Learning LaTeX

Week 2: Tables & Graphics

Philip D. Waggoner*

*philip.waggoner@gmail.com. This document was prepared by Philip Waggoner for the *Weekly Workshops on Learning LaTeX*, hosted by the Deparment of Political Science, University of Houston.

Contents

1 Introductory Remarks

Drawing tables in LaTeX like most other things, can be simple or quite difficult. The idea behind drawing tables is in line with doing anything else in LaTeX where we need to start with environments, and then proceed to complicate as much as we like.

It can be frustrating to draw tables, becuase each cell must be in alignment, though there are no boundaries keeping each cell in place, like there is in Excel, for example. Thus, drawing tables requires meticulous care and attention. But the result is well-worth the trouble.

There are many different table environments in LaTeX (e.g., table, tabular, tabu, tabularx, and so on). For today's purposes, and for most research, table and tabular should be sufficient. The difference between these two environments in that *tabular* is the default table maker in LaTeX, and *table* allows for a "floating" environment, where we can move tables around more efficiently and easily. We will be using each of these environments in today's workshop.

Our goals for today:

- Draw and manipulate tables, positions, & content
- Reference tables within text
- Insert notes within tables
- Insert graphics, plots, & figures

2 Tables

Let's get started with a simple table. To do so, we need to know a couple of useful commands and syntax. First, columns are generated by a position indicator such as "c" (for "center", or "l", "r" and so on), cell delineators are generated by "|", horizontal lines are inserted into the table with the command "hline", and we tell LaTeX where to put our values with the use of the ampersand, "&".

Thus, for a simple table, let's start with the following code,

```
\begin{document}
        \begin{tabular}{ |c|c|c| }
            \hline
            1 & cell A & cell Y \\
            2 & cell B & cell Z \\
            \hline
        \end{tabular}
\end{document}
```

And this code gives us,

1	cell A	cell Y
2	cell B	cell Z

So from the output, we can see that we told LaTeX to generate a table with three columns, with dividing lines between each column. And there should be two rows, given the two rows of information we provided. Let's change some of these parameters to see how the simple table changes. Specifically, let's add another line on top of the table, addional lines on the ends, but remove the column dividers within the table, and let's also add another row of information.

Try this new code,

```
\begin{document}
    \begin{center} % we can use the center environment to center our table
        \begin{tabular}{ ||c c c|| }
            \hline
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{document}
```

Now our table looks like,

1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

2.1 Combining Columns

Now that we have the basic intuition behind creating a table, let's complicate things just a little to give us more flexibility for customizing tables. We will start by combining columns in only part of the table. To do so, we need a new command, "multicolumn". This means, then, that we will also need to add a new line to our original table. Let's start with the following code,

```
\begin{document}
    \begin{center} % we can use the center environment to center our table
        \begin{tabular}{ ||c c c|| }
            \hline
                \multicolumn{3}{|c|}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{document}
```

Then, we get the updated table,

A Combined Column		
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

We can see our new table has a new row added to the top of the table, which is comprised of three columns. This is useful for delineating between different models presented in a single table, for example. We can also combine several columns within the table, rather than the full length of the table. So for our simple table,

```
\begin{document}
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{document}
```

And our output looks like,

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

Notice the combined two columns over the right two columns in the table. This is possible through the addition of the "{2}{c||}", which tells multicolumn to combine only 2 columns, even though there are three, and then to place the double lines to the right of it, to be consistent with our table format. Also notice the blank space in front of the & in the multicolumn line. This tells the tabular environment to not place anything in that space, but to still include it as a column in the table.

2.2 Position Tables Using the "Table" Environment

To this point, we have only been using the tabular environment to draw a table in LaTeX. But let's say we want more flexibility over precisely where our table is placed in the body of the text. The default position using only the tabular environment, is exactly where you insert the code. Thus, the table environment lets us specify the table's location.

To get started, we simply nest the tabular table within the table environment, and then specify the location using brackets after "table", and then one of the following positional identifiers: h ("here"), t ("top"), b ("bottom"), p (specific "page"), ! ("override internal position from LaTeX), H ("exactly here", which also equals, "h!"). To update our simple table code, type,

```
\begin{document}
 \begin{table}[h!]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{table}
\end{document}
```

And here's the output,

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

But, to see the precise impact of these positional identifiers, let's see all positions at once. Try several of them back to back,

```
\begin{document}

 \begin{table}[t]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{table}
%
\begin{table}[h!]
        \begin{center}
            \begin{tabular}{ ||c c c|| }
                \hline
                & \multicolumn{2}{c||}{A Combined Column} \\
                \hline
                1 & cell A & cell X \\
                2 & cell B & cell Y \\
                3 & cell C & cell Z \\
                \hline
            \end{tabular}
        \end{center}
\end{table}
%
\begin{table}[b]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
            & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
    \end{center}
\end{table}

\end{document}
```

And here's the output,

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

1

2.3 Captions, Labels, & In-Text Referencing

One of the best features of LaTeX is the ability to keep track of all tables, figures, and equations, regardless of how the text may change. For example, let's say you have a document with four tables in it. But you update the paper and decide you need to add a table of descriptive statistics. So instead of having to go back and manually change all table numbers, and also change any place in the text you reference the old table numbers, LaTeX allows you to imply reference the table with an in-text command based on the caption you gave the table. Thus, adding this fifth table is no problem at all. Nothing in your text changes, and LaTeX does the work of updating all references to table as well as table numbers throughout your text.

To get a better sense of this, we need to use the "caption" and "label" commands. The caption command gives the table a name, which is attached to the table wherever it goes in the text. The label command provides the reference we can cite within the text to keep us on track with our table numbering. To actually cite within the text, we type "ref" after the backslash to call the reference command.

So let's update some code for two tables in order to see precisely how these commands work. Type,

```
\begin{document}

\noindent First, I would like to talk about my first simple table seen below in Table \ref{table:simple.tab1} with
the caption on top. But then, we should discuss oursecond simple table below in Table \ref{table:simple.tab2}, where
the caption is placed below.


 \begin{table}[h!]
    \begin{center}
     \caption{This is the Title of Our First Simple Table}
        \begin{tabular}{ ||c c c|| }
            \hline
              & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}

     \label{table:simple.tab1} % note the difference in our labels
    \end{center}
\end{table}
%
%
\begin{table}[h!]
    \begin{center}
        \begin{tabular}{ ||c c c|| }
            \hline
            & \multicolumn{2}{c||}{A Combined Column} \\
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
        \end{tabular}
        \caption{This is the Title of Our Second Simple Table}
        \label{table:simple.tab2}
    \end{center}
\end{table}


\end{document}
```

Then, we can see the ouput as,

First, I would like to talk about my first simple table seen below in Table 1 with the caption on top. But then, we should discuss oursecond simple table below in Table 2, where the caption is placed below.

Table 1: This is the Title of Our First Simple Table

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z

Table 2: This is the Title of Our Second Simple Table

2.4 Table Notes

Finally, you almost always see some kind of note at the bottom of a table (e.g., denoting levels of statistical significance, references to standard errors, and so on). There are many ways to do this (e.g., threeparttable - you should look this up and use it on your own), but the simplest way for our purposes, is to add subsequent lines to the bottom of our table and manually format to resemble a proper table footnote using our multicolumn command from earlier, where the syntax following the multicolumn command is, "{columns}{position}{text}."

To do this, let's update our code to make our simple table look a little nicer with our new footnote,

```
\begin{document}

 \begin{table}[h!]
    \begin{center}
        \caption{This is the Title of Our First Simple Table}
        \label{table:simple.tab1}
        \begin{tabular}{ c c c }
            \hline
            Obs. & \multicolumn{2}{c}{A Combined Column} \\
            \hline
            \hline
            1 & cell A & cell X \\
            2 & cell B & cell Y \\
            3 & cell C & cell Z \\
            \hline
            \textit{Note:}  & \multicolumn{1}{r}{Here is our table note.}
        \end{tabular}
    \end{center}
\end{table}

\end{document}
```

Now, we can see our updated table is starting to look a bit more like something we would see in a journal (though we still have a ways to go),[1]

Table 1: This is the Title of Our First Simple Table

Obs.	A Combined Column	
1	cell A	cell X
2	cell B	cell Y
3	cell C	cell Z
Note:	Here is our table note.	

[1] In R, we can generate base tables of results very easily in many ways. The two main ways to get you started are using either "stargazer" or "xtable" packages. In stargazer, you can simultaneously print and save results, while also generating LaTeX code to simply copy and paste into LaTeX. In xtable, you can print the LaTeX code as well.

3 Graphics, Plots, & Pictures

To conclude, we will take a look at inserting and manipulating figures/plots/graphics/images (hereafter, *images*) into a LaTeX document. First, *we must save all images in the same location as our main TeX document*. So for example, if we have a paper saved in the folder called, "Class 1 Final Paper," all images we want to include in that paper for Class 1 must also be saved in the the "Class 1 Final Paper" folder. Once we have saved all images in the same location, we need to now need to load a package called "graphicx." This let's us place images in a file, as well as allows us to tell LaTeX where to track down our images. Once the package is included in our preamble (at the top of our TeX document), below it we need to set the "graphicspath." This is simply the file path to the location of your main document, set within braces, which specifies the location of both your main document and your images. The synax would look like,

```
\usepackage{graphicx}

\graphicspath{/Users/bpwaggo/Dropbox/LaTeX Workshop Series/Week 2}
```

Once we have our preamble set, and our graphics path specified, we can now insert images easily into any location in our paper using the "figure" environment.[2] The figure environment, like the "table" environment, allows us to specify precisely where we want to place our images using the same positional identifiers (e.g., *h*, *t*, *b*, etc.).

Let's first go to the internet and find an image of anything you want to include in your document. Once you have this, place it in the same location as your main document, whereever that is. Then, to call and place this image, we simply type,

```
\begin{document}

\begin{figure}[!h]
    \includegraphics{UHSEAL}
\end{figure}

\end{document}
```

[2] You can simply use the command "includegraphics." But this limits our flexibility for manipulating the image as we see fit. Thus, let's focus on using the figure environment instead.

Then, our output is the image,

3.1 A Few Notes on Scaling & Sizing

Now there are a few things to consider here. First, I always make the names of my files as simple and clear as possible to limit the threat of confusing LaTeX, such as removing file types from the title (.png, etc.). Also, for our purposes, let's stick with only .png files. Its cleaner and easier for the figure environment, though we can include other image types. As we are using the figure environment, we can manipulate the alignment of the opbject as with a table, using commands such as "center" (see above on centering our tables). Also, and importantly, notice how our image is a little fuzzy. This can easily be corrected by either changing the dimensions of height and width of the image manually using brackets just after the "includegraphics" command (e.g., "width=12cm, height=11cm"), or for a more careful and simpler fix, we can simply alter the scale of the image. To get a sense of this, let's update our code with a few options of rescaling,

```
\begin{document}

\begin{figure}[!h]
    \includegraphics[scale=.4]{UHSEAL}
\end{figure}
%
%
\begin{figure}[!h]
    \includegraphics[scale=.25]{UHSEAL}
\end{figure}
%
%
\begin{figure}[!h]
    \includegraphics[scale=.15]{UHSEAL}
\end{figure}

\end{document}
```

Now, let's see each of the three iterations of the rescaled University of Houston seal,