage}[b]{0.5\linewidth}
  \centering
  \includegraphics[width=1.5in]{box.eps}
  \caption{Big Box}  \label{fig:side:b}
 \end{minipage}
\end{figure}
```

produces Figures 7 and 8.

Although the above commands include *one* figure environment, the commands produce *two* figures. Since the `\caption` command actually produces the figure, figure environments with multiple `\caption` commands produce multiple figures.

3.2.1 Alignment Problems with Side-by-Side Figures

The `[b]` options aligned the bottoms of Figures 7 and 8. However, long captions may affect this alignment. For example,

```
\begin{figure}
 \begin{minipage}[b]{.333\linewidth}
  \centering
  \includegraphics[width=1in]{box.eps}
  \caption{Small Box with a Long Caption}
  \label{fig:side:c}
 \end{minipage}%
 \begin{minipage}[b]{.333\linewidth}
  \centering
  \includegraphics[width=1.5in]{box.eps}
  \caption{Medium Box}  \label{fig:side:d}
 \end{minipage}%
 \begin{minipage}[b]{.333\linewidth}
  \centering
  \includegraphics[width=2.0in]{box.eps}
  \caption{Big Box}  \label{fig:side:e}
 \end{minipage}
\end{figure}
```

produces Figures 9, 10, and 11.

The long caption of Figure 9 means that it is not aligned with the other figures. In this case, the baselines of all the figures are their bottoms, so the alignment can be corrected by changing the `minipage` positioning option from `[b]` to `[t]` which aligns the baselines of the graphics (see Section 4 for information). If the baselines of the graphics do not correspond to their bottoms, the `[t]` option does not produce the desired positioning. Instead, invisible vertical lines (called *struts*) can be placed in the captions of the other figures to make LaTeX think that all the captions are two lines long.

```
\begin{figure}
 \begin{minipage}[b]{.333\linewidth}
  \centering
  \includegraphics[width=1in]{box.eps}
  \caption{Small Box with a Long Caption}
  \label{fig:side:cc}
 \end{minipage}%
 \begin{minipage}[b]{.333\linewidth}
  \centering
  \includegraphics[width=1.5in]{box.eps}
  \caption[Medium Box]
   {Medium Box
       \protect\rule[-\baselineskip]{0pt}
                           {2\baselineskip}}
  \label{fig:side:dd}
 \end{minipage}%
 \begin{minipage}[b]{.333\linewidth}
  \centering
  \includegraphics[width=2.0in]{box.eps}
  \caption[Big Box]
    {Big Box \protect\rule[-\baselineskip]{0pt}
                           {2\baselineskip}}
  \label{fig:side:ee}
```

Figure 7: Small Box

Figure 8: Big Box

Figure 9: Small Box with a Long Caption

Figure 10: Medium Box

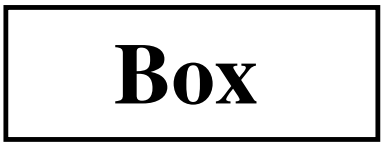

Figure 11: Big Box

```
 \end{minipage}
\end{figure}
```

which produces Figures 12, 13, and 14.

`\rule[`*start*`]{`*width*`}{`*height*`}` produces a vertical line with a width of *width* starting *start* above the baseline and with a height *height*. When the width is zero, the line becomes invisible and is called a *strut*. In the above captions, the strut

```
\rule[-\baselineskip]{0pt}{2\baselineskip}
```

starts one line below the baseline and continues to the top of the current line. This makes LaTeX think that, like the Figure 12 caption, the captions for Figures 13 and 14 are two lines tall. Since the `\rule` command is fragile, the `\protect` command must be used so that `\rule` can be used inside the `\caption` command. The `\caption[Big Box]` option specifies that the text "Big Box" should be used in the list of figures (where the extra vertical space is not desired).

3.3 Side-by-Side Subfigures

It is often desirable to refer to side-by-side graphics both individually and as a group. The `\subfigure` command (from the subfigure package) defines the group of side-by-side graphics as a single figure and defines each graphics as a subfigure. For example,

```
\begin{figure}
 \centering
 \subfigure[Small Box with a Long Caption]%
  \label{fig:subfig:a} %% first subfigure
  \includegraphics[width=1.0in]{box.eps}}%
 \hspace{1in}%
 \subfigure[Big Box]{
  \label{fig:subfig:b} %% second subfigure
  \includegraphics[width=1.5in]{box.eps}}
 \caption{Two Subfigures}
 \label{fig:subfig}    %% entire figure
\end{figure}
```

produces Figure 15; label `{fig:subfig:a}` refers to subfigure 15(a), label `{fig:subfig:b}` refers to subfigure 15(b), and label `{fig:subfig}` refers to Figure 15.

3.3.1 Subfigures Inside `minipage` Environments

Like other side-by-side graphics, subfigures are often put inside `minipage` environments. For example,

```
\begin{figure}
    \centering
    \begin{minipage}[b]{0.5\textwidth}
 \centering
 \subfigure[Small Box with a Long Caption]{
 \label{fig:subfig:mini:a}
 \includegraphics[width=1.0in]{box.eps}}

 \end{minipage}%
    \begin{minipage}[b]{0.5\textwidth}
 \centering
 \subfigure[Big Box]{
 \label{fig:subfig:mini:b}
 \includegraphics[width=1.5in]{box.eps}}

 \end{minipage}
    \caption{Subfigures Inside Minipages}
    \label{fig:subfig:mini}
\end{figure}
```

produces Figure 16 which contains subfigures 16(a) and 16(b).

3.3.2 `minipage` Environments Inside Subfigures

Since Subfigure 16(a) does not contain anything except the `\includegraphics` command, the caption in subfigure 16(a) is only as wide as the included graphic. If the subfigure instead consists of the entire minipage, the caption is made as wide as the minipage. For example,

Figure 12: Small Box with a Long Caption

Figure 13: Medium Box

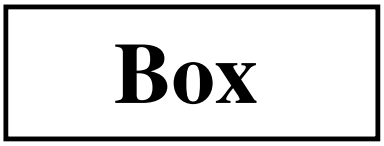

Figure 14: Big Box

(a) Small Box with a Long Caption

(b) Big Box

Figure 15: Two Subfigures

(a) Small Box with a Long Caption

(b) Big Box

Figure 16: Subfigures Inside Minipages

```
\begin{figure}
\subfigure[Small Box with  a Long
   Caption]{\label{fig:mini:subfig:a}
 \begin{minipage}[b]{0.5\textwidth}
   \centering
   \includegraphics[width=1in]{box.eps}
 \end{minipage}}%
\subfigure[Big Box]{
\label{fig:mini:subfig:b}
 \begin{minipage}[b]{0.5\textwidth}
   \centering
   \includegraphics[width=1.3in]{box.eps}
 \end{minipage}}
\caption{Minipages Inside Subfigures}
\label{fig:mini:subfig}
\end{figure}
```

produces Figure 17. Note that the caption of subfigure 17(a) is considerably wider than that of subfigure 16(a).

3.3.3 Changing Subfigure Numbering

The subfigure labels have two forms:

1. One which appears under the subfigure as part of the caption, produced by `\@thesubfigure`.
2. One which appears when the `\ref` command is used, produced by concatenating the output of `\p@subfigure` to the output `\thesubfigure`.

These commands use the `subfigure` counter and the `\thefigure` command, making the subfigure label formatting be controlled by the following commands:

- `\thefigure` prints the current figure number.
- The counter `subfigure` counts the subfigures. `\alph{subfigure}` prints the value of the `subfigure` counter in lowercase letters. `\roman{subfigure}` prints the value of the `subfigure` counter in lowercase Roman numerals. (See [1, page 98] or [2, page 446] for a list of counter output commands.)
- `\thesubfigure` is `(\alph{subfigure})` by default, which produces (a), (b), etc.
- `\@thesubfigure` `\thesubfigure\space` by default, which adds a space between the caption label and the caption.

(a) Small Box with a Long Caption

(b) Big Box

Figure 17: Minipages Inside Subfigures

- `\p@subfigure` is `\thefigure` by default

These commands make the default caption labels (a), (b), etc. and the default `\ref` labels 12(a), 12(b), etc. See [10] for controlling the size and font of the subfigure labels.

3.3.4 Subfigure Examples

1. To make the caption labels (i), (ii), etc. and make the `\ref` labels 12i, 12ii, etc., enter the following commands (preferably in the LaTeX file's preamble):

```
\renewcommand{\thesubfigure}
   {\roman{subfigure}}
\makeatletter
\renewcommand{\@thesubfigure}
   {(\thesubfigure)\space}
\renewcommand{\p@subfigure}
   {\thefigure}
\makeatother
```

   `\makeatletter` and `\makeatother` protect the `@` signs in the `\renewcommand` statements.

2. To make the caption labels 12.1:, 12.2:, etc. and make the `\ref` labels 12.1, 12.2, etc., enter the following commands:

```
\renewcommand{\thesubfigure}
  {\thefigure.\arabic{subfigure}}
\makeatletter
\renewcommand{\@thesubfigure}
  {\thesubfigure:\space}
\renewcommand{\p@subfigure}{}
\makeatother
```

3.3.5 Adding Subfigures to List of Figures

The List of Figures generated by `\listoffigures` includes only figures by default, *not* subfigures. To add the subfigures to the List of Figures, type

```
\setcounter{lofdepth}{2}
```

before the `\listoffigures` command.

4 Minipage Placement Option Details

The manner in which `minipage` environments are vertically aligned may be confusing. For example, one might think the commands

```
\begin{figure}
 \centering
 \begin{minipage}[b]{.25\textwidth}
  \centering
  \includegraphics[width=1in]{box.eps}
 \end{minipage}
 \begin{minipage}[b]{.25\textwidth}
  \centering
  \includegraphics[width=1in,angle=-90]{box.eps}
 \end{minipage}
 \caption{\texttt{minipage} with
    \texttt{[b]} option}
\end{figure}
```

which use the minipage `[b]` option would align the bottoms of the graphics. Instead they produce Figure 18.

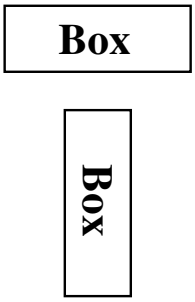

Figure 18: `minipage` with `[b]` option

Similarly, one might think the commands

```
\begin{figure}
 \centering
 \begin{minipage}[t]{.25\textwidth}
  \centering
  \includegraphics[width=1in]{box.eps}
 \end{minipage}
 \begin{minipage}[t]{.25\textwidth}
  \centering
  \includegraphics[width=1in,
            angle=-90]{box.eps}
 \end{minipage}
 \caption{\texttt{minipage} with
    \texttt{[t]} option}
\end{figure}
```

which use the minipage [t] options would align the tops of the graphics. Instead they produce a figure which is *exactly* the same as Figure 18.

The [b] and [t] options produce the same figure because the minipage environment's [b] option does *not* align the bottoms of the minipages. Rather, it aligns the baselines of the minipages' bottom lines. Similarly, the [t] option aligns the baselines of the minipages' top lines. Since the minipages in the above examples only have one line, the [t] and [b] use the same line for alignment. In this case, the reference point of the minipage is the reference point (original lower-left corner) of the EPS graphic.

4.1 Aligning the Bottoms of Minipages

One method for aligning the bottoms of minipages is to make the bottom of the minipage be the baseline of the minipage. If a line with zero height and zero depth is added inside the minipage after the graphics then the [b] option makes the bottom of the minipage be the minipage's baseline. The command \par\vspace{0pt} creates such a zero-height, zero-depth line. Since the baseline of this zero-depth line is the bottom of the minipage, the [b] option now aligns the bottom of the minipage. For example,

```
\begin{figure}
 \centering
 \begin{minipage}[b]{.25\textwidth}
  \centering
  \includegraphics[width=1in]{box.eps}
  \par\vspace{0pt}
 \end{minipage}
 \begin{minipage}[b]{.25\textwidth}
  \centering
  \includegraphics[width=1in,
      angle=-90]{box.eps}
  \par\vspace{0pt}
 \end{minipage}
 \caption{Minipages with Bottoms Aligned}
\end{figure}
```

produces Figure 19.

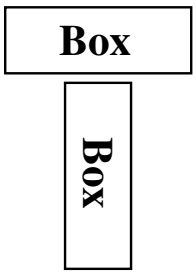

Figure 19: Minipages with Bottoms Aligned

4.2 Aligning the Tops of Minipages

To align the tops of the minipages, one must add a zero-height, zero-depth line to the top of the minipage. Then the [t] option makes the top of the minipage be the baseline of the minipage. Preceding \includegraphics command by \vspace{0pt} inserts a zero-height, zero-depth line above the graphic. Since the baseline of this zero-height line is the top of the minipage, the [t] option now aligns the top of the minipage. For example,

```
\begin{figure}
 \centering
 \begin{minipage}[t]{.25\textwidth}
  \vspace{0pt}
  \centering
  \includegraphics[width=1in]{box.eps}
 \end{minipage}
 \begin{minipage}[t]{.25\textwidth}
  \vspace{0pt}
  \centering
  \includegraphics[width=1in,angle=-90]{box.eps}
 \end{minipage}
 \caption{Minipages with Tops Aligned}
\end{figure}
```

produces Figure 20.

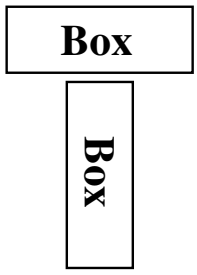

Figure 20: Minipages with Tops Aligned

This aligns the tops of the minipages with the current baseline. If you prefer to align the tops of the minipages with the top of the current line of text, use \vspace{-\baselineskip} instead of \vspace{0pt}. This topic is discussed in [2, pages 456-457].

5 Boxed Figures

The term *Boxed Figure* usually refers to one of two situations:

- A box surrounds the figure's graphic but not the figure's caption.
- A box surrounds the figure's graphic and its caption.

The basic method for boxing an item is to simply place the item inside an \fbox command, which

surrounds the object with a rectangular box. The fancybox package provides boxes of different styles.

5.1 Box Around Graphic

Placing `\includegraphics` inside an `\fbox` command produces a box around the included graphic. For example, the commands

```
\begin{figure}
 \centering
 \fbox{\includegraphics
        [totalheight=2in]{file.eps}}
 \caption{Box Around Graphic,
      But Not Around Caption}
 \label{fig:boxed_graphic}
\end{figure}
```

place a box around the included figure, as shown in Figure 21.

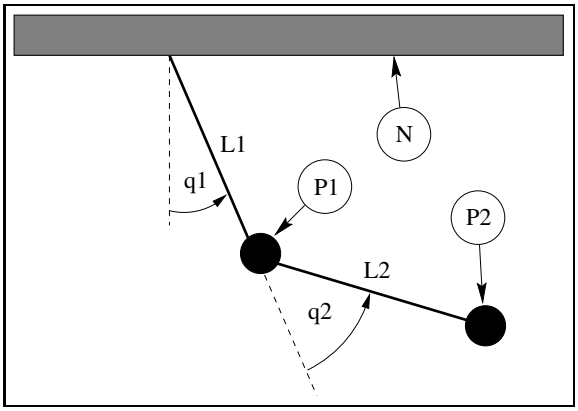

Figure 21: Box Around Graphic, But Not Around Caption

5.2 Box Around Figure and Caption

To include both the figure's graphic and its caption, one may be tempted to move the `\caption` command inside the `\fbox` command. However, this does not work because `\caption` can only be used in paragraph mode, while the contents of an `\fbox` command are processed in LR mode. (LaTeX uses three modes: LR mode, paragraph mode, and math mode. See [1, pages 36,103-5] for an explanation.)

Since the contents of minipage environments and `\parbox` commands are processed in paragraph mode, the `\caption` command can be included in the `\fbox` by enclosing the `\fbox` contents inside a minipage environment or a `\parbox` command. Since both minipages and parboxes require a width specification, there is no direct way to make the `\fbox` exactly as wide as the graphic and caption.

For example, the commands

```
\begin{figure}
 \centering
 \fbox{
  \begin{minipage}{3in}
  \centering
  \includegraphics
       [totalheight=2in]{pend.eps}
  \caption{Box Around Figure
         Graphic and Caption}
  \label{fig:boxed_figure}
 \end{minipage}
 }
\end{figure}
```

place a box around the figure's graphic and caption, as shown in Figure 22.

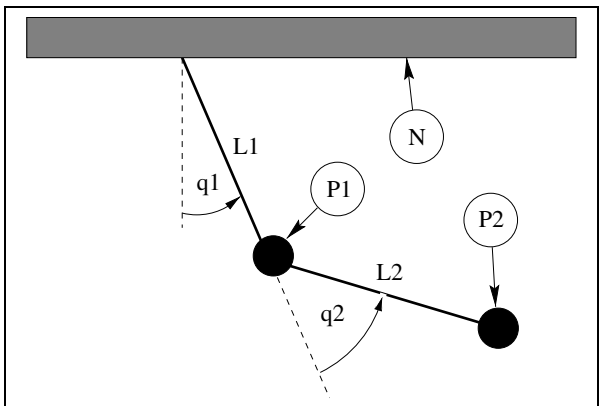

Figure 22: Box Around Figure Graphic and Caption

The determination of a proper minipage width is usually a trial-and-error process. If the caption is wider than the graphic, the minipage can be made as wide as the caption by estimating the caption width with a `\settowidth` command:

```
\begin{figure}
 \centering
 \newlength{\mylength}
 \settowidth{\mylength}
   {Figure XX: Box Around
     Figure Graphic and Caption}
 \fbox{
 \begin{minipage}{\mylength}
  \centering
  \includegraphics[totalheight=2in]{pend.eps}
  \caption{Box Around Figure Graphic
     and Caption}
   \label{fig:boxed_figure_length}
 \end{minipage}
 }
\end{figure}
```

5.3 Customizing \fbox Parameters

In Figures 21 and 22, the box is constructed of 0.4 pt thick lines with a 3 pt space between the box and the graphic. These two dimensions can be customized by setting the LaTeX length variables \fboxrule and \fboxsep, respectively, with the \setlength command. For example, the commands

```
\begin{figure}
 \centering
 \setlength{\fboxrule}{3pt}
 \setlength{\fboxsep}{1cm}
 \fbox{\includegraphics
       [totalheight=1.5in]{pend.eps}}
 \caption{Graphic with Customized Box}
 \label{fig:boxed_custom}
\end{figure}
```

place a box with 3 pt thick lines which is separated from the graphic by 1 centimeter, as shown in Figure 23.

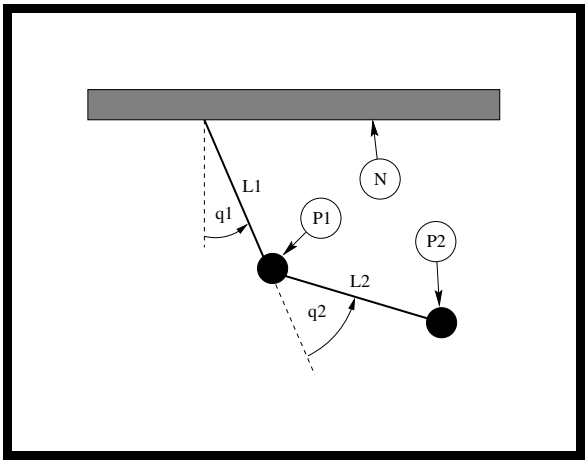

Figure 23: Graphic with Customized Box

5.4 The fancybox Package

In Figures 21, 22, and 23, the \fbox command was used to place standard rectangular boxes around the figures. Alternatively, you can use the fancybox package, which provides the commands \shadowbox, \doublebox, \ovalbox, and \Ovalbox to produce other types of boxes. Details of the commands are given in Table 1.

Like \fbox, the separation between these boxes and their contents is controlled by the LaTeX length \fboxsep. The length \shadowsize is set with the \setlength command, as was done for \fboxrule and \fboxsep in section 5.3. The lines for \ovalbox and \Ovalbox have thicknesses corresponding to the picture environment's \thickline and \thinline, which are *not* lengths and thus cannot be changed with the \setlength command. The values of \thickline and \thinline depend on the size and style of the current font. Typical values are 0.8 pt for \thickline and 0.4 pt for \thinline.

For example, the commands

```
\begin{figure}
  \centering
  \shadowbox{ \begin{minipage}{2.8 in}
 \centering
 \includegraphics[totalheight=1.5in]{pend.eps}
 \caption{Shadowbox Around Entire Figure}
 \label{fig:boxed_fancy}
 \end{minipage} }
\end{figure}
```

place a shadow box around the figure's graphic and caption, as shown in Figure 24.

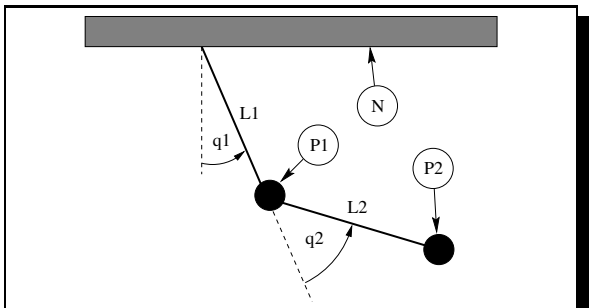

Figure 24: Shadowbox Around Entire Figure

6 Customizing Captions

6.1 Captions Next to Figures

The \caption command places the caption under the figure or table. Minipage environments can be used to trick the caption command into placing the caption next to the figure. For example, the commands

```
\begin{figure}
 \centering
 \begin{minipage}[c]{1.5in}
 \centering
 \caption{Caption on the Side}
 \label{fig:side:caption}
 \end{minipage}
 \hfill
 \begin{minipage}[c]{1.5in}
  \centering
  \includegraphics[width=\textwidth]{box.eps}
 \end{minipage}
\end{figure}
```

produces Figure 25. Likewise, the caption can be placed to the right of the figure by changing the order of the minipages.

Table 1: Fancybox commands

Command	Parameters
`\shadowbox{Example}` 	The frame thickness is `\fboxrule`. The shadow thickness is `\shadowsize` (which defaults to 4 pt).
`\doublebox{Example}` 	The inner frame thickness is .75`\fboxrule` and the outer frame thickness is 1.5`\fboxrule`. The spacing between the frames is 1.5`\fboxrule` + 0.5pt.
`\ovalbox{Example}` 	The frame thickness is `\thinlines`. Entering `\cornersize{x}` makes the diameter of the corners `x` times the minimum of the width and the height. The default is `\cornersize{0.5}`. The corner diameter can be set directly by `\cornersize*` command. For example, `\cornersize*{1cm}` makes the corner diameters 1 cm.
`\Ovalbox{Example}` 	`Ovalbox` is exactly the same as `ovalbox` except that the line thickness is controlled by `\thicklines`.

Figure 25: Caption on the Side

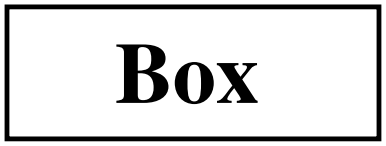

Figure 26: Graphic with a Very, Very, Very, Very, Very, Very Long Caption

Because the figure environment defined by the float package places the caption *below* the body, Figure 25 cannot be produced with the float package's figure environment. Other parts of the float package can be used as long as `\restylefloat{figure}` is not issued.

6.2 Controlling Caption Width

Since placing the `\caption` command inside a minipage environment makes the caption as wide as the minipage, this can be used to control the caption width. For example, the commands

```
\begin{figure}
  \centering
  \includegraphics[width=2in]{box.eps}
  \caption{Graphic with a Very, Very, Very,
    Very, Very, Very Long Caption}
\end{figure}
```

produce the graphic in Figure 26.

Note that the caption in Figure 26 is as wide as the page text. The width of the caption can be limited by placing it inside a minipage environment. For example, the commands

```
\begin{figure}
 \centering
 \begin{minipage}{2in}
  \centering
  \includegraphics[width=1.5in]{box.eps}
  \caption{Graphic with a Very, Very, Very,
     Very, Very, Very Long Caption}
 \end{minipage}
\end{figure}
```

produce the graphic in Figure 27. The minipage limits the width of the caption in Figure 27 to 2 inches.

A more general approach to controlling caption width is provided by the caption package, described in section 6.3.5.

Figure 27: Graphic with a Very, Very, Very, Very, Very, Very Long Caption

6.3 The caption Package

Since the format of LaTeX figure and table captions (especially for multi-line captions) may not be exactly what users desire, the caption package was written by Harald Axel Sommerfeldt to add flexibility to the caption formatting. Since the original caption package had some bad side-effects (particularly the requirement that it be loaded *after* other packages) it was totally re-written and renamed caption2. Although the caption2 package is technically still a beta version, it is quite stable and performs well.

The caption2 package can be used with many types of floats as it directly supports the float, longtable, and subfigure packages, and also works with the floatfig, rotating, supertabular, and wrapfig packages.

Reference [13] describes the commands for the original caption package, while the caption2 reference [14] currently includes only minimal documentation. The `test2.tex` test file demonstrates many of the caption2 capabilities. The options are described in Table 2.

6.3.1 Caption Styles

The caption2 package defines the following caption styles:

normal Full lines are justified (aligned with both left and right margins) with the last line being left-justified.

center All lines of the caption are centered.

flushleft All lines of the caption are left-justified, leaving the right side ragged.

flushright All lines are right-justified, leaving the left side ragged.

centerlast All the lines are justified with the last line being centered.

indent Same as "normal" style except that the second and subsequent lines are indented by the length `\captionindent`. `\captionindent` is zero by default, so a command such as

```
\setlength{\captionindent}{1cm}
```

must be used to set the indentation.

hang Same as "normal" style except that the second and subsequent lines are indented by the width of the caption label (e.g., "Figure 12:").

Usually these styles are specified as `\usepackage` options such as

```
\usepackage[centerlast]{caption2}
```

which makes all the captions in the document have `centerlast` style. Examples of the caption styles are shown in Figures 28–34.

6.3.2 Changing the Caption Style

The `\captionstyle` command changes the caption style. Placing the `\captionstyle` command inside an environment changes only those captions in that environment. For example, the commands

```
\begin{figure}
\captionstyle{centerlast}
\centering
\includegraphics[width=3in]{box.eps}
\caption{Centerlast Caption Style.
   Centerlast Caption Style.}
\end{figure}
```

give only the current figure a `centerlast` style because `\captionstyle` is inside the figure environment. The commands

```
\captionstyle{centerlast}
\begin{figure}
 \centering
 \includegraphics[width=3in]{box.eps}
 \caption{Centerlast Caption Style.
    Centerlast Caption Style.}
\end{figure}
```

give subsequent figures a `centerlast` style because `\captionstyle` is outside the figure environment.

6.3.3 One-Line Captions

If the caption is only one line, all of the above styles center the caption. To force the styles to be enforced even for one-line captions, one must include the `nooneline` option:

```
\usepackage[nooneline,flushleft]{caption2}
```

This formats *all* captions (including one-line captions) with the `flushleft` style. If you want to change the `nooneline` option inside the document, `\onelinecaptionstrue` centers one-line captions, and `\onelinecaptionsfalse` formats them as normal. For example, the commands

```
\begin{figure}
 \captionstyle{flushleft}
 \onelinecaptionstrue
 \centering
 \begin{minipage}[c]{2.5in}
```

Table 2: caption2 options

Caption Style	`normal`, `center`, `flushleft`, `flushright`, `centerlast`, `hang`, `indent`	Selects the caption style (see section 6.3.1).
Caption Fontsize	`scriptsize`, `footnotesize`, `small`, `normalsize`, `large`, `Large`	Select the fontsize for the caption label (e.g., "Figure 12:") and the caption text.
Caption Label Font Shape	`up`, `it`, `sl`, `sc`	Makes the caption label (e.g., "Figure 12:") have upright, italic, slanted, or small caps shape, respectively. Does not affect caption text.
Caption Label Font Series	`md`, `bf`	Makes the caption label (e.g., "Figure 12:") have a medium or boldface series font, respectively. Does not affect caption text.
Caption Label Font Family	`rm`, `sf`, `tt`	Makes the caption label (e.g., "Figure 12:") have roman, sans serif, or typewriter font, respectively. Does not affect caption text.
One-Line Caption Formatting	`oneline`, `nooneline`	Controls the formatting for one-line captions (see section 6.3.3)

```
   \includegraphics[width=\textwidth]{box.eps}
   \caption{First Caption}
 \end{minipage}
\end{figure}
```

center one-line captions as shown in Figure 35. The commands

```
\begin{figure}
 \captionstyle{flushleft}
 \onelinecaptionsfalse
 \centering
 \begin{minipage}[c]{2.5in}
   \includegraphics[width=\textwidth]{box.eps}
   \caption{Second Caption}
 \end{minipage}
\end{figure}
```

format one-line captions as shown in Figure 36.

6.3.4 Linebreaks in Captions

When the caption fits in one line, it is processed in an `hbox`, which ignores any \\ or `\par`. Thus one cannot generally specify linebreaks in captions. However, the caption2 package provides the `\onelinecaptionsfalse` command (or `nooneline` option) to turn off this behavior. For example, the commands

```
\begin{figure}
 \centering
 \includegraphics[width=3in]{box.eps}
 \captionstyle{center}
 \onelinecaptionsfalse
 \caption{First Line of Caption
   \protect\\ Second Line of Caption}
 \label{fig:caption:linebreak}
\end{figure}
```

produce the caption in Figure 37. Since \\ is fragile, it must be preceded by `\protect`.

6.3.5 Caption Widths

Section 6.2 demonstrated that a `\caption` command appearing in outer paragraph mode can become as wide as the page text as shown in Figure 26. Placing a `\caption` command in a minipage limits the width of the caption to the width of the minipage as shown in Figure 27. The caption2 package provides functions which directly specify the captions' width/margins.

- `\setcaptionwidth{`*width*`}` sets the width of the caption to *width*, where *width* can be in any valid TEX units.
- `\setcaptionmargin{mar}` sets the margins to *mar*, making the caption width be the standard width minus 2 times *mar*.

  If *mar* is negative, the caption is made wider than the standard width, which is useful in subfigures and minipage environments.

Figure 28: Normal Caption Style. Normal Caption Style.

Figure 29: Center Caption Style. Center Caption Style.

Figure 30: Centerlast Caption Style. Centerlast Caption Style.

Figure 31: Flushleft Caption Style. Flushleft Caption Style.

Figure 32: Flushright Caption Style. Flushright Caption Style.

Figure 33: Indent Caption Style. Indent Caption Style.

Figure 34: Hang Caption Style. Hang Caption Style.

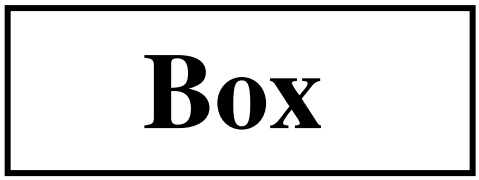

Figure 35: First Caption

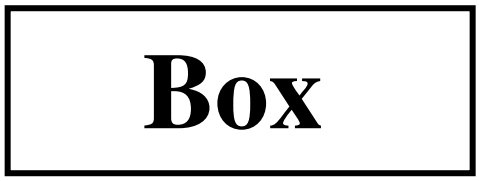

Figure 36: Second Caption

For example, the commands

```
\begin{figure}
 \setcaptionwidth{3in}
 \centering
 \includegraphics[width=2in]{box.eps}
 \caption{Figure Caption Limited to Three Inches}
\end{figure}
```

make the caption 3 inches wide, as shown in Figure 38.

While in the previous example we directly set the width of the caption, alternatively the width can be indirectly set by specifying the caption's margin. For example, the commands

```
\begin{figure}
 \captionstyle{normal}
 \setcaptionmargin{1in}
```

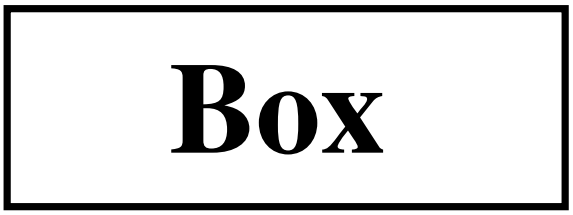

Figure 37: First Line of Caption Second Line of Caption

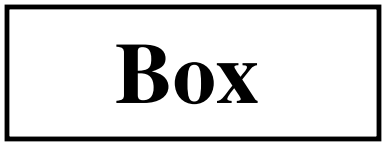

Figure 38: Figure Caption Limited to Three Inches

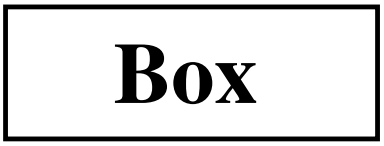

Figure 39: Figure Caption With One-Inch Margins on Each Side

```
\centering
\includegraphics[width=2in]{box.eps}
\caption{Figure Caption With
  One-Inch Margins on Each Side}
\end{figure}
```

indent both sides of the caption one inch from the page margins, as shown in Figure 39.

6.3.6 Caption Font and Delimiter

While the `scriptsize`, `Large` and other options for `\usepackage{caption2}` change the size of both the caption label (e.g., "Figure 12:") and the caption text, the `up`, `it`, `sl`, `sc`, `md`, `bf`, `rm`, `sf`, and `tt` options affect only the caption label.

Users can achieve more flexibility by redefining the `\captionfont` and `\captionlabelfont` commands. The caption is created by the following commands

```
{\captionfont%
{\captionlabelfont \captionlabel
   \captionlabeldelim}%
\captiontext}
```

where the `\captionlabel` command produces "Figure 12", the `\captionlabeldelim` command produces ":", and the `\captiontext` command produces the caption text. Thus `\captionfont` affects both the caption label and caption text, while `\captionlabelfont` affects only the caption label.

LaTeX fonts are described by encoding, size and three type style components: shape, series, and family ([1, pages 37,115], [2, pages 170-71]). These characteristics can be specified in the `\captionfont` and `\captionlabelfont` commands. For example, the commands

```
\begin{figure}
 \renewcommand{\captionfont}
   {\Large \bfseries \sffamily}
 \renewcommand{\captionlabelfont}{}
 \centering
 \includegraphics[width=2in]{box.eps}
 \caption{Test Caption}
\end{figure}
```

Figure 40: Test Caption

Figure 41: Test Caption

produce Figure 40. Here the `\captionlabelfont` command does nothing. This means that it does not overwrite any font characteristics and all the `\captionfont` settings are carried over to the caption label. Since no shape declaration was specified, the entire caption has the default upright shape. The commands

```
\begin{figure}
 \captionstyle{normal}
 \renewcommand{\captionfont}
   {\Large \bfseries \sffamily}
 \renewcommand{\captionlabelfont}{\small}
 \centering
 \includegraphics[width=2in]{box.eps}
 \caption{Test Caption}
\end{figure}
```

produce Figure 41. In this example, the `\small` in `\captionlabelfont` overwrites the `\Large` from `\captionfont`. However, since there are no series or family changes in `\captionlabelfont`, the `\bfseries` and `\sffamily` declarations carry over to the caption label.

The default colon delimiter can be changed by redefining the `\captionlabeldelim` function. For example, the commands

```
\begin{figure}
 \captionstyle{normal}
 \renewcommand{\captionlabeldelim}{.\quad}
 \centering
 \includegraphics[width=2in]{box.eps}
 \caption{Caption with New Delimiter}
\end{figure}
```

change the delimiter in Figure 42 from the default colon to a period followed by a quad space.

Figure 42. Caption with New Delimiter

Figure 43
Customized Caption Style

6.3.7 Custom Caption Styles

The caption2 package also allows users to create their own caption styles. For example, the following commands

```
\newcaptionstyle{mystyle}{%
\usecaptionmargin\captionfont%
{{\centering\bfseries
  \captionlabelfont\captionlabel\par}%
\centering\captiontext\par}}

\begin{figure}
\captionstyle{mystyle}
\centering
\includegraphics[width=2in]{box.eps}
\caption{Customized Caption Style}
\end{figure}
```

makes the caption label boldface and places it on a separate line from the caption text, as shown in Figure 43.
See the caption2 test file [15] for more user-defined caption style examples.

7 The PSfrag System

While there are many drawing and analysis packages which produce EPS files, most of them do not support symbols and equations as well as LaTeX. The PSfrag system allows LaTeX users to replace text strings in EPS files with LaTeX text or equations. Currently available for both DOS and Unix, the PSfrag system consists of the LaTeX style file `psfrag.sty` and the `perl` script `ps2frag` and is well-documented by [11].

PSfrag currently does not support compressed or non-EPS graphics. This means that if PSfrag is used for even one graphic in a document, all of the document's graphics must be non-compressed EPS files.

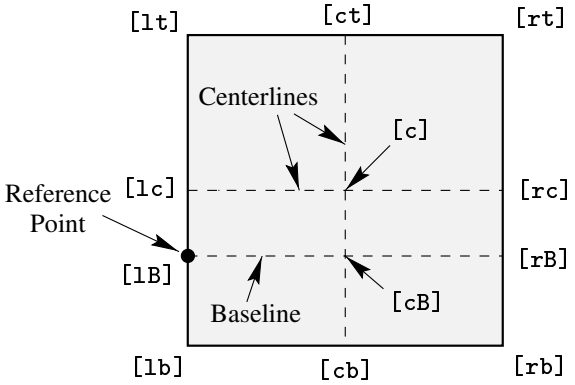

Figure 44: Available Origin Points

The procedure for using `PSfrag` is:

1. Create an EPS file.
2. At the operating system prompt, type:
   `ps2frag file.eps`
   which scans `file.eps` for text strings and then records these locations in the EPS file. Since this added information is in the form of header comments in the EPS file, it does not change the appearance of the EPS output.
3. In the LaTeX document, use the following commands:
   (a) Include `\usepackage{psfrag}` in the preamble.
   (b) Use the `\psfrag` command to specify the EPS text and the LaTeX string to replace it. This makes the specified substitution occur in any subsequent `\includegraphics` command issued in the same environment.
   (c) Use the `\includegraphics` command as usual.

The LaTeX `\psfrag` command has the following syntax

`\psfrag{PStext}[posn][PSposn][scale][rot]{text}`

with its arguments described in Table 3.

The `posn` and `PSposn` options are one of the 12 points shown in Figure 44 on page 301, except that the `c` specifier is not available (e.g., to align the left-center, use `[l]` instead of `[lc]`; to align centers, use `[]` instead of `[cc]`). See [11] for examples of various combinations of placement points.

7.1 PSfrag Example

The commands

```
\includegraphics{pend.eps}
```

include the graphic without any PSfrag replacement, producing Figure 45. The commands

Table 3: \psfrag Options

PStext	Text in EPS file to be replaced. PSfrag is sensitive about what type of text it replaces. For example, if the EPS file contains the text `Error (%)`, the percent sign confuses LaTeX and PSfrag *cannot* be used on the file, regardless of whether PSfrag replaces `Error (%)`. Instead, regenerate the EPS file using text such as Error (percent) which does not contain any of the LaTeX special characters.
posn	*(Optional, Defaults to `[Bl]`.)* Position of placement point relative to new LaTeX text. `[]` indicates center.
PSposn	*(Optional, Defaults to `[Bl]`.)* Position of placement point relative to existing EPS text. `[]` indicates center.
scale	*(Optional, defaults to 1.)* Scaling factor for the text. For best results, avoid using the scaling factor and instead use LaTeX type-size commands such as `\small` and `\large`.
rot	*(Optional, defaults to zero.)* When this rotation angle is zero, the new text is inserted at the same angle as the existing EPS text. When an angle is specified here, it is the angle of rotation of the new text relative to the existing text. The angle is in degrees with a counterclockwise rotation being positive. This option is useful when dealing with EPS files generated by applications which only allow horizontal text. This option effectively adds rotated-text capabilities to those applications.
text	The LaTeX text to insert into the EPS graphic. Like regular LaTeX text, math formulas must be enclosed by dollar signs (e.g., `$\frac{1}{2}$` or `$x^2$`) and special symbols can be used (e.g., `\%` produces %).

```
\psfrag{q1}{$\theta_1$}
\psfrag{q2}{$\theta_2$}
\psfrag{L1}{$L_1$}
\psfrag{L2}{$L_2$}
\psfrag{P1}[][]{$P_1$}
\psfrag{P2}[][]{\Large $P_2$}
\includegraphics{pend.eps}
```

include the graphic with PSfrag replacement, producing Figure 46. The first four `\psfrag` commands position the new LaTeX text such that its left baseline point corresponds to the left baseline point of the EPS text. The last two `\psfrag` commands position the new LaTeX text such that its center corresponds to the center of the EPS text.
Note that one need not replace all the EPS text with LaTeX text. For example, the `N` tag is left unchanged in Figure 46. Also note that `\psfrag` matches *entire* text strings. Thus the command

```
\psfrag{pi}{$\pi$}
```

replaces the string `pi` with π, but does not affect the strings `pi/2` or `2pi`. Separate `\psfrag` commands must entered for these strings.

7.2 LaTeX Text in EPS File

In the previous section, the `\psfrag` command specified the LaTeX text in the LaTeX file. While this is the most popular method, PSfrag's `\tex` command includes LaTeX text directly in EPS files. The `\tex` command has the following syntax

```
\tex[posn][PSposn][scale][rot]{text}
```

which is the same as `\psfrag` command, except that there is no `{PStext}` argument. Unlike the `\psfrag` command, the `\tex` command is placed in the EPS file.

For example, if an EPS file contains the text `\tex{$x^2$}` PSfrag automatically replaces it with x^2. The left-baseline point of x^2 is aligned with the left-baseline point of `\tex{$x^2$}`. Note that PSfrag does the replacement automatically; apart from the `\usepackage{psfrag}` command, it does not require any commands in the LaTeX file. Placement, scaling, and rotation options can be specified as with the `\psfrag` command. If an EPS file contains the text `\tex[][]{$x^2$}` PSfrag replaces it with a centered x^2. The `\tex` command must be *entire* text string. For example, the text

```
Transfer Function \tex{$\frac{s+a}{s+b}$}
```

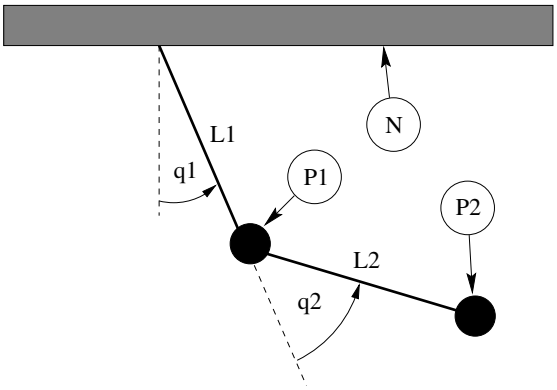

Figure 45: Without PSfrag Replacement

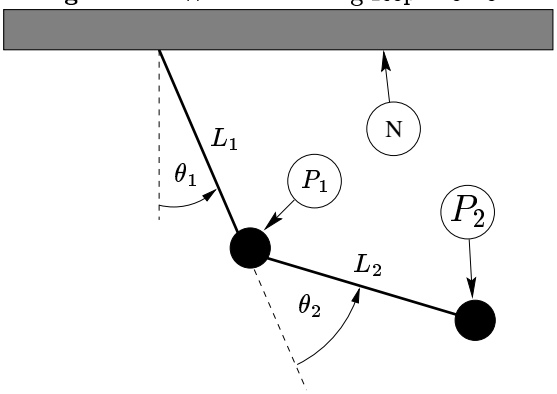

Figure 46: With PSfrag Replacement

produces an error. Instead use

```
\tex{Transfer Function $\frac{s+a}{s+b}$}
```

The advantage of the `\tex` command is that the LaTeX file doesn't need to be edited when an EPS file is modified. The `\tex` command has two disadvantages. First, the EPS file cannot be used for non-LaTeX purposes, while the EPS graphic in Figure 45 could be used without replacement. Second, if `\tex` command contains complicated formulas, the text can extend beyond the edge of the graphics, enlarging the EPS BoundingBox. This oversized BoundingBox causes incorrect graphic placement in LaTeX.

7.3 Text Scaling in PSfrag

A subtlety of the `\includegraphics` command (see [6, page 3]) comes into play with PSfrag. When scaling options are specified *before* rotation

```
\includegraphics[width=3in,angle=30]{file.eps}
```

the scaling is implicitly handled by the graphics inclusion function. However, when scaling options are specified *after* rotation

```
\includegraphics[angle=30,width=3in]{file.eps}
```

the graphic is first included at its natural size, then rotated, and then scaled. Since PSfrag replaces the new text during the graphics inclusion, the second command scales the new PSfrag text while the first command does *not*. When the included size of the EPS graphic greatly differs from its natural size, the two commands produce very different results. See [11, pages 10-11] for information.

8 Graphics in Page Header or Footer

The fancyhdr package (an improved version of the old fancyheadings package) makes it easy to customize a document's page headers and footers. It is often desired to place a logo or other EPS graphics in the header or footer, which results in the same EPS file being included multiple times.

8.1 Including An EPS File Multiple Times

There are three common methods for including the same EPS graphics many times

1. Use `\includegraphics{file}` in the places where you want the graphic. This has two problems

   (a) LaTeX must find and read the file every time the `\includegraphics` command is used.

   (b) The repeated graphics commands may result in a very large final PostScript file.

2. Save the graphics in a LaTeX box, using the box wherever you want the graphic. This saves LaTeX time since it must only find and read the file once. However, it does not reduce the size of the final PostScript file.

   At the beginning of the file, include the following commands

   ```
   \newsavebox\mygraphic
   \sbox\mygraphic{\includegraphics{file.eps}}
   ```

   Then use the command `\usebox{\mygraphic}` wherever you want the graphic.

3. Define a PostScript command which draws the graphics, and then issue the PostScript command wherever you want the graphic. Since the final PostScript file includes the graphics commands only once, the final PostScript file is much smaller.

   Since the graphics commands are stored in printer memory while the final PostScript file is being printed, this method may cause the printer to run out of memory and not print the document.

8.2 Defining a PostScript Command

To convert the EPS graphics into a PostScript Command, the EPS file must be broken into two files, one which defines the PostScript dictionary and the graphics commands, and another which includes the header information and uses the previously-defined PostScript command. For example, an EPS file created by xfig has the form

```
%!PS-Adobe-2.0 EPSF-2.0
%%Title: /tmp/xfig-fig017255
%%Creator: fig2dev Version 2.1.8 Patchlevel 0
%%CreationDate: Sun Sep  3 15:36:01 1995
%%Orientation: Portrait
%%BoundingBox: 0 0 369 255
%%Pages: 0
%%EndComments
/$F2psDict 200 dict def
$F2psDict begin
...
%%EndProlog

$F2psBegin
...
$F2psEnd
```

Where ... indicates unlisted commands. The EPS file generally contains three parts

1. The header commands which begin with %
2. The Prolog section which starts with

   `/$F2psDict 200 dict def`

   and ends with `%%EndProlog`. The Prolog defines the commands in the PostScript dictionary used by the EPS file. In this example, the dictionary is named `$F2psDict` although other names can be used.
3. The last part contains the commands used to draw the graphics.

Suppose the above EPS file is named `file.eps`. Now create the files `file.h` and `file.ps` where `file.h` contains

```
/$F2psDict 200 dict def
$F2psDict begin
...
%%EndProlog

/MyFigure {
$F2psBegin
...
$F2psEnd
} def
```

and `file.ps` contains

```
%!PS-Adobe-2.0 EPSF-2.0
%%Title: /tmp/xfig-fig017255
%%Creator: fig2dev Version 2.1.8 Patchlevel 0
%%CreationDate: Sun Sep  3 15:36:01 1995
%%Orientation: Portrait
%%BoundingBox: 0 0 369 255
%%Pages: 0
%%EndComments
$F2psDict begin MyFigure end
```

`file.h` defines the dictionary and defines the PostScript command `/MyFigure`, while `file.ps` contains the header information and uses the PostScript command defined in `file.h`. In particular, it is important that the `file.ps` header includes the `%!PS...` line and the BoundingBox line. The graphics can then be used in the LaTeX document as

```
   \documentclass{article}
   \usepackage{graphicx}
   ...
   \special{header=file.h}
   ...
   \begin{document}
...
\includegraphics[width=2in]{file.ps}
...
\includegraphics[totalheight=1in]{file.ps}
...
   \end{document}
```

Note that the original file `file.eps` is not used. Since the graphics commands in `file.h` are only included once, the final PostScript file remains small. However, this still requires LaTeX to find and read `file.ps` whenever the graphics are used. The following commands produce a small final PostScript file while reading `file.ps` only once.

```
\documentclass{article}
\usepackage{graphicx}
...
\special{header=file.h}
\newsavebox\mygraphic
\sbox\mygraphic{
   \includegraphics[width=2in]{file.ps}}
\begin{document}
...
\usebox{\mygraphic}
...
\resizebox*{1in}{!}{\usebox{\mygraphic}}
...
\end{document}
```

Like the previous example, these commands produce a 2-inch wide graphic and another graphic whose totalheight is 1 inch.

8.3 The fancyhdr Package

An easy method of including graphics in the heading is to use the fancyhdr package, which is documented by [8]. The header consists of three parts: its left field, its center field, and its right field. The

`\fancyhead` command specifies the contents of the header fields, with the `L,C,R` options specifying the field(s) which the command should modify. For example

```
\pagestyle{fancy}
\fancyhead[C]{My Paper}
```

causes the center header field to be "My Paper", while

```
\pagestyle{fancy}
\fancyhead[L,R]{\textbf{Confidential}}
```

causes the text of left and right header fields to be "**Confidential**". If no `L,C,R` option is specified, it applies to all three header fields, so `\fancyhead{}` is used to clear all the header fields. Similarly, `\fancyfoot{}` specifies the left, center, and right footer fields.

Note that the `\fancyhead` commands only apply to pages whose style is "fancy". Even though `\pagestyle{fancy}` causes the document to have a fancy page style, some pages (title pages, table of contents pages, the first page of chapters, etc.) are still given a plain pagestyle by default.

8.3.1 Graphics in Page Header/Footer

The commands in the fancyhdr package can insert graphics in the headers and footers. For example, after splitting the EPS file `file.eps` into the two file `file.h` and `file.ps` as described in section 8.2, the commands

```
\documentclass{article}
\usepackage{fancyhdr,graphicx}
%% must be large enough for graphic
\renewcommand{\headheight}{0.6in}
\renewcommand{\textheight}{7.5in}

% Define PostScript graphics command
\special{header=file.h}

% Save graphics in LaTeX box
\newsavebox\mygraphic
\sbox\mygraphic{\includegraphics
  [totalheight=0.5in]{file.ps}}

\pagestyle{fancy}
\fancyhead{}   % clear all header fields
\fancyhead[L]{\usebox{\mygraphic}}
\fancyfoot{}   % clear all footer fields
\fancyfoot[C]{\thepage}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}

\begin{document}
...
\end{document}
```

places the graphics at the top left of each "fancy" page with a 0.5 pt horizontal line drawn under the header. Additionally, the page number is placed at the bottom center of each page, with no horizontal line drawn above the footer.

8.3.2 Odd/Even Headings

When the `[twoside]` documentclass option is used, one may want to individually specify the odd and even page headers/footers. The `E,O \fancyhead` options specify the even and odd page headers, respectively. If the `E,O` options are not specified, the command applies to both even and odd pages. Likewise the `E,O \fancyfoot` options specify the even and odd page footers. For example,

```
\pagestyle{fancy}
\fancyhead[LE]{My Paper}
\fancyhead[RO]{My Name}
\fancyfoot[C]{\thepage}
```

places "My Paper" in the upper left of even fancy pages, "My Name" in the upper right of odd fancy pages, and the page number in the bottom center of all fancy pages. Replacing the

```
\fancyhead[L]{\usebox{\mygraphic}}
```

command in the above example with

```
\fancyhead[LE,RO]{\usebox{\mygraphic}}
```

places the graphic at the top outside (the left side of even pages, right side of odd pages) of all fancy pages.

8.3.3 Modifying Plain Pages

Although the above commands do not affect pages with plain pagestyles, the `\fancypagestyle` command can be used to modify the plain pagestyle. For example

```
\documentclass{article}
\usepackage{fancyhdr,graphicx}

%% must be large enough for graphic
\renewcommand{\headheight}{0.6in}
\renewcommand{\textheight}{7.5in}

% Define PostScript graphics command
\special{header=file.h}

% Save graphics in LaTeX box
\newsavebox\mygraphic
\sbox\mygraphic{\includegraphics
    [totalheight=0.5in]{file.ps}}

\pagestyle{fancy}
\fancyhead{}   % clear all header fields
\fancyhead[L]{\usebox{\mygraphic}}
\fancyfoot{}   % clear all footer fields
\fancyfoot[C]{\thepage}
```

```
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}

\fancypagestyle{plain}{%
  \fancyhead{}   % clear all header fields
  \fancyhead[L]{\usebox{\mygraphic}}
  \fancyfoot{}   % clear all footer fields
  \fancyfoot[C]{\thepage}
  \renewcommand{\headrulewidth}{0.5pt}
  \renewcommand{\footrulewidth}{0pt}}

\begin{document}
...
\end{document}
```

place the graphic at the upper left of every page (both plain and fancy). Likewise, when the `twoside` documentclass option is used, replacing both of the

```
\fancyhead[L]{\usebox{\mygraphic}}
```

commands with

```
\fancyhead[LE,RO]{\usebox{\mygraphic}}
```

places the graphic at the top outside of every page (both plain and fancy).

Acknowledgements

I would like to thank the contributors to the newsgroup `comp.text.tex`, whose posts and replies provided me with the information for this document. In particular, David Carlisle provided a great deal of assistance. I would also like to acknowledge Jim Hafner for providing the procedure in section 8.2. Finally, I would like to thank the readers of previous versions who provided me with feedback.

References

[1] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1.

[2] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8.

[3] Helmut Kopka and Patrick Daly. *A Guide to LaTeX 2ε.* Addison-Wesley, Reading, Massachusetts, 1995, ISBN 0-201-4277-X.

[4] D. P. Carlisle. *Packages in the 'graphics' bundle.* Available from CTAN as `grfguide.tex` or `grfguide.ps`.

[5] D. P. Carlisle and S. P. Q. Rahtz. *The graphics package.* Available from CTAN as `graphics.dtx`.

[6] D. P. Carlisle and S. P. Q. Rahtz. *The graphicx package.* Available from CTAN as `graphicx.dtx`.

[7] Trevor Darrell. *Psfig/TeX 1.10 Users Guide.* Available from CTAN as `psfig-doc.tex`.

[8] Piet van Oostrum, *Page layout in LaTeX*, Available from CTAN as `fancyhdr.tex`

[9] Sebastian Rahtz and Leonor Barroca, *The `rotating` package*, Available from CTAN as `rotating.dtx`

[10] Steven Douglas Cochran. *The `subfigure` package.* Available from CTAN as `subfigure.dtx`.

[11] Craig Barratt and Michael C. Grant. *The PSfrag system.* Available from CTAN as `pfgguide.tex`.

[12] Timothy Van Zandt. *Documentation for fancybox.sty* Available from CTAN as `fancybox.doc`

[13] Harald Axel Sommerfeldt. *The caption package* Available from CTAN as `caption.dtx`

[14] Harald Axel Sommerfeldt. *The caption package.* Available from CTAN as `caption2.dtx`

[15] Harald Axel Sommerfeldt. *Test of the caption package.* Available from CTAN as `test2.tex`

◇ Keith Reckdahl
Stanford University
Box 9030
Palo Alto, CA 94309
USA
`reckdahl@leland.stanford.edu`