

Finely-tuned Plots in \LaTeX for Statistics Education utilizing an R-based \TeX pic Plug-In

Shunji Ouchi

Economics
Shimonoseki City University

Setsuo Takato

Pharmaceutical Sciences
Toho University

Abstract

We have been developing \TeX pic as a library of macro package of Computer Algebra Systems(CAS) to generate standard \LaTeX source code for high-quality scientific artwork. We have recently implemented \TeX pic in R, which is a popular open-source software tool used in statistical analysis and for graphic output. It is often the case that the default or standard output from R is not exactly what the users requires, particularly when producing graphics for educational purposes. Through taking full advantage of the functionality of R and \LaTeX , \TeX pic enables us to produce teaching/learning materials incorporating figures which are designed to help the learner better understand statistical ideas and theories. In this paper we look at the use of the plug-in to generate two basic statistical plots, the histogram and boxplot, which are most useful in descriptive statistics. We will also describe \TeX pic functionality that can be used to produce enhanced graphic output.

1 Introduction

According to a recent survey conducted by the authors in Japan[1], about 74 percent from a sample of 378 who teach mathematics to first and second-year university or technical college students utilise \LaTeX to create teaching/learning materials. Nowadays \LaTeX is particularly valuable for university and college mathematics teachers as a tool for preparing printed materials in Japan.

When teachers want to include graphics into a \LaTeX document, most of them will do so using data formatted as EPS files. These files are then inserted into the \LaTeX text file using the 'includegraphics' command. However, this method has some disadvantages. There are a number of reasons for this: the size of the EPS file is not small; the canvas dimension of the graphic is not easily handled; and the graphics embedded in the \LaTeX file is difficult to fine-tune. What \LaTeX actually provides is very limited set of graphical capabilities to yield drawings.

On the other hand our \TeX pic plug-in, finely-tuned control of various graphical features such as line style, shading, and text display is enabled until the user's needs are fully satisfied. R-based \TeX pic commands allow us to convert graphic outputs of R into \TeX pic specials subsequently stored in \TeX pic files. Such files can be embedded into a standard \LaTeX file. So we can

easily produce \LaTeX documents incorporating figures which are designed to help the learner better understand statistical ideas and theories using \KTeXpic .

In this paper we look at the use of R-based \KTeXpic plug-in to enhance standard statistical graphic output of R in \LaTeX and some interesting \KTeXpic capabilities are discussed by means of illustrative examples.

2 R and Its Graphic Output

R is a popular open-source software environment licenced under the GNU General Public Licence used in statistical computing and production of graphics. It provides a wide variety of statistical and graphical techniques, and is highly extensible. R compiles and runs on a wide variety of platforms, such as Windows, the Macintosh operating system and UNIX. The software can be obtained from the Comprehensive R Archive Network (CRAN) accessible via the main R web site <http://www.r-project.org>.

In R functions the graphics systems and graphics packages can be divided into three main types: *high-level* functions that produce complete plots; *low-level* functions that add further output to an existing plot; and functions for working interactively with graphical output (see [7]). The main *high-level* plotting functions are the ones used to produce complete plots such as scatterplots, histograms, and boxplots.

Here we demonstrate simple usage of *high-level* and *low-level* functions in R session. Producing a histogram using *high-level* function `hist` is done by:

```
c1 <- c(2,2.5,3,4,5,6,8,10,12,15,20,25,30)
hist(data,breaks=c1) # 'data' is data set
```

with the output as seen in **Figure 1**.

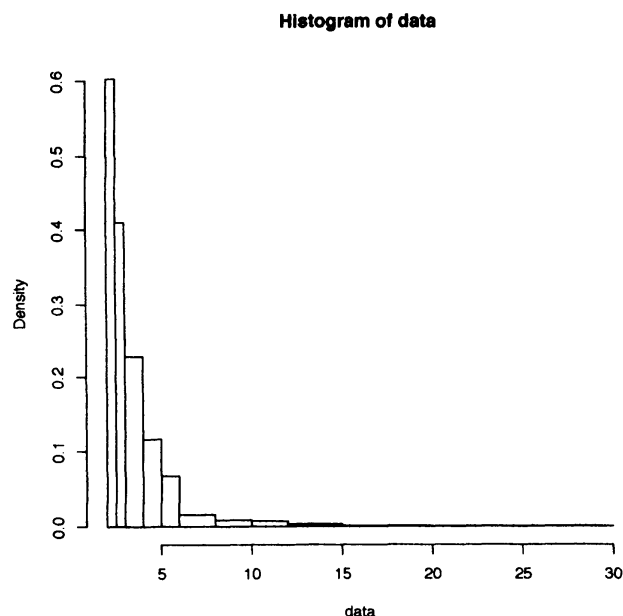


Figure 1 Graphic output of histogram using R

Next we add a density estimate to the existing graphic using *low-level* function `lines`.

```
lines(density(data))
# The density function finds a density estimate from the data
```

giving us the graph below (Figure 2).

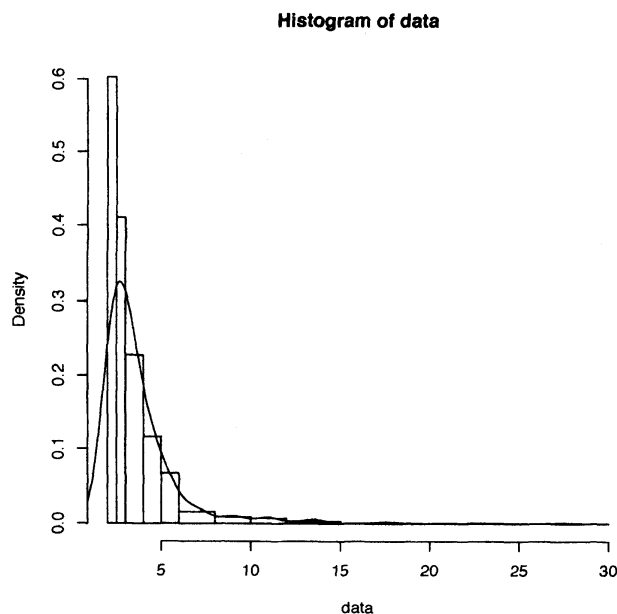


Figure 2 Graphic output of histogram with density estimate

3 Flow of K_ETpic drawing

Figure 3 illustrates the K_ETpic Graphical Pipeline for R.

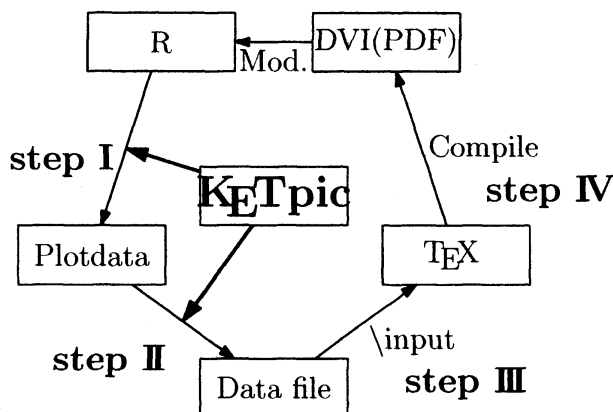


Figure 3 K_ETpic graphical pipeline for R

We will demonstrate the K_ETpic session workflow by outlining of process of enhancing the graph (Figure 1 shown in section 3) according to the pipeline. Our final aim is illustrated in

Figure 4. `KETpic` enable us to achieve this aim. While R itself is equipped with some basic commands or functions for modifying the standard graphic output, it doesn't include powerful features users demand. Often to obtain high-quality graphics, users must work with the R-created graphics in a third-party graphical editor such as Adobe Illustrator. `KETpic` provides an economical alternative to this, and also has lower requirements producing smaller sized files than the EPS format files researchers often need to work with.

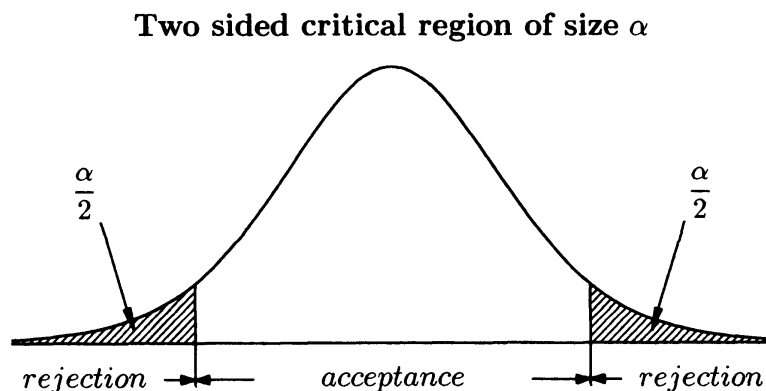


Figure 4 Graphic output of `KETpic`

The user begins by opening R for a new session. We start by loading the plug in:

```
load("ketpic.Rdata")
```

This is an important step as it ensures all new `KETpic` commands are automatically available from the very beginning.

step I After setting up the canvas dimensions for \LaTeX drawing, the user runs R commands, routines and libraries to perform computations and generate graphic output.

```
Setwindow(c(-3,3),c(-0.1,0.41))
Setscaling(6.46)
G1 <- Plotdata("dt(x,10)","x","N=100")
G2 <- Listplot(c(XMIN,0),c(XMAX,0))
```

step II `KETpic` commands allow us to convert our graphic data into *Tpic* special code subsequently stored in *Tpic* files.

```
Openfile("fig.tex") # open tex file at folder
Beginpicture("1.5cm")
# to create \begin{picture} ... \end{picture} in LATEX
Drwline(G1,G2,1.2)
Endpicture(0)
Closefile()
```

The output of this R session is a collection of plain \TeX files containing data for graphical objects.

step III Such files can then be invoked from a source \TeX file which should, when run, be compiled to generate a DVI file (`fig.tex` in the sample below).

```
\documentclass[10pt]{article}
\usepackage{ketpic}
\begin{document}

\input{fig.tex}

\end{document}
```

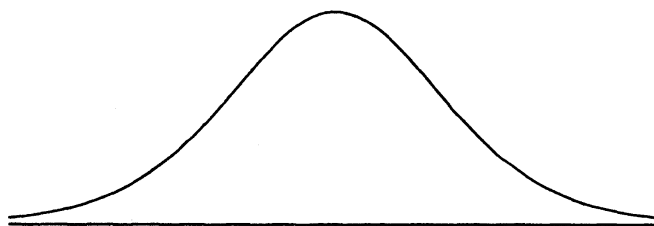


Figure 5 DVI file of graphic output

step IV The DVI file can be further converted into other formats or printed as a paper hardcopy. This cycle can be repeated any number of times allowing the user to fine-tune the graphic output to his/her demands.

For example, if we want to shade part of the right tail area of the distribution, we simply need to add following commands in **step I** and **step II** respectively:

in **step I**

```
X1 <- qt(0.95,10) # The R function qt is quantile of t distribution
P1 <- c(X1,0)
P2 <- c(X1,dt(X1,10))
G3 <- Listplot(P1,P2)
G5 <- Hatchdata(list("iii"),list(G1,"s"),list(G3,"e"),
               list(G2,"n"),45,0.4) # shade area
X2 <- (2*X1+XMAX)/3
P3 <- c(X2,dt(X2,10)/2)
P4 <- P3+0.3*c(1,3)
```

in **step II**

```
Drwline(G3)
Drwline(G5,0.5)
```

giving us the graph below (**Figure 6**).

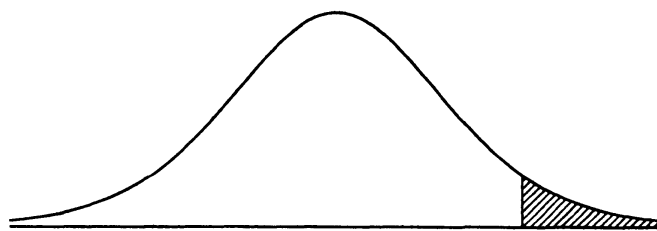


Figure 6 Modified graphic output

Using several $\text{K}\epsilon\text{Tpic}$ commands in the manner described above, we can produce our final graph (Figure 4).

4 High-Quality Statistical Plots using $\text{K}\epsilon\text{Tpic}$

Currently the R-based $\text{K}\epsilon\text{Tpic}$ plug-in includes a powerful draw function; `Drwhistplot` and `Drwboxplot`. This draw function has been developed to meet various user demands and create high-quality detailed graphs. The function is composed of three main parts:

1. It generates plot data from the data set by the R function;
2. It produces ‘graphical framework data’ (data for adding a title and setting axis styles) and converts this into *Tpic* special code;
3. It outputs the command sequence to be executed in **step II** (see section 4) and returns the required information to create graphic output.

The new Japanese mathematics curricula, which was implemented on April 1, 2009 for the lower-secondary schools and will begin on April 1, 2012 for the upper-secondary schools, aims to identify and explain trends by using histograms for first year junior high school students. Boxplot will be covered in the first year of senior high school under the new curricula. In the following subsections we look at two basic statistical plots, histogram and boxplot, and describe $\text{K}\epsilon\text{Tpic}$ functionality to create enhanced graphic output for them.

4.1 Program for Histograms

The program for creating histogram output is as follows:

```
Drwhistplot(Data, "H.m", c(15,10),"c",
            title=list("Histogram","l","bf"),
            xlab=list("Annual income","n","",5) ,
            ylab=list("No. of persons"),
            plot=TRUE, densplot=FALSE,breaks=c(2,2.5,⋯,30))
# 'Data' is data set
# c(15,10) sets actual veiwing canvas dimensions (in cm)
# optional argument 'breaks=' controls the bin size
```

The character string `H.m` is a variable name. Information on a title and axis styles, histogram plotting data, and a command sequence (`Cmd` shown below) is substituted for it when the `Drwhistplot` function was executed.

```
Cmd <- H.m$commands
fix(Cmd) # open R data editor if necessary
Maketexfile(Cmd,"fig.tex")
```

The content of `Cmd` is as follows:

```
$commands
  [,1]
[1,] ""
[2,] ""
[3,] "Beginpicture('0.4cm')"
```

```
[4,] ""
[5,] ""
[6,] "Drwhistframe(H.m)"
[7,] "HtickLV(H.m$info$mids,1,1)" #set tick mark on horizontal axis
[8,] "VtickLV(max(H.m$info$counts),0,0)"
[9,] "Drwline(H.m[['plotdata']]$histplot)"
[10,] "Dashline(H.m[['plotdata']]$fpplot)"
[11,]
[12,] ""
[13,] ""
[14,] ""
[15,] "Endpicture(1)"
```

This is a matrix which is comprised of a number of character strings (7 in this case) which are `KETpic` commands. `Maketexfile(Cmd,"fig.tex")` executes `KETpic` commands stored in `Cmd` and converts our graphical data into `Tpic` special code. `Maketexfile` significantly simplifies the process in **step II**. After executing this command and compiling the `LATEX` file shown in **step III**, we obtain a DVI file.

Users can easily finely-tune the existing graphical output according to his/her demands as described in section 3. This can be done by either of the following two ways. The first is to type the necessary commands in R data editor (**Figure 7**), the other is by adding the commands to the existing program using R's edit commands. For example, if we want to shade the second bar from the left, we simply need to use the following process dependent on the user interfaces:

graphical interface case:

After typing `fix(Cmd)` in an open R console window, an R data editor window appears on the screen. Then we enter `Shade(list(Hd[[2]]),0.2)` into a blank line in R data editor.

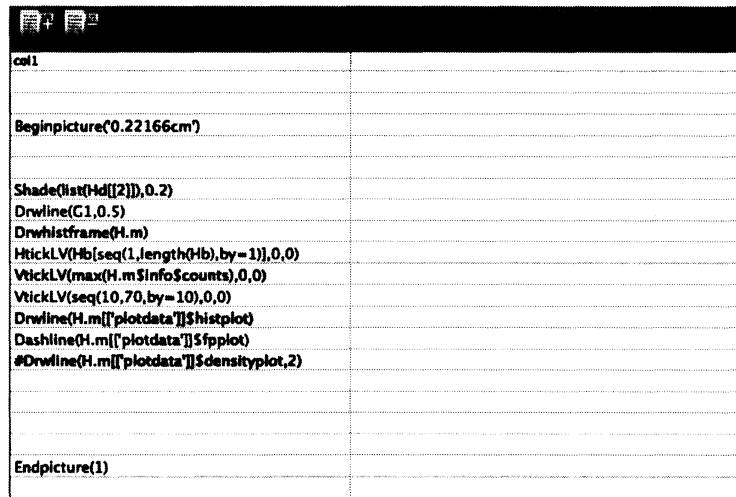


Figure 7 Window of R data editor

command-line interface case:

Add `Insertcom("Cmd",6,"Shade(list(Hd[[2]]),0.2)")` to the existing program. After adding other graphics and annotations to the plot, we obtain the graph illustrated below. It is not necessary to delete other graphics and annotations in this example, however it is possible to do so.

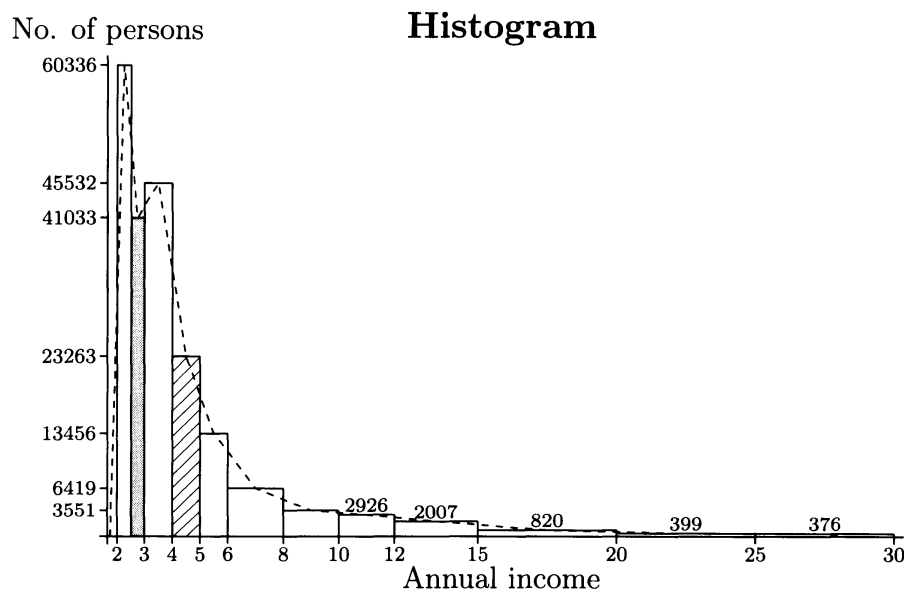


Figure 8 Finely-tuned graphic output for histogram

4.2 Program for Boxplots

The program for creating boxplot output is as follows:

```

capnames <- c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width")
Title <- list("Boxplot", "1", "bf")
Cap <- list(capnames, "n", 0, 0.5)

```



```

Drwboxplot(iris[1:4], "BoxD", c(10,10), title=Title, cap=Cap,
           ylab=list(""), ptsize=5, plot=TRUE)
# iris[1:4] is data set

Cmd <- BoxD$commands
fix(Cmd) # if necessary
Maketexfile(Cmd,"fig.tex")

```

The function `Drwboxplot` works on the same principle as the function `Drwhistplot`. We can shade any box individually and indicate the figures in the y axis showing the locations of the boxes which stand for the median, and the 25th and 75th percentiles. After running the program and adding other required graphics and annotations to the plot, we obtain the graph below.

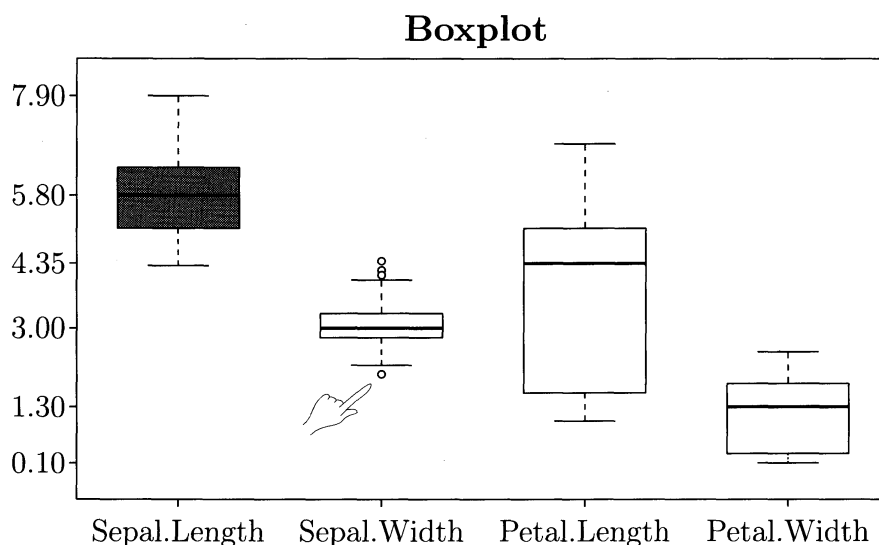


Figure 9 Finely-tuned graphic output for boxplot

Boxplots are diagrams for presenting necessary information to see the center, spread, skew, and length of tails in a data set. This type of graph allows us to compare many distributions in one figure.

4.3 $\text{K}_{\text{E}}\text{T}_{\text{pic}}$ Metacommands

`Cmd` is a matrix which is comprised of a number of character strings, as we saw in the previous subsection. Each character string stands for a $\text{K}_{\text{E}}\text{T}_{\text{pic}}$ command as listed in the example in section 4.1. When the draw function `Drwhistplot` or `Drwboxplot` is executed, `Cmd` is substituted for the relevant variable `H.m` or `BoxD`, and the draw function interprets character strings stored in `Cmd` as $\text{K}_{\text{E}}\text{T}_{\text{pic}}$ commands and executes them. `Maketexfile` also interprets character strings stored in `Cmd` as $\text{K}_{\text{E}}\text{T}_{\text{pic}}$ commands and executes them. R is equipped with the function `eval(parse(text="a character string"))` which parses a character string and then evaluates it in the environment from which `eval(parse(...))` was called. Using this function allows the draw functions `Drwhistplot` and `Drwboxplot`, and `Maketexfile` to work as a single command executing $\text{K}_{\text{E}}\text{T}_{\text{pic}}$ commands themselves—in other words, functioning as a 'metacommand'.

5 Conclusion and Further Development

We have developed an R-based KETpic plug-in to yield high-quality statistical graph output to be embedded into standard $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. Currently the draw function is able to produce histograms and boxplots. In the future we intend to expand the scope of the function to enable the output of a greater range of statistical graphs designed to help the learner better understand statistical ideas. We will enhance the power of the R-based KETpic plug-in, bringing increased functionality, and creating a user-friendly system.

References

- [1] Kiyoshi Kitahara, Takayuki Abe, Masataka Kaneko, Satoshi Yamashita and Setsuo Takato, "Towards a More Effective Use of 3D-Graphics in Mathematics Education – Utilization of KETpic to Insert Figures into LaTeX Documents–", to appear in The International Journal for Technology in Mathematics Education, Vol. 17, Number 3, 2010.
- [2] Koshikawa, H., Kaneko, M., Yamashita, S., Kitahara, K., Takato, S.,: Handier Use of Scilab to Draw Fine $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ Figures – Usage of KETpic Version for Scilab–. Proc. ICCSA 2010, IEEE Press, 39-48
- [3] M. Kaneko, T. Abe, M. Sekiguchi, Y. Tadokoro, K. Fukazawa, S. Yamashita and S. Takato, "CAS-aided Visualization in LaTeX documents for Mathematical Education," Teaching Mathematics and Computer Science, vol. 8, issue 1, (2010)
- [4] A. Galvez, A. Iglesias and S. Takato, "New Matlab-Based KETpic Plug-In for High-Quality Drawing of Curves," 2009 International Conference on Computational Sciences and its Applications, IEEE Press, 2009, pp.123-131.
- [5] M. Sekiguchi, M. Kaneko, Y. Tadokoro, S. Yamashita and S. Takato, "A New Application of CAS to LaTeX -Plottings," Lecture Notes in Computer Science, Springer-Verlag, 4488, pp. 178-185, 2007.
- [6] M. Sekiguchi, S. Yamashita and S. Takato, "Development of a Maple Macro Package Suitable for Drawing Fine TeX -Pictures," Lecture Notes in Computer Science, Springer-Verlag, 4151, pp. 24-34, 2006.
- [7] P. Murrell, "R Graphics," Chapman & Hall/CRC, 2006.