Creation and Incorporation of PostScript Graphics with TEX-formatted Labels into TEX Documents

Neil A. Weiss
Department of Mathematics
Arizona State University
Tempe, AZ 85287
Phone: 602-965-3951; FAX: 602-965-8119

Abstract

In this paper, we will discuss the incorporation of PostScript-created graphics with TEX-formatted labels into TEX documents using the *Mathematical Graphics System*, or *MG* for short. We will provide a brief overview of the creation of such graphics and explain in detail the TEX macros employed to accomplish the inclusion of the graphics and the TEX labels into the output of a TEX document. Additionally, we will provide illustration of how to customize figure insertions for particular applications.

What is the Mathematical Graphics System?

The *Mathematical Graphics System*, *MG*, is a menu-based program designed for the creation and display of both two- and three-dimensional PostScript graphics on DOS based computers. Although it is not the purpose of this paper to discuss the graphics capabilities of *MG*, we should point out that they are considerable.

Two-dimensional graphs can be specified as Cartesian, polar, or parametric curves, straight lines, and vectors. Three-dimensional graphs can involve Cartesian or parametric surfaces, straight lines, and parametric curves. In addition, surfaces can be ruled by arbitrary families of curves and can be shaded.

MG also has *free-drawing* capabilities where the user can specify points of interpolation for filled or unfilled polygons and for filled or unfilled splines. The scatter-plot feature of *MG* permits the plotting of points read from a data file or for drawing curves or polygons (filled or unfilled) through the points. It also allows for the shading of regions in two-dimensions. For a detailed explanation of the graphics capabilities of *MG*, we refer the reader to Israel and Adams (*Mathematical Graphics System User's Manual*).

The output of *MG* can be obtained in two forms: (1) as a pair of files to be incorporated into the source code of a TEX document or (2) as an encapsulated PostScript file that can either be sent directly to a PostScript output device (printer or phototypesetter) or included in a larger document that will be printed on such a device. In this paper, we will concentrate on the first form of output.

The two files referred to in (1) consist of an `lbl` (label) file and a `ps` (PostScript) file. The `lbl` file contains, among other things, information regarding the TEX-formatted labels and their placement. The `ps` file is a PostScript file delineating the graphic. This latter file is incorporated into the TEX output by employing the `\special` command, the exact syntax of which depends on the `dvi` to PostScript driver being used.

To obtain the `lbl` and `ps` files for a graphic constructed by *MG*, the user first selects "Make PostScript file" in one of the several menus where that option occurs. The user is then asked to choose a file name that will serve as the name for both the file with the `.lbl` extension and the file with the `.ps` extension. Once that is accomplished, the user should specify "importing into TEX" for the type of PostScript file to be saved. (The other option, "printing directly," yields an encapsulated PostScript file with the labels being set by PostScript.) We will discuss both the `lbl` and `ps` files created by *MG* in greater detail shortly.

We should note that the user can also elect to save the `grf` file corresponding to a graphic constructed by *MG*. That file records the menu selections and other information required by *MG* to reproduce the graphic at a later time for editing or electronic viewing purposes.

The LBL and PS Files

Let us now discuss, in detail, the `lbl` and `ps` files obtained when an *MG* graphic is saved using the "importing into TEX" option. The `lbl` file is an ASCII file that contains the positions, justification codes, and text for each of the labels. These labels, which were specified in *MG*, are assumed to be in TEX form; e.g., labels containing mathematical expressions should be enclosed in dollar signs. The `lbl` file also includes information about the dimensions of the PostScript figure as well as a few other pieces of data.

To illustrate, we will examine the `lbl` file of the graphic displayed in Figure 1, the graph of the polar equation $r = \sin 8\theta$:

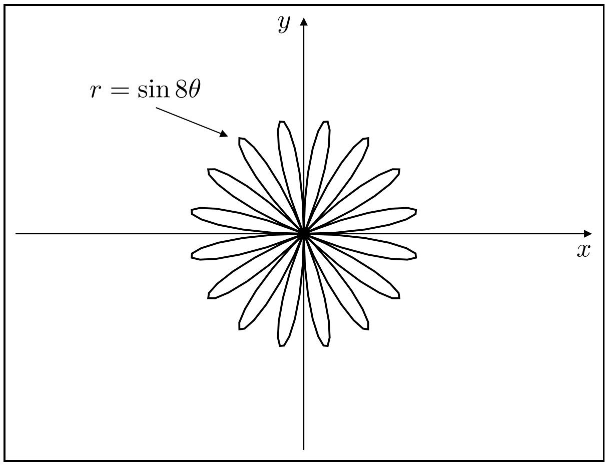

Figure 1: Graph of a polar curve

Note that there are three labels in Figure 1, all of which are set by TEX: the labels for the x and y axes and the label describing the equation of the curve, $r = \sin 8\theta$.

A listing of the `lbl` file that supplies the requisite information to TEX for the proper location and coding of the labels is depicted in Table 1 at the top of the next column. The annotations in the right column of the table are included for explanatory purposes and are not part of the `lbl` file.

In understanding the position coordinates for the labels, it is important to note that the dimensions are in points with reference to an origin at the upper left corner of the figure and that the x-dimension increases to the right whereas the y-dimension increases downward. Since, as we mentioned earlier, the `lbl` file is an ASCII file, it can be edited using a text editor. Thus, to move the equation label, $r = \sin 8\theta$, up five points, we need only change its y-coordinate from 27 to 22.

We now see that the `lbl` file provides the information necessary for TEX to set the labels in Figure 1. The remaining portion of Figure 1 — the axes, graph, and leader — are described in the `ps` file. TEX plays no role in processing the information in the `ps` file except to pass the instructions in the `\special` command to the `dvi` file.

Table 1: Label file with annotation for Figure 1

`101040`	Version of *MG*
`217`	Width of figure, in points
`163`	Height of figure, in points
`104`	x-coord of the y-axis label
`0`	y-coord of the y-axis label
`2`	Right just'n of y-axis label
`2`	Top just'n of y-axis label
`$y$`	y-axis label
`217`	x-coord of x-axis label
`86`	y-coord of x-axis label
`2`	Right just'n of x-axis label
`2`	Top just'n of x-axis label
`$x$`	x-axis label
`49`	x-coord of eqn label
`27`	y-coord of eqn label
`1`	Horiz centering of eqn label
`1`	Vert centering of eqn label
`$r=\sin 8\theta $`	Equation label
`-1000`	End of label file

How MG Incorporates the Labels and Graphics into TEX

In order to incorporate the TEX-formatted labels, specified in the `lbl` file, and the PostScript graphic, specified in the `ps` file, the distribution of *MG* includes a collection of TEX macros in the file `fig.tex`. The macro that provides the commands for the importation of the labels and the calling of the PostScript graphic is called `\figinsert`. For reference purposes, we have presented a listing of that macro in the appendix to this paper.

The argument to `\figinsert` is the common name of the `grf`, `lbl`, and `ps` files. In the case of Figure 1, it is `polar`. The `\figinsert` macro first prepares the input stream, `labelfile`, to read from the specified label file (in Figure 1, `polar.lbl`). Then it reads the first three lines of the label file, which contain the version of *MG* and the width and height of the PostScript figure, to `\mgversion`, `\pswidth`, and `\psheight`. At this time, the name of the file is also displayed on the terminal by use of the `\message` command.

If boxfigures is true, then the figure will be surrounded by a rectangle, as is done in Figure 1. The line width and border width can be changed from the default in *MG* (1 pt and 2 pt, respectively) by altering the values of the dimension registers, \boxrulewidth and \boxborderwidth. In Figure 1, \boxrulewidth=.4pt and \boxborderwidth=4pt.

Next, \figinsert constructs a \vbox whose height is \psheight, the height of the PostScript figure, and uses the \special command to call the PostScript graphic (in Figure 1, the graphic is in the file polar.ps). The actual description of the graphic will be imported into the PostScript file for the main document by the driver. The default syntax for the \special command can be changed to accommodate any dvi to PostScript driver.

The control sequence, \setlabelsize, can be used to alter the default type size for the labels. For example, to change to nine point, we use the definition \def\setlabelsize{\ninepoint}. We should mention that fig.tex includes the \ninepoint and \eightpoint macros provided in Appendix E of *The TEXbook* (pages 414 and 415).

The next portion of the \figinsert macros continues reading the label file using a loop. It begins by reading the fourth line of the lbl file to \xcoord. That line will either be the x-coordinate of the first label or the code number, −1000, for the end of the file. In the latter case, the reading terminates and the final positioning, boxing (if any), and closing of the input stream occurs. Otherwise, the y-coordinate and justification codes are read to \ycoord, \justx, and \justy, and the label itself is read to \label. Then the label is placed in the box \labox which is copied to the appropriate location using the coordinates and justification codes. Iteration now occurs for further labels or the end of the file.

High-level Macros for Figure Format

MG supplies three high-level macros for figure placement and captions. Of course, these macros all call the low-level \figinsert macro.

The first high-level macro for figure placement and captioning is \cfig, whose listing is as follows:

```
\def\cfig#1#2{\par\smallskip
\openin\labelfile=#1.lbl
\ifeof\labelfile\immediate
\write16{Can't find #1.LBL; I quit!}
\end\fi \closein\labelfile
\vbox{
 \centerline{\figinsert{#1}}\smallskip
 \centerline{\figfont#2}}\smallskip}
```

As we can see, the \cfig macro takes two arguments, the figure file name (e.g., polar) and the figure caption. The figure is centered on the \hsize with the caption centered below. Figure 1 uses the \cfig macro.

Note the call to the \figinsert macro. Also note that the caption (#2) is set in the font specified by \figfont. The default for that font in *MG* is boldface but that can be changed as desired.

The second high-level macro supplied by *MG* for figure placement and captions, \rfig, provides for text on the left side of a page and the figure with its caption on the right. This macro takes three arguments: the text, the figure file name, and the figure caption. The width of the text is determined by the width of the figure; namely, it is the \hsize minus the width of the figure, \pswidth, minus 1 pc. As with \cfig, the caption is centered below the figure.

The third high-level placement macro, called \twofigs, allows for two side-by-side figures with captions. Each caption is centered below its figure. The entire display is also centered on the \hsize.

Of course, the user can modify the high-level placement macros supplied by *MG* or define his/her own high-level placement macros. For instance, one might want to increase the skips occurring above and below the figure when applying the \cfig macro, say, by replacing two of the \smallskips by \medskips.

Or suppose, for example, that a book being prepared in TEX uses count registers to keep track of the chapter number and figure number and that the caption of a figure consists of the word "FIGURE" followed by the chapter number, a period, the figure number, and a colon, all set in the font specified by \figurefont, and then by the description of the figure, set in the font specified by \figuretitlefont. Further suppose that figures are insertions using the \midinsert macro of plain.tex. Then the following adaptation of the \cfig macro can be used to accomplish the setting of the figures:

```
\newcount\chapterno \newcount\figureno
\newif\iffigtex

\def\fig#1\par{\iffigtex\relax
 \else \globaldefs=1 \input fig
 \figtextrue
 \def\setlabelsize{\ninepoint}
 \boxrulewidth=.4pt \boxborderwidth=9pt
 \boxfigurestrue \globaldefs=0 \fi
 \goodbreak \midinsert
\global\advance\figureno by1
\cfig
{fig\number\chapterno-\number\figureno}
{\figurefont \uppercase{Figure}
\number\chapterno.\number\figureno:
\figuretitlefont #1}\endinsert}
```

Note that, at the beginning of the `\fig` macro, TeX checks to see whether the `fig.tex` macros have been input; if they haven't, they are input along with some changes to some of the *MG* parameters.

The following figure, the file name of which is `fig4-1`, was obtained by applying the `\fig` macro. We simply typed

```
\fig A $\Gamma$ section.
```

when `\chapterno=4` and `\figureno=0`.

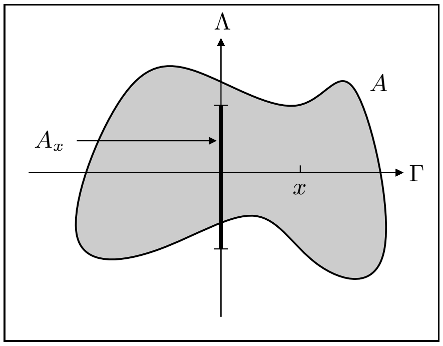

FIGURE 4.1: A Γ section.

Bibliography

Israel, R. B. and R. A. Adams. *MG, Mathematical Graphics System User's Manual.* MG Software, 4223 W. 9th Avenue, Vancouver, B.C., Canada V6R 2C6, 1990.

Knuth, Donald E. *The TeXbook*. Reading, Mass.: Addison-Wesley, 1984.

Appendix

The following is a listing of the \figinsert macro. Also included are the requisite definitions for that macro. As can be seen, the \figinsert macro utilizes both the lbl (label) and ps (graphic) files corresponding to the graph constructed by *MG*.

```
\newcount\vpos
\newread\labelfile
\newif\ifdoit
\newbox\labox

\def\figinsert#1{\par  %#1=filename
\openin\labelfile=#1.LBL
\global\read\labelfile to\mgversion\message{#1}
\global\read\labelfile to\pswidth
\global\read\labelfile to\psheight
\ifboxfigures\boxit{\fi\vbox to\psheight pt{\vfill
\special{ps: plotfile #1.PS}% Version for DVILASER/PS!
\vskip-\psheight pt \setlabelsize
\hbox to\pswidth pt{\hss}%
\parindent=0pt\offinterlineskip
\vpos=0
\loop\global\read\labelfile to\xcoord
\ifnum \xcoord < -999 \doitfalse\else\doittrue\fi
\ifdoit \global\read\labelfile to\ycoord
\global\read\labelfile to\justx
\global\read\labelfile to\justy
\global\read\labelfile to\label
\global\setbox\labox=\hbox{\label\hskip-0.3em}%
\advance\vpos by-\ycoord
\vskip-\vpos pt \vpos=\ycoord
\hbox to\pswidth pt{\hskip\xcoord pt
\hbox to 0pt{\ifnum\justx>0\hss\fi
\vbox to0pt{
\ifnum\justy<2\vss\fi
\copy\labox\kern0pt
\ifnum\justy>0\vss\fi}\ifnum\justx<2\hss\fi}\hss}
\repeat
\advance\vpos by-\psheight
\vskip-\vpos pt}\ifboxfigures}\fi\closein\labelfile}
```