

Our Second Latex Document

Capt. James T. Kirk

Physics Department
3601 Pacific Ave.

University of the Pacific
Stockton, CA 95211

Abstract

The purpose of this report is to introduce you to a few more aspects of Latex and get you ready to produce high quality scientific documents.

Introduction

Now that you are familiar with Latex, we will do some more fancy things.

First, notice the format of this document. The two column format was made by adding the option `twocolumn` in the brackets in the `documentclass` declaration. There are four main document classes: *article*, *report*, *letter*, and *book*. Each has special commands (lots for book!). In addition publishers often specify a class for their journal and supply you with a `.sty` file which formats your document for their journal. One day, we'll have a Pacific Physics `.sty` file for our lab reports!

Also, if you are following along in the Latex source file `second.tex`, then you will see many lines with `%` at the beginning. These are latex *comments*. Similar to the gnuplot comment `#` character, latex uses the percent sign: `%`. To put a real percent sign in your document, use backslash percent: `\%`.

You may have noticed that I can change the font to emphasize words, like **this** or *that*, or even **this** and this!. All we need to do is enclose the word in `{}`'s and include the appropriate font type at the beginning. You can change

	<code>{\em this}</code>	<i>this</i>
	<code>{\bf this}</code>	this
fonts like this	<code>{\tt this}</code>	this
	<code>{\sf this}</code>	this
	<code>{\underline {this}}</code>	<u>this</u>

You can also do `\textbf{this}` to get **this**, etc.

Environments and Modes

To do special kinds of presentation, like *tables*, or *mathematics* (among many others), Latex uses *environments* (which we sometimes call *modes*, as in “math mode”).

The way this works is that at some point in your document, you enter a new mode by typing

```
\begin{figure}
...stuff for figure mode goes here...
\end{figure}
```

as shown here for the `figure` mode. This is the environment you would use to set aside some space for inclusion of a figure. Another common mode is the `tabular` mode for making tables, which we’ll look at below.

Tables

Let’s look at one of the common modes: `tabular` for making tables.

In the previous document, `first.tex`, we made a simple table with several rows and columns. You can certainly make much more complicated tables, and below we make a simple change with some entries that span two columns.

Suppose that you want to create the following table showing your data, but you need some “cells” to span 2 columns (Trial 1 and Trial 2 below).

Trial 1		Trial 2	
Bin	Number	Bin	Number
1	3	5	9
2	4	6	3
3	7	7	1
4	1	8	2

This table was created with the following code:

```
\begin{center}
```

```

\begin{tabular}{|c|c|c|c|}\hline
\multicolumn{2}{|c|}{\bf Trial 1}
& \multicolumn{2}{|c|}{\bf Trial 2}\\
\hline
Bin & Number & Bin & Number \\
\hline
1 & 3 & 5 & 9 \\
\hline
2 & 4 & 6 & 3 \\
\hline
3 & 7 & 7 & 1 \\
\hline
4 & 1 & 8 & 2 \\
\hline
\end{tabular}
\end{center}

```

The line:

```
\begin\{tabular\}\{|c|c|c|c|\}\hline}
```

enters *tabular mode* and sets up the table, with 4 columns, separated by vertical lines, and column entries center justified: this is what `|c|c|c|c|` does.

The `\hline` makes a horizontal line at the top of the table.

Below this line is the first table entry with some special commands:

```

\multicolumn{2}{|c|}{\bf Trial} &
  \multicolumn{2}{|c|}{\bf Trial 2}\\

```

This defines a row with 2 columns that each span 2 of the previously defined columns. The argument 2 to `multicolumn` tells Latex how many columns to span by this entry. An `\hline` causes a horizontal line to be added to the table after this line.

Following this come the “normal” 4-column table entries:

```

1 & 3 & 5 & 9 \\
\hline

```

Notice that an ampersand (&) is used to separate the table fields and a LaTeX newline, `\\`, ends each line in the table. The `\hline` puts a horizontal line between table rows. Take it out and see the difference.

Try changing the `\begin{table}` line and see what happens. For example, what if you remove the vertical bars from the `|c|c|c|c|`? What if you only have some: `|cc|cc|`? How about `|l|c|c|r|`?

Also, I put a `\begin{center}` `\end{center}` pair around the table so that it will be centered in the text column.

Doing the Math

There are several ways to handle mathematics¹ in `LATEX`.

You can slip in a few mathematical characters **in a line of text**, such as “it follows that $\int \cos(x) dx = \sin(x) + C$ ”, or you can focus on multiple equations or formulas that you want to stand out, such as

$$\begin{aligned} \int_0^{\pi/3} \cos(x) dx &= \sin(x)|_{x=0}^{\pi/3} \\ &= \sin\left(\frac{\pi}{3}\right) - \sin(0) \\ &= \frac{\sqrt{3}}{2} - 0 \\ &= \frac{\sqrt{3}}{2} \end{aligned}$$

In either condition, you must be in **Math Mode** in `LATEX` to accomplish these tasks.

To enter math mode within the text of a paragraph (sometimes referred to as “inline” or “in-text”), you could use:

```
\begin{math} ...your math here... \end{math}
```

However it's usually much easier to use the shorthand notation of two \$ signs (the first \$ begins math mode and the second \$ leaves math mode),

¹Adapted from notes by Dr. Erin McNelis, Penn State University.

which is completely equivalent to the above longer method, as shown below:

`$ \dots your math here \dots $`

The other math mode is its own environment sometimes called “display mode” since it sets the mathematics apart, centered, on the page. There are several ways again to enter math display mode:

For a *single* equation

```
\begin{displaymath} \dots \end{displaymath}
\begin{equation} \dots \end{equation}
\begin{eqnarray} \dots \end{eqnarray}

$$\dots$$

```

are all equivalent. The last is my favorite since it’s the fewest characters to type. An example is:

```

$$\int x^n e^{ax^{n+1}} dx = \frac{e^{ax^{n+1}}}{a(n+1)}$$

```

which gives this

$$\int x^n e^{ax^{n+1}} dx = \frac{e^{ax^{n+1}}}{a(n+1)}$$

in your document.

Note that you can use `eqnarray` even if you have only one equation (more on this mode below).

Common Structures in Math Mode

Subscripts and Superscripts

Subscripts and superscripts are indicated by use of the `_` and `^` commands immediately following the item with this sub- or superscript in \LaTeX . If the elements in the subscript or superscript are *more than one character long*, they must be placed in curly braces, `{\dots}`. If something has both a subscript and a superscript, the order in which you use the commands does

not matter. Here are some examples of the displayed text and the \LaTeX code to generate them:

x^2	<code>x^2</code>
x^{5y}	<code>x^{5y}</code>
x^{5y^2}	<code>x^{5y^2}</code>
x_2	<code>x_2</code>
x_2^{5y}	<code>x_2^{5y}</code>
x_{5y}	<code>x_{5y}</code>
x^{5y_2}	<code>x^{5y_2}</code>
$x_1^{5y_2}$	<code>x_1^{5y_2}</code>

NOTE: It does not hurt, and it is good practice, to put single character sub- or superscripts inside the curly braces $\{\cdot\cdot\}$ themselves.

Fractions

You may still write fractions as a/b using the `/` key, but most fractions are better presented in a vertical format, with the numerator physically over the denominator. To do this in \LaTeX , use the `\frac{\{}{\}}` command (this command ONLY works when you are IN MATH MODE). This command has two arguments, each in its own set of curly braces, $\{\cdot\cdot\}$. First is the numerator, the second is the denominator. Here are some examples using the `\frac` command.

This one is in display math mode:

$$f(x) = \frac{x^2 + x - 2}{\sqrt{x^2 + y^2}}$$

The code which produced it is here.

```

$$
f(x) = \frac{x^2 + x - 2}{\sqrt{x^2 + y^2}}
$$

```

Using the `\frac` command from *inline* math mode as shown here “here is an inline fraction: $\frac{a}{2\pi}$ in the middle of a sentence”, produces this: “*here is an inline fraction: $\frac{a}{2\pi}$ in the middle of a sentence*”.

Roots

The `\sqrt` command is used to get square roots as well as other roots. To get a square root, simply use the `\sqrt{stuff under the radical}` syntax. To have an n^{th} root, put the value of the n in square brackets, $[n]$, between the `\sqrt` and the `{...}`. Some examples:

Here is the square root of 7: $\sqrt{7}$, and here is the 5th root of x^2+7 : $\sqrt[5]{x^2+7}$

This was produced with

Here is the square root of 7:~ `\sqrt{7}`~, and here is the 5th root of x^2+7 :~ `\sqrt[5]{x^2+7}`~

Notice in the “source file: `second.tex`”, that I’ve also used the `~` character in these lines. Putting a `~` in your latex file inserts about one character’s worth of blank space at that location.

Ellipsis

This could also be entitled the “dot dot dot” section. There are two types of horizontal ellipsis, the “low” ellipsis made by `\ldots` command and the “centered” ellipsis made by the `\cdots` command. The ellipsis can be vertical as well as diagonal too. See:

x_1, \dots, x_n	<code>\x_1, \ldots, x_n</code>
$a + \dots + d$	<code>a + \cdots d</code>
\vdots	<code>\vdots</code>
\ddots	<code>\ddots</code>

The `mbox` Environment in Math Mode

All text in math mode is italicized, unless it is a special math term such as `\cos(x)` or `\det(A)`, were these functions have their own L^AT_EX commands (see next section). If you want to have text presented in the normal typeset while you’re in math mode, simply include that text within the `mbox` environment as such `\mbox{ ... }`. Here’s an example. The following snippet:

$$\det A = \sum_{j=1}^n a_{ij} A_{ij} \quad \text{for any } i = 1, 2, \dots, n$$

is generated by typing

```
\begin{eqnarray*}
\det A = \sum_{j=1}^n a_{ij} A_{ij}
\hspace{0.25in}
\mbox{ for any } i = 1, 2, \cdots n
\end{eqnarray*}
```

Note that the * at the end of the word `eqnarray*` turns OFF equation numbering (making it equivalent to `$$`), and `\hspace{0.25in}` moves things 1/4 of an inch horizontally.

Special Mathematical Formulas in \LaTeX

Here is a small list of some useful special mathematical formulas that can be helpful in Math mode:

$\cos(x)$	<code>\cos(x)</code>
$\sin(x)$	<code>\sin(x)</code>
$\tan(x)$	<code>\tan(x)</code>
$\cot(x)$	<code>\cot(x)</code>
$\arccos(x)$	<code>\arccos(x)</code>
$\det(A)$	<code>\det(A)</code>
$\lim_{x \rightarrow 0}$	<code>\lim_{x \rightarrow 0}</code>
$\ln(x)$	<code>\ln(x)</code>
$\log(x)$	<code>\log(x)</code>
$\max_{t \in [a,b]}$	<code>\max_{t \in [a,b]}</code>
$\min_{t \in [a,b]}$	<code>\min_{t \in [a,b]}</code>

Special Mathematical Symbols in \LaTeX

There are MANY math symbols such as α , \int , ζ and many more.

As these are too numerous to type at the moment, please see the hand-out of L^AT_EX math for the most commonly used mathematical symbols and formulas.

Look in Course Materials—Latex for the file `mathcheatsheet.pdf` to find a list of common math symbols.

The Eqnarray Environment

The `eqnarray` environment is similar to a table in the fact that it defaults to having three columns whose entries are to be separated by `&`'s. One column is for the left hand side of an equation, one column for the equal or relation sign, and the last column for the right hand side of the equation. Remember, that if you do NOT want to produce numbers next to each equation, you simply put a `*` after the `eqnarray`. Here are a couple of examples:

```
\begin{eqnarray*}
\int_0^{\pi/3} \cos(x) \; dx & = & \sin(x) \Big|_{x=0}^{\pi/3} \\
& = & \sin \left( \frac{\pi}{3} \right) - \sin(0) \\
& = & \frac{\sqrt{3}}{2} - 0 \\
& = & \frac{\sqrt{3}}{2} \\
\end{eqnarray*}
```

$$\begin{aligned}
 \int_0^{\pi/3} \cos(x) \, dx &= \sin(x) \Big|_{x=0}^{\pi/3} \\
 &= \sin \left(\frac{\pi}{3} \right) - \sin(0) \\
 &= \frac{\sqrt{3}}{2} - 0 \\
 &= \frac{\sqrt{3}}{2}
 \end{aligned}$$

Notice in this example, that we are carrying down the operations on the right, so there is no need for a left hand side of the equation after the first line. Thus, no text preceded the `&` that separated the first from the second column. As in the `tabular` mode, each line ends with `\\` (a newline) except the last one.

The Array Environment

On occasions when you would like to align an equation or elements in an equation with more than the three allotted columns of the `eqnarray` environment, you may use the `array` environment within any of the centered display math mode environments. This essentially gives you a table setting for mathematical terms.

Just as with the tables, you must indicate the number of columns and the alignment of the columns immediately after starting the `array` environment. Here are a few samples.

```
\begin{eqnarray*}
\begin{array}{rcrcrcr}
2x_1 & + & 3x_2 & - & 5x_3 & = & 7 \\
3x_1 & & & - & x_3 & = & -2 \\
& & 4x_2 & + & 2x_3 & = & 5
\end{array}
\end{eqnarray*}
```

$$\begin{array}{rccccr} 2x_1 & + & 3x_2 & - & 5x_3 & = & 7 \\ 3x_1 & & & - & x_3 & = & -2 \\ & & 4x_2 & + & 2x_3 & = & 5 \end{array}$$

This is a good way to display matrices:

```
\begin{eqnarray*}
A = \left[
\begin{array}{rrrr}
3 & 4 & -1 & 2 \\
-2 & 3 & 5 & 7 \\
6 & -5 & -3 & 11 \\
1 & 2 & 6 & -3
\end{array}
\right]
\end{eqnarray*}
```

$$A = \begin{bmatrix} 3 & 4 & -1 & 2 \\ -2 & 3 & 5 & 7 \\ 6 & -5 & -3 & 11 \\ 1 & 2 & 6 & -3 \end{bmatrix}$$

Use of `\left` and `\right` in Math Mode

You may have noticed that we introduced some new notation in the previous example as well as the definite integral example earlier. In order to make the square brackets for our matrix (or our parenthesis for our tall fraction) match the size of the thing they were delimiting, we needed the use of the `\left` and `\right` commands. Essentially, if you need delimiters such as the `()`'s, `[]`'s, or `{ }`'s to stretch to fit around the object they're delimiting, you need to put a `\left` immediately before the opening delimiter and a `\right` immediately before the closing delimiter. This is also how we handle piecewise defined functions. There is a catch though. The `\left` and `\right` come in pairs and may never be stranded alone. With piecewise defined functions we want only one of the `{ }` pairing, so there's essentially no right delimiter. In these cases, simply use a period as the missing delimiter. See the following example for help:

```
\begin{eqnarray*}
f(x) = \left\{
\begin{array}{cl}
x^2 - 2x & \mbox{ if } x < 2 \\
x - 2 & \mbox{ if } 2 \leq x \leq 6 \\
\sqrt{4x} & \mbox{ if } x > 6
\end{array}
\right.
\end{eqnarray*}
```

which produces this:

$$f(x) = \begin{cases} x^2 - 2x & \text{if } x < 2 \\ x - 2 & \text{if } 2 \leq x \leq 6 \\ \sqrt{4x} & \text{if } x > 6 \end{cases}$$

Figures

Often you will want to include a figure such as a graph of data showing theoretical fits. First create the figure using gnuplot or MATLAB. Then include the graph in your report.

To do so, you need to “*use the package graphicx*”. You do this by including the line:

```
\usepackage{graphicx}
```

somewhere between

```
\documentclass{...}
```

and

```
\begin{document}.
```

Look at the source file for this document. You will find the `\usepackage{graphicx}` line near the beginning. This is an example of one of many packages for latex.

Graphs are now easy to include by using the `\includegraphics` command anywhere in the document. This keyword takes an argument to set the size and then the filename of the image. The particular image used here was of course created in Gnuplot, by setting the terminal type to type PNG: (`set term png`).

The `\includegraphics` command can display images which are in any of these formats:

- PDF
- PNG
- JPEG
- GIF (only with the latest versions of pdflatex)

Latex will put your figure where, in its own calculations, it can best fit your figure. Sometimes this is on the next page or the bottom or top of the current page. It can be frustrating... Notice the line `\begin{figure}[htb]`.

The options `[htb]` indicate to Latex to try to put the figure *Here*, then the *Top*, or the *Bottom* of the page, in that order.

The figure below was produced with the code

```
\begin{figure}[htb]
\includegraphics[width=12cm]{s8.png}
\caption{This is the graph of the Hydrogen 1420 MHz
line from {\bf S8}, recorded by the Pacific Physics
Dept's Radio Telescope}
\label{fig:S8}
\end{figure}
\end{center}
```

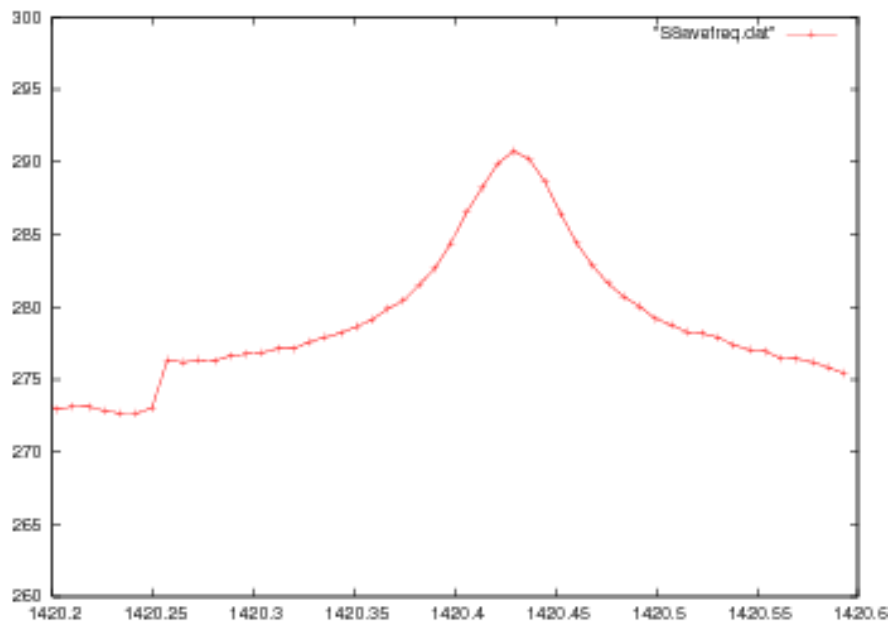


Figure 1: This is the graph of the Hydrogen 1420 MHz line from S8 recorded by the Pacific Physics Dept's Radio Telescope

Notice that `\includegraphics` command can take optional arguments such as `[width=12cm]` which sets the width of the figure on the page. You can

use centimeters (cm), millimeters (mm) or inches (in), as well as several other units. Separate options are separated by a comma as in:

```
\includegraphics[angle=45,width=52mm,scale=0.75]{myfig.jpg}
```

followed by the name of the image file enclosed in curly {} braces.

The `\includegraphics` command is usually encased in a `\figure` environment which sets off the figure by itself and provides the `\caption{}` command and the `\label{}` command.

The `\label{}` command allows you to give the figure an easy to remember nickname, which can be referred to anywhere in your document (like equation references). Just put the following line

```
...in Figure \ref{fig:S8}, we see...
```

which produces:

```
...in Figure 1, we see...
```

If you look back to the page where the figure was placed, you see that latex automatically labelled it as **Figure 1**.

The label can be set of characters. Here I used `fig:S8`; I like the `fig:` in front because it helps me remember that it's a figure. You could similarly use `eq:Newton` for your equation labels. Recall that we learned about equation labels in the last tutorial.

Remember that with all references, you have to run `pdflatex` twice to get figure and equation references correct. This is because on the first pass, `pdflatex` produces a file called `myfile.aux` with all the reference page numbers and locations. The second time `pdflatex` reads this file and uses it for the final version.

Verbatim Environment

Another useful environment is the `verbatim` mode, in which latex types things as they appear, without reformatting spaces, etc. An example is below, where I have included a code snippet.

```

int update() {
    int step, iters=0;
    double startaction, endaction, d_action();
    /* refresh the momenta */
    ranmom();
    /* do "steps" microcanonical steps" */
    for(step=1; step <= steps; step++){
        update_u(epsilon*(0.5-nflavors1/8.0));
        clear_latvec( F_OFFSET(xxx1));
        grsource_imp( F_OFFSET(phi1), mass);
    }
    ...
}

```

As you can see, the text inside the `verbatim` environment is printed in teletype font so that it stands out, and text spacing is preserved. You enter `verbatim` mode as you do with other modes:

```

\begin{verbatim}
...your verbatim stuff here...
\end{verbatim}

```

For small amounts of verbatim text the `\verb` command is very useful. In the expression `\verb#some text#` the `#` character acts as a switch. All the text that comes between the two `#` characters is output verbatim. For example, latex commands are not interpreted (so you can put `$'s` & `'s %'s`, etc. between the `#'s`). Any character, which does not occur in the string `some text`, other than the `*` character, can be used as the switch. The equals sign `=` and the pipe `|` are commonly used as switches.

There is a star version `\verb*` of the of the `\verb` command. It behaves exactly the same as the `\verb` command except spaces are rendered visible. They are replaced by the `␣` character. Thus the command

```
\verb*#Here is some sample text.#
```

produces the output

```
Here␣is␣some␣sample␣text.
```

Running pdf_latex

To produce this document, you run `pdflatex second.tex`. You may be surprized at some of the output:

```
Underfull \hbox (badness 6141) in paragraph at lines 422--422
[]\OT1/cmr/bx/n/10 Special Math-e-mat-i-cal For-mu-las in
```

```
Underfull \hbox (badness 10000) in paragraph at lines 445--445
[]\OT1/cmr/bx/n/10 Special Math-e-mat-i-cal Sym-bols in
[4]
```

These are warnings from pdf_latex saying that it is a bit difficult to layout your document. Usually these errors are not a problem, as long as you get

Output written on `second.pdf` (7 pages, 170908 bytes).

at the end. If you *DO* get errors, latex will stop and say something like this:

```
! Undefined control sequence.
l.710 \ssection
           {Conclusions.}
?
```

This means that latex encountered a semi-fatal error on line 710 of your document. It is trying to tell you what the offending command is (in this case it doesn't recognize my misspelling of `ssection` with 2 "s"s.

At the ? prompt, type `x` to exit latex and fix the error, or `r` to continue and ignore the error as best it can.

We'll learn more about deciphering latex errors later.

Odds and Ends

If you read tutorials about Latex on the web, you'll find that there are other routes to produce a final latex document. In this course, I've streamlined our discussion and only taught you the easiest:

- `pdflatex myfile.tex → myfile.pdf`

However, the original latex package used the command:

- `latex myfile.tex → myfile.dvi`

One would then use a *DVI Viewer*, an application that can display the DVI file (such as `xdvi`).

Finally, one ran the `dvips` program:

- `dvips myfile.dvi → myfile.ps`

which produced a *Postscript* file that could be sent to the printer (Postscript is the predecessor to PDF). There are also programs like `dvi2pdf` and `ps2pdf` which convert Postscript to PDF. (you could also use `convert`—see below!) If you use this route, with the `latex` command, then `\includegraphics` will only be compatible with Postscript (or PS or EPS) figures. If this all sounds confusing, don't worry about it. Just use `pdflatex` as I've shown you.

Since PDF has become a very popular standard for documents, and `pdflatex` does everything in one go (or maybe two if you have references), I've decided to teach you that method.

Convert

As part of the installation, I had you get the *ImageMagick* program `convert`. If you have a file in a non-`pdflatex`-compliant format, you can easily convert it to one of the above, by typing

```
convert myimage.booya myimage.jpg
```

at the xterm prompt. The first filename, `myimage.booya`, is the name of the image you have. Simply give the second filename the suffix of the format you would like, and hit enter. *Convert* will translate it for you. Since it converts to and from PDF, it can be handy to send an image file to someone who is having trouble opening it.

Conclusion

As you can see, Latex is rather powerful, especially for producing documents which contain a significant amount of mathematical formulae. Here we have just scratched the surface, but I hope you have learned enough so that you

can produce a basic, but sophisticated looking paper. From this start, you should be able to teach yourself more. There is a lot of documentation online about Latex, as well as several books.

One really good introduction to Latex is available at

`http://www.ctan.org/tex-archive/
info/lshort/english/lshort.pdf`

called *A Not-So-Short Introduction to Latex2e*. I've put a link to it on our course webpage.

Be sure to get the `mathcheatsheet.pdf` that I've also put up on the course webpage. showing many of the math special symbols for Greek letters and other math symbols.