LaTeX for Absolute Beginners: A Short Course

Part Two

Prepared for the Star Lab by C. Basu*

Last updated: 05 October 2014

Contents

*This document was prepared by the author for the incoming graduate student cohort at the University of Rochester, and adapted from previous versions of the course taught by Jonathan Olmsted, Dave Armstrong and Arthur Spirling.

1 Review Exercise II

Open a new `.tex` file, and load the packages `amsmath`, `amssymb`, and `amsfonts`. Write the LaTeX code necessary to produce the following expressions.

1. $\left|4x^3 + \left(x + \frac{42}{1+x^4}\right)\right|$
2. $\frac{\partial u}{\partial t} = h^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right)$
3. $x^n = \frac{-b \pm \sqrt[n]{b^2-4ac}}{2a}$
4. $(A \cup B) \cup C = A \cup (B \cup C)$
5. $\{x \in \mathbb{R} : x^2 = 1\}$

More on TeXMaker

Although it may be helpful to learn the commands you use most frequently by heart, it may be heartening to know that a large fraction of these commands are just a click away in TeXMaker (and in many other similar text editors).

Take a moment to click on the various buttons on your screen and explore the kind of functionality TeXMaker (or your text editor) offers. If not already selected, click on the 'Structure' tab in the bottom left corner of your screen. This will bring up yet another menu of mathematical symbols for you to explore. Clicking on one of these symbols will print the associated LaTeX command inside your `.tex` file, although you will still have to ensure that the command is contained within the appropriate mode.

2 More on Typesetting Math

We have discussed how to format a wide range of mathematical expressions using LaTeX. Next, we will discuss how to organize these mathematical expressions into systems of equations, matrices and cases.

2.1 Equations, Matrices & Cases

To produce a single displayed mathematical formula, we can just use the display math mode, which we have already encountered. However, if we wish to present a *numbered* equation or formula, then the `equation` environment is necessary. For example,

```
\begin{equation}
(x-2)(x+4)=x^2+4x-8
\end{equation}
```

gives you

$$(x-2)(x+4) = x^2 + 2x - 8 \tag{1}$$

Make sure not to leave a space (e.g. a blank line) between your equation and either the `begin{equation}` or `\end{equation}` command, or LaTeX will give you an error, thinking that you have omitted an equation. (Try it out, just for kicks.) If you don't want your equation(s) to be numbered, use the commands `\begin{equation*}` and `\end{equation*}` instead. (Though, in that case, you might as well just use the display math mode.) Note that here, math mode does not need to be explicitly defined, but is implied within the `equation` environment.

To present equation arrays or systems of equations, we can use one of two methods. First, we can use the `eqnarray` environment. As an example,

```
\begin{eqnarray}
x &=& v + 6b -f \\
  &=& (p+q)(p-q)\\
  &=& p^2 - q^2
\end{eqnarray}
```

produces

$$\begin{aligned} x &= v + 6b - f && (2)\\ &= (p+q)(p-q) && (3)\\ &= p^2 - q^2 && (4) \end{aligned}$$

You will notice that:

1. Each of the equations is numbered, but the numbers follow on from the last numbered equation in the document. This means that each new equation environment does not restart the counter at (1).

2. Math mode is, again, implied.

3. To obtain an array of equations that are not numbered, use the `eqnarray*` environment instead.

4. To move to a new line within your array, we use the `\\` command. We tend not to use this command on the last line of the array, or there will be an unusually large space before the text picks up again.

5. The special character `&` is used repeatedly in the code above. These are variously called 'span marks', 'alignment stops' or 'alignment characters' depending on who you ask. Their function is pretty self-evident from the above example. Note that you must use the character `&` on *either* side of the operator around which you want the equations to align.

6. If we want to only number some of the equations in our equation array, we add the `\notag` command before the line break command `\\` for the equations we want ignored. For example, the code

```
\begin{eqnarray}
x &=& v + 6b -f          \\
  &=& (p+q)(p-q) \notag \\
  &=& p^2 - q^2
\end{eqnarray}
```

produces

$$\begin{aligned} x &= v + 6b - f && (5)\\ &= (p+q)(p-q) \\ &= p^2 - q^2 && (6) \end{aligned}$$

We can also use the `align` environment. The resulting output has slightly more compact spacing relative to output from `eqnarray`, but the end results are pretty similar. The following code:

```
\begin{align}
x &= v + 6b -f          \\
  &= (p+q)(p-q) \notag \\
  &= p^2 - q^2
\end{align}
```

produces

$$\begin{aligned} x &= v + 6b - f && (7)\\ &= (p+q)(p-q) \\ &= p^2 - q^2 && (8) \end{aligned}$$

From the user's perspective, the only noteworthy difference is that you only need to use **one** alignment character in each line to anchor your equations.

To create arrays or matrices, we can use the `array` environment. Constructing an array is slightly more complicated than the commands we have been using so far, but is fairly similar to constructing tables, and therefore good practice! The `begin{array}` command requires a mandatory argument specifying the number of columns in the array and how these columns must be justified - centre, left or right. So, if to produce an array of zeros with 3 center-justified columns and 2 rows, we write:

```
\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 0 & 0
\end{array}
```

```
\]
```

which produces

$$\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}$$

To instead produce an array with left- or right-justified columns, we replace the `c` in the mandatory argument with `l` or `r`, respectively. Some things to note:

- Unlike the `equation` or `align` environments, the `array` environment does not assume math mode, and we must define the mode explicitly.
- The alignment character `&` is now used to demarcate columns within the array.
- As with the `eqnarray` environment, we must use the `\\` command to move to a new line within the array.

In order to produce a matrix of zeros, and not simply an array, we must add delimiters. We can do so in the following manner.

```
\[
\left[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 0 & 0
\end{array}
\right]
\]
```

produces

$$\left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \end{array} \right]$$

It may be simpler to use the `bmatrix` environment instead, which is included in the `amsmath` package. By this approach, delimiters are implied, and you don't need to code them in yourself. For example,

```
\[
\begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0
```

```
\end{bmatrix}
\]
```

yields

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Moreover, unlike `\begin{array}`, the `\begin{bmatrix}` command takes no mandatory arguments, and columns are center-justified by default. However, you will still need to define math mode explicitly. You can use the same approach to create vectors. The following code:

```
\[
\begin{bmatrix}
y_1\\
y_2\\
y_3\\
\vdots\\
y_n
\end{bmatrix}
\]
```

yields

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}$$

where the `\vdots` command produces a series of three vertical dots. You can use the `\vdots` command in paragraph mode as well. (It is a relative of the `\ldots` command, which we have already encountered.) We can instead produce horizontal dots using `\hdots` and diagonal dots using `ddots`. This may come in handy when constructing, for instance, a variance-covariance matrix. For example, the following matrix

$$\begin{bmatrix} \sigma_{11} & \sigma_{12} & \dots & \sigma_{1n} \\ \sigma_{21} & \sigma_{22} & \dots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \dots & \sigma_{nn} \end{bmatrix}$$

was produced using the following code:

```
\[
\begin{bmatrix}
\sigma_{11} & \sigma_{12} & \hdots & \sigma_{1n} \\
\sigma_{21} & \sigma_{22} & \hdots & \sigma_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\sigma_{n1} & \sigma_{n2} & \hdots & \sigma_{nn} \\
\end{bmatrix}
\]
```

An even faster way of constructing an array or a matrix is by using TeXMaker's 'Wizard' function. Click on the Wizard tab on the toolbar near the top of your screen, and select the option 'Quick Array'. You can create a matrix with innumerable rows and columns with fewer than five clicks, and you have six environments to choose from!

Finally, the `cases` environment is useful for typesetting a definition by cases. Like the `array` and `bmatrix` environments, you must first enter math mode. So, the code

```
\[
f(y) =
\begin{cases}
2y, & 0 \leq y \leq 1 \\
0,  & \textrm{otherwise}
\end{cases}
\]
```

yields

$$f(y) = \begin{cases} 2y, & 0 \leq y \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

Here, the alignment character `&` separates each 'case' into two columns, and again, we must use `\\` command to move to a new line. Also note that we must use the `\textrm{}` to make our text stand upright while in math mode. Without `\textrm{}`, our output would look like this:

$$f(y) = \begin{cases} 2y, & 0 \leq y \leq 1 \\ 0, & \mathit{otherwise} \end{cases}$$

2.2 Theorems, Proofs & Definitions

When writing any formal theory paper (or problem set), you will likely need to typeset proofs, propositions and theorems. LaTeX offers a way of doing so formally. This requires the `\newtheorem` command, which allows you to define new environments. The `\newtheorem` command takes two mandatory arguments. The first argument tells LaTeX what the new environment is called, and the second argument tells LaTeX what to label the new environment in the compiled document. For example, the command `\newtheorem{definition}{Definition}` tells LaTeX that there is now a new environment, `definition`, which will be labeled as `Definition` whenever it occurs. Each new `definition` environment will also be numbered in order. This command can appear anywhere in your input file, but it is probably best to include it in your preamble. Now, the code

```
\begin{definition}
A \textbf{homogenous equation system} is of the form $\mathbf{Ax=0}$.
\end{definition}
```

yields

Definition 1. *A **homogenous equation system** is of the form* $\mathbf{Ax} = \mathbf{0}$.

Note that this does not create environments for which math mode is implied. The procedure to define lemmas, propositions, etc. is analogous. Moreover, each of these 'environments' will be numbered separately.

Finally, accompanying most theorems and propositions is a proof (or several). LaTeX has an in-built `proof` environment. We can use it as follows:

```
\begin{theorem}
$\sqrt{2}$ is irrational.
\end{theorem}
\begin{proof}
For a proof by contradiction, suppose that $\sqrt{2}$ is rational.
Then, we can express $\sqrt{2}$ as the fraction $\frac{a}{b}$ in
its most simplified form, where both $a$ and $b$ are integers. Then,
\begin{align*}
\frac{a^2}{b^2}&=2 \\
```

```
\implies a^2&=2b^2
\end{align*}
so $a$ must be even. Then, we can write $a$ as $2m$, for some integer
$m$. It follows that
\begin{align*}
2b^2&=a^2=(2m)^2=4m^2 \\
\implies b^2&=2m^2
\end{align*}
which implies, in turn, that $b$ must be even. However, it then follows
that $\frac{a}{b}$ cannot be an irreducible fraction, a contradiction.
\end{proof}
```

yields

Theorem 1. $\sqrt{2}$ *is irrational.*

Proof. For a proof by contradiction, suppose that $\sqrt{2}$ is rational. Then, we can express $\sqrt{2}$ as the fraction $\frac{a}{b}$ in its most simplified form, where both a and b are integers. Then,

$$\begin{aligned} \frac{a^2}{b^2} &= 2 \\ \implies a^2 &= 2b^2 \end{aligned}$$

so a must be even. Then, we can write a as $2m$, for some integer m. It follows that

$$\begin{aligned} 2b^2 = a^2 &= (2m)^2 = 4m^2 \\ \implies b^2 &= 2m^2 \end{aligned}$$

which implies, in turn, that b must be even. However, it then follows that $\frac{a}{b}$ cannot be an irreducible fraction, a contradiction. □

3 Writing Articles in LaTeX

An article in quantitative political science would not be complete without some tables and figures.

3.1 Tables

We can use the `tabular` environment to create a table like the following in LaTeX:

Name	Country of Origin	Favorite Greek Letter
Chitralekha Basu	Singapore	ψ
Trung Dang	Vietnam	δ
Rabia Malik	Pakistan	β
Mattan Sharkansky	Israel	σ
William Spaniel	USA	ϵ
Svanhildur Thorvaldsdóttir	Iceland	ω

The corresponding code is presented below, and as well as in the `SampleTable.tex` file on the course page. Download and open the file `SampleTable.tex` file on your computer.

```
\begin{center}
\begin{tabular}{|c|c|c|}
\hline
Name & Country of Origin & Favorite Greek Letter \\
\hline
\hline
Chitralekha Basu & Singapore & $\psi$ \\
\hline
Trung Dang & Vietnam & $\delta$ \\
\hline
Rabia Malik & Pakistan & $\beta$ \\
\hline
Mattan Sharkansky & Israel & $\sigma$ \\
\hline
William Spaniel & USA & $\epsilon$ \\
\hline
Svanhildur Thorvaldsd\'ottir & Iceland & $\omega$ \\
\hline
```

```
\end{tabular}
\end{center}
```

Some things to note.

1. The `tabular` environment, like the `array` environment, takes as its only mandatory argument the number of columns in the table and how these columns must be justified (left, right or center). Within this argument, we use the `|` command to specify if and where vertical borders should be drawn within a table. We may wish to only have borders along the perimeter of a table. Then, our `\begin{tabular}` command will take as its mandatory argument `{|ccc|}` instead of `{|c|c|c|}`.

2. As in the `array` environment, the alignment character `&` is used to separate content into different columns.

3. The `\hline` command is used to separate rows within a table. It can be omitted. Delete all of the `\hline` commands in your input file and compile the document. What happens?

4. The `\\` command is necessary to move content to the next row of the table.

5. You will notice that the table is nested within the `center` environment. Try commenting this out. What happens?

6. Now, write a couple of sentences immediately before your table. With the `center` environment still commented out, compile your document. What happens?

7. Math mode is not implied; if you want to include math in your table, you will need to switch to in-line math mode for that particular cell.

You can add a bit more information to your table by nesting the `tabular` environment within the `table` environment, which is a float (a kind of container for the table). The `table` environment allows you to to add captions, footnotes and numbers to each table. Try nesting your table within a `table` environment. You can now include the `\caption{}` command inside the `table` environment, which tells LaTeX to number the table. You can include a title for your table in the mandatory argument to this command. (If you leave the argument empty, your table will just be titled "Table 1:".)

There exist a number of useful packages that will allow you to design more complex tables. Some examples are: `longtable`, `multirow`, `tabularx`. We won't discuss them in

this course, but you can find out more about the functionality these packages offer (and about others too) here.

3.2 Figures

To include images in your document, you will need to first load the `graphicx` package. For the approach we have been using - which produces `.pdf` files as its final output - the external files you import will have to be in either `.jpg`, `.png` or `.pdf` (i.e. not `.gif` or `.bmp`) format. Download the file `Image.jpg` from the course page to your computer, and save it in the same directory as your `.tex` file. Load the `graphicx` package, and include the following code in your input file:

```
\begin{figure}[h!]
\centering
\includegraphics[scale=0.5]{Image.jpg}
\end{figure}
```

You should now have the following image in your compiled document.

- The command that is doing most of the work is `\includegraphics[scale=]{}`. This command requires as a mandatory argument the name and format of the image you are trying to import; it does not require a full file path for the image so long as it is saved in the same directory as your input file. As an optional argument, you can re-scale your image to a more appropriate size for your document.

- The `\centering` command and the `figure` environments are optional. To see this, try commenting them out.
- The `\centering` command is self-explanatory.
- The `figure` environment is another float (like the `table` environment). It creates a "container" for your image, and again, allows you to title and number the image using the `\caption{}` command. Try giving this image a caption.
- Finally, you will have noticed that in this case, the `\begin{figure}` environment includes an optional argument `[h!]`. This tries to persuade LaTeX to insert the figure HERE! Omit this argument from your code. What happens?

3.3 Headers, Abstracts, Acknowledgments...

Now we are getting to the miscellany.

Fancy Headers

When formatting, for example, problem sets, you may want to include key information like your name and the course code in a header to your document. The `fancyhdr` package allows you to do this. Load the `fancyhdr` package into your machine's operating memory, and include the command `\pagestyle{fancy}` in your preamble. Compile your document. What changes?

To actually add content to your header, you will need to use some combination of the `\chead{}`, `\lhead{}` and `\rhead{}` commands in your preamble. The command `\chead{}`, for instance, places whatever text you enter inside the curly braces at the center of the header. The `\lhead` and `\rhead` commands are analogous.

As an exercise, add your name and Alma Mater to the header of your document.

Now, comment out the command `\pagestyle{fancy}` and replace it with the command `\pagestyle{plain}`. What happens?

Title Pages, Etc.

When preparing a paper (or a second-year paper) for submission, you may want to include a title page, and maybe even an abstract and some acknowledgments.

To add a title to your document, enter the following code in your preamble and compile the document.

```
\title{My First Document}
\date{\today}
\author{Your Name}
```

For the above to actually appear in your output, you will need to add the `\maketitle` command to the **body** of your document. Do so and compile your document again.

- Note that in order for the `\maketitle` command to work, you need to define a title; an author and date, on the other hand, are optional.
- To add acknowledgments to your title, author or date, we nest the `\thanks{}` command inside that command. The text of your acknowledgments then appears as a footnote to your title, (or author, or date). You can also use `\thanks{}` to add institutional information or contact information to your title.
- Suppose your paper has more than one author. You can then amend the `\author{}` command as follows:

  ```
  \author{Your Name \and Her Name}
  ```

  Now, by repeating the `\thanks{}` command next to each author's name, you can enter separate contact details (or other information) for each author.

Suppose that you thought your title deserved more bombast. Then, you could create a title page. To do so, place the `\maketitle` command within the `\titlepage` environment. This moves all of your title and author information to a page of its own. On this title page, you can also include an abstract by nesting the `abstract` environment inside the `titlepage` environment. Try it out!

4 Presentations in LaTeX: The Basics

To create presentations in LaTeX, we use a package called `beamer`. The `beamer` package is loaded by specifying `beamer` as the argument to the `\documentclass{}` command in your preamble. To see an example of this, download the `Beamer.tex` file from the course page.

As before, all of the content of your presentation must be inside the `document` environment. However, each slide is now enclosed within a separate `frame` environment. Each `\begin{frame}` command takes as an optional argument (although within curly braces) the title for the slide.[1] Within each slide, you can use most of the environments we have discussed: `itemize`, `enumerate`, `tabular`, `figure`...

4.1 Theorems & Propositions

You can even define new `theorem` and `proposition` environments for your presentation, and include proofs of said theorems and propositions. The appearance of these theorems in the final output will look slightly different from what we have already seen. As an example, enter the following code into one of the slides in your input file.

```
\begin{theorem}
Pythagoras, the story goes, proved that $a^2+b^2=c^2$.
\end{theorem}
```

Unlike in the `article` class of documents, the `theorem` environment is already defined. Moreover, theorems are not automatically numbered; to number each theorem, you will need to issue an optional argument to the `\begin{theorem}` command containing the number. The environments `corollary` and `definition` are also pre-defined in beamer. However, to include propositions or lemmas—as distinct from theorems—in your presentation, you *will* need to define new environments.

4.2 Themes

As it stands, your presentation looks kind of boring. You can jazz it up by loading a `theme` for your presentation. To see the difference it makes, include the command

[1] This argument is optional in the sense that, if you were to omit it, your final output would still contain that slide, albeit sans title.

`\usetheme{Madrid}` in your preamble. What changes?

There is a large number of built-in themes for you to choose from (almost all named after cities - and yes, there is a theme called 'Rochester'). A partial list of available themes is given below.

- Antibes
- Bergen
- Copenhagen
- Frankfurt
- Madrid
- Warsaw

After picking a theme, you can also customize the color theme of your presentation by including the `\usecolortheme{}` command in your preamble. Again, a partial list of available color themes is given below.[2]

- albatross
- beaver
- dolphin
- lily
- orchid
- seagull

4.3 Title Page

You can add a title page to your presentation by the almost exactly the same procedure as in the `article` class. Provide information about the title and author in your preamble - as with the following code:

[2]Click here to view the full spread of different themes and color schemes (and combinations thereof) you can choose from.

```
\title{My First Presentation}
\author{No One \and Some One}
\date{\today}
```

Next, include as your first frame, the following–and voila!

```
\frame{\titlepage}
```

5 BibTeX & Bibliography Management

Finally, we briefly cover the basics of including citations and references in your compiled document with the aid of BibTeX. You can always enter references into your document manually, but the advantages of using BibTeX are considerable.

1. BibTeX comes bundled with LaTeX—so you don't need to download anything else.
2. In the long run, it will save you a lot of typing.
3. You will not have to type the author, title, date of publication, or page references for the book, article, or unpublished manuscript you want to cite more than once.
4. BibTeX and LaTeX will format (and re-format) your references according to whichever bibliography style you require.

We will not go into how BibTeX works, but we will go through the steps you must follow in order to use BibTeX to format your citations. Each BiBTeX database is a plain text file, and as with `.tex` files, you can create a `.bib` file using any text editor, so long as your references are formatted correctly. For example, within a `.bib` file, a properly formatted reference for a journal article will look like this:

```
@Article{Stokes1963,
  Title                    = {Spatial Models of Party Competition},
  Author                   = {Stokes, Donald E.},
  Journal                  = {The American Political Science Review},
  Year                     = {1963},

  Month                    = {June},
  Number                   = {2},
  Pages                    = {368--377},
  Volume                   = {57},

  Publisher                = {JSTOR}
}
```

5.1 Using JabRef

That said, it will save you time to use a reference management software like `JabRef` or `BibDesk` (Mac OS only), which will format your references for you. If not already

installed, download `JabRef` onto your computer and open it (link here). Here is how you can use JabRef to create a new entry in your database.

1. Clicking on the green plus sign on the toolbar near the top of your screen, and select the type of entry you are looking for (book, article, PhD thesis...?).

2. Fill in all of the required fields after clicking on the appropriate tab. If you want to include page references or a volume number, you can enter this information under 'Optional Fields'.

3. Frequently, you can download the BibTeX citation for a published book or journal article from the internet. For example, search for Anthony Downs' *The Economic Theory of Democracy* on Google Scholar. By clicking on 'Cite', you will be able to easily locate the BibTeX citation. Copy and paste the citation into the 'BibTeX Source' tab on JabRef. Click back to the 'Required Fields' tab. Now, all of the fields that can be filled out will have been filled out for you!

4. Once you have filled in all the information you have or want for a particular entry, click on the sparkling wand to the bottom left of your screen. This will generate a BibTeX key for you - which is all the information you will need to cite a reference in your `.tex` file.

5. To delete a bibliography entry, simply right-click on that entry and select 'Delete'.

5.2 LaTeX with BibTeX

- Download the `LaTeX.bib` file from the course page. Open the `SampleTable.tex` file again, and load the `natbib` package.

- At the bottom of your document, but before the `\end{document}` command, enter the following code:

```
\nocite{Downs1957}

\bibliographystyle{apsr}
\bibliography{LaTeX}
```

- Compile your document. LaTeX will tell you that you have undefined citations. Don't panic yet.

- Instead of 'Quick Build', select 'BibTeX' from the drop-down menu near the top of your screen, and compile your document again. You will not get any errors this time.
- Next, compile your document using the 'Quick Build' command for the second time. You will now receive an error, telling you that your citations may have changed.
- Compile your document for the third and last time. It will now compile correctly.
- Unfortunately, you will have to go through this each time you add a new citation to the same `.tex` file.

5.3 Adding Citations

All of this effort would be quite a waste without learning how to include citations in your input file. We have already encountered the `\nocite{}` command, which requires as its only argument a BibTeX key. This includes a citation in your references at the end of your document, even if you have not explicitly cited it in your text. A partial list of other useful citation commands is presented below.

Command	Output
`\cite{Downs1957}`	Downs (1957)
`\cite{Downs1957,Stokes1963}`	Downs (1957); Stokes (1963)
`\citep{Downs1957}`	(Downs, 1957)
`\citeauthor{Downs1957}`	Downs
`\citeyear{Downs1957}`	1957
`\citep[p.~215]{Downs1957}`	(Downs, 1957, p. 215)
`\citet[p.~215]{Downs1957}`	Downs (1957, p. 215)

6 Resources for LaTeX Users

There are numerous online and offline resources that you can consult to find out more about handy packages and commands (and how and when to use them). I have found that the LaTeX Wikibook provides excellent guidance on basically everything you might ever need. Also, Google is your friend.