Appendix B

Creating LaTeX Arrays, Tables, and Figures

B.1 Introduction

When it comes to making tables and charts, the main words you need to know are *array, tabular, table*, and *figure*. The first two describe environments for creating an organized structure of information while the latter two refer to organizational units within a LaTeX document. This handout will go over some code for making arrays, tables, and figures in LaTeX. It also contains information about importing PostScript graphics.

B.2 Arrays

Arrays are environments that *must be in math mode* to work. Arrays line items up in columns. Here are some basic steps for making arrays:

(a) Set the array in math mode with \[.

(b) Type `\begin{array}`.

(c) Use an argument to describe how you want your table to be justified. Immediately following the `\begin{array}` command, add a set of brackets. Inside the brackets, use the letters r (right), c (center), and l (left) for each column to describe how it will be formatted. For example, if you have a three-column array and you want the text to be right-justified in the first column, centered in the second, and left-justified in the third, the argument would be {rcl}.

(d) Type your data, using & to separate columns and \\ to move to the next row.

(e) End the array with `\end{array}` and \].

Optional Features:

(a) **Changing the inter-row spacing**
Use the `arraystretch` command:

```
\renewcommand{\arraystretch}{number of your choice}
```

"Number of your choice" is the factor by which the spacing will be increased; for example, typing 1.5 inside the brackets will increase the spacing between rows by a factor of 1.5.

(b) **Centering** Include `\begin{center}` and `\end{center}` to center the table.

(c) **Adding vertical lines** If you wish to have lines between columns or around the sides, add | between the "{rcl}", or whatever your justifying argument happens to be. For example: {r|c|l} creates lines between columns, and {|r|c|l|} creates lines between columns and around the outside of the array.

(d) **Adding horizontal lines** To create horizontal lines between rows, use `\hline` (h stands for horizontal) at the end of each line following the \\. To create horizontal lines around the top and bottom, add `\hline` after the justifying argument (see step 3), and following the last \\.

(e) **Normal text** If you want any regular text in your array, you need to use `\mbox{text}`.

Example array # 1

$$\begin{array}{rcl}
f(t) & F(s) & \mbox{Remark} \\
\delta(t) & 1 & \mbox{impulse function} \\
u(t) & \frac{1}{s} & \mbox{unit step function} \\
e^{at}u(t) & \frac{1}{s-a} & \mbox{one-sided exponential}
\end{array}$$

The array is generated with the code:

```
\begin{center}                      %  <-- optional b
\[                                  %  <-- step a
\begin{array}{rcl}                  %  <-- steps b and c
f(t) & F(s) & \mbox{Remark}\\       %  <-- step d, optional e
\delta(t) & 1 & \mbox{impulse function}\\
u(t) & \frac{1}{s} & \mbox{unit step function}\\
e^{at}u(t) & \frac{1}{s-a} & \mbox{one-sided exponential}
\end{array}                         %  <-- step e
\]                                  %  <-- step e
\end{center}
```

Example array #2

$f(t)$	$F(s)$	Remark
$\delta(t)$	1	impulse function
$u(t)$	$\frac{1}{s}$	unit step function
$e^{at}u(t)$	$\frac{1}{s-a}$	one-sided exponential

This table was created with the following code:

```
\renewcommand{\arraystretch}{1.5}   %  <-- optional a
\begin{center}                      %  <-- optional b
\[                                  %  <-- step a
\begin{array}{|r|c|l|} \hline       %  <-- steps b, c; optional c, d
f(t) & F(s) & \mbox{Remark}\\  \hline%  <-- step d, optional e
\delta(t) & 1 & \mbox{impulse function}\\
u(t) & \frac{1}{s} & \mbox{unit step function}\\
e^{at}u(t) & \frac{1}{s-a} & \mbox{one-sided exponential} \\
\hline
\end{array}                         %  <-- step e
\]                                  %  <-- step e
\end{center}
```

B.3 Tabular

The `array` environment is great for math, but if you only want text, you can use the `tabular` environment in the same way as `array` except without the math mode.

1. Begin with `\begin{tabular}`.
2. Follow steps 3-4 listed above for arrays. Optional features 2-4 are also applicable to `tabular`.
3. Finish with `\end{tabular}`.

You can also adjust the width of the table by adding a *: `\begin{tabular*}{width}`.

Tabular Example

Name	E-mail
Michael Gustafson	mrg@duke.edu
Michael Ehrenfried	mje7@duke.edu

Here is the code. Note that it is very similar to that of `array`, except there is no math mode:

```
\begin{center}
\begin{tabular}{|r|l|}\hline
Name & E-mail\\ \hline\hline  % <-- note that two \hlines produce a double line
Michael Gustafson & mrg@duke.edu\\ \hline
Michael Ehrenfried & mje7@duke.edu\\ \hline
\end{tabular}
\end{center}
```

B.4 Tables and Labeling Environments

A table is a floating environment that surrounds an array or tabular environment. The `table` command allows you to add a caption, add a label, create more than one table on a page, and move it where you want it to go. The commands for creating a table environment are

```
\begin{table} \end{table}
```

Note: `\begin{table}` must be typed *before* `\begin{tabular}` or `\begin{array}`, and `\end{table}` must be typed *after* `\end{tabular}` or `\end{array}`.

- **Captions:** To add a caption, after the `\end{tabular}` or `\end{array}` commands, but before `\end{table}`, type

  ```
  \caption{Your caption here}
  ```

- **Labels:** Labeling allows you to reference a table, figure, or section later in the document. Use the command `\label{Your label here}`. For example, if you want to label a table as "table:one," inside the `caption` command type:

  ```
  \caption{\label{table:one}Your caption here}
  ```

  Now, anytime you want to refer to this particular table, use the `ref` command:

  ```
  \ref{table:one}
  ```

  Now the number of the table will be inserted. You can also refer to the page number where table:one is located:

  ```
  \pageref{table:one}
  ```

Labels may also be included in section headings and figures so you can refer to these later. See Kopka & Daly for more details.

- **Placing two narrow tables next to each other:** You can use the `minipage` environment to place two narrow tables next to each other. You will also need to use the `picture` environment within the minipage and specify the width of the figure. The command `\hfill` creates white space between the two tables. See Kopka & Daly for more examples and details.
- **Positioning the table where you want it:** The `table` environment allows your table to "float" around. Often LaTeX will place the table wherever it fits. However, you can tell LaTeX to put it in a specific location with [h] (here), [t] (top), and [b] (bottom). If you want the table to appear exactly [h]ere, after `\begin{table}` include [h]:

  ```
  \begin{table}[h]
  ```

  - Note that [h], [t], and [b] could be used just with tabular or array environments, following `\begin{tabular or array}`, without a table environment.
  - You can also center the table by including a `\begin{center}` command after the `\begin{table}` but before `\begin{tabular or array}`.

Example Table

Name	E-mail
Michael Gustafson	mrg@duke.edu
Michael Ehrenfried	mje7@duke.edu

Table B.1: Caption Text

This will now be referred to as Table B.1 on page App B – 4. I can also say that Table B.1 is in Section B.4 because I put a label command in the section title.

All this code is:

```
\begin{table}[h]                                      % <-- note the [h]
\begin{center}
\begin{tabular}{|r|l|}\hline
Name & E-mail\\ \hline\hline
Michael Gustafson & mrg@duke.edu\\ \hline
Michael Ehrenfried & mje7@duke.edu\\ \hline
\end{tabular}
\caption{\label{table:one}Caption Text}               % <-- caption & label
\end{center}
\end{table}
This will now be referred to as Table \ref{table:one} on % <-- ref to table
page \pageref{table:one}.                             % <-- pageref
I can also say that Table \ref{table:one} is in Section
\ref{section:tl} because I put a label command in the    % <-- ref to section
section title.
```

Note that the section reference required me to put the following in the section title:

```
\section{\label{section:tl}Tables and Labeling Environments}
```

Labels may also be associated with `chapter` or `subparagraph`.

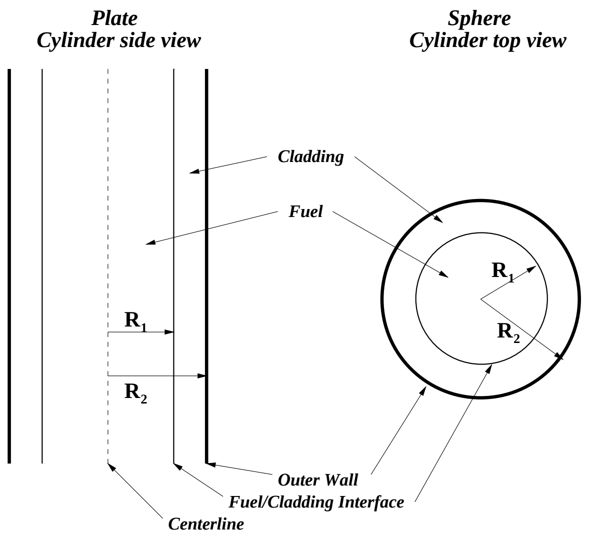

Figure B.1: Drawing from ME 150L test

B.5 Figures

The last topic covered here is how to get a PostScript file into a document. The basic command sequence for inserting figures is as follows:

```
\begin{figure}[PLACEMENT]
\epsfig{file=FIGURENAME.eps, ANY EXTRA COMMANDS}
~\\
\caption{YOUR CAPTION}
\end{figure}
```

The PLACEMENT command works exactly as it did for tables- you can type h, t, or b. Refer back to the section titled "Positioning the table where you want it" for details. Replace YOUR CAPTION and FIGURENAME with the appropriate names. Note that the figure must be saved as an **.eps** file. ANY EXTRA COMMANDS may include width= or height=, and these are explained below.

The code to get Figure B.1 of Section B.5 on page App B – 5 (as well as this line) is:

```
The code to get Figure \ref{figure:drawing} of Section
\ref{section:fig} on page \pageref{figure:drawing} (as well as this
line) is:
\begin{figure}[t]
\begin{center}
\epsfig{file=drawing.eps, angle=-90, width=3in}
~\\
\caption{\label{figure:drawing}Drawing from ME 150L test}
\end{center}
\end{figure}
```

You must include the `epsfig` style file as a package of your LaTeX document with the `usepackage` command in the header. To get more information on how to use the file, you can look at the style file itself. The lines below are copied directly from the style file itself.

```
% usage: \epsfig{file=, figure=, height=, width=,
%                      bbllx=, bblly=, bburx=, bbury=,
%                      rheight=, rwidth=, clip=, angle=, silent=}%
%
%       "file" is the filename.  If no path name is specified and the
%               file is not found in the current directory,
%               it will be looked for in directory \psfigurepath.
%       "figure" is a synonym for "file".
%       By default, the width and height of the figure are taken from
%               the BoundingBox of the figure.
%       If "width" is specified, the figure is scaled so that it has
%               the specified width.  Its height changes proportionately.
%       If "height" is specified, the figure is scaled so that it has
%               the specified height.  Its width changes proportionately.
%       If both "width" and "height" are specified, the figure is scaled
%               anamorphically.
%       "bbllx", "bblly", "bburx", and "bbury" control the PostScript
%               BoundingBox.
%       "rheight" and "rwidth" are the reserved height and width
%               of the figure, i.e., how big TeX actually thinks
%               the figure is.  They default to "width" and "height".
%       The "clip" option ensures that no portion of the figure will
%               appear outside its BoundingBox.  "clip=" is a switch and
%               takes no value, but the '=' must be present.
%       The "angle" option specifies the angle of rotation (degrees, ccw).
%       The "silent" option makes \epsfig work silently.
%
```

Note that the only required value is the file name. The rest of the values, like “width” and “height” can be included after the file name in place of ANY EXTRA COMMANDS. You can type the preferred width in inches (in) or centimeters (cm). Important: Do **not** put the file name in quotation marks!

B.6 Conclusion

LaTeX is one of the most powerful text and graphics processors around. With the information included in this document, you should be able to insert arrays, tables, figures and table and figure environments into your documents. You should also by able to dynamically reference them using the `label`, `ref`, and `pageref` commands.