PHYS 87: Notes for Seminar on How to Make Beautiful Technical Documents with LaTeX

Benjamín Grinstein
Department of Physics
UCSD

March 27, 2020

Preface

This book is based on a Freshman Seminar I have taught at UCSD since the Winter-2015 quarter of instruction. It is organized by subject, and it follows roughly the organization of the material in the LaTeX Wikibook that I encourage the students to explore. The lectures themselves are not organized this way, as you can see from the lecture slides available from my course website that can be found in the physics department course web pages. This is because, experience shows, it is incredibly boring to students to spend several lectures in a row learning math typesetting. So instead I mix math with other topics.

I encourage students to ask questions. I often do not know the answer and show them how to look them up — I often do that even if I do know the answer. Students quickly figure out they can look up answers too. The LaTeX Wikibook is a good first stop. Internet search engines are very good at finding questions similar or related to the one posed by the student in blogs, and hopefully they come with good answers. Typically the best answers are from StackExchange.

My course slides contain examples that the students copy into their computers to start learning LaTeX. After a couple of weeks I include "exercises" which are little challenges. It is important for the students to try these exercises: it is very different to copy somebody else's work — and think one knows how one would do this of ones own initiative — than to do the work independently. That said, I allow, even encourage, discussion among students, since I realize they speak a common language and they communicate and understand better their peers than me.

Contents

Chapter 1

Getting started

This is a quick course on how to use LaTeX to produce typeset quality documents. LaTeX is particularly powerful in producing typeset quality documents with complex mathematical expressions, but it can do much more. It is also much more difficult to learn to use than popular comercial editors, like Microsoft Word or OpenOffice Writer, with their mathematical expression drop-down menus. But once you pass the steep learning curve it is easy to use and the results are fantastic.

These notes are typeset with LaTeX. This gives you a first impression of what LaTeX can do. Here is an equation, just to show how well it looks

$$\langle \phi(x_1) \cdots \phi(x_n) \rangle = \frac{1}{Z} \int [d\phi]\, e^{i \int d^4x\, \mathcal{L}(x)} \phi(x_1) \cdots \phi(x_n)\,.$$

Being designed with scientists in mind, LaTeX helps you number equations and refer to them in text, organize references and cite them in text, include graphics and tables, and much of this is done automatically. Once you get used you will want to write even simple memos with LaTeX, but if all you need is to write simple memos you probably don't want to spend the time and effort it takes to learn it.

The course is hands-on. You are expected to bring your laptop to class so we can work real time with LaTeX and learn together.

There are many resources at your disposal for learning LaTeX. You can find a variety of books in the library and the bookstore. There really is no need to spend your hard earned money on these books. For a textbook we will use the readily available LaTeX Wiki book. By the way, this text in red is a live link: click on it. You can consult it on-line or download the pdf version, or both. For our purposes this not only suffices, but it has too much information. I will guide you through it (that's what you pay me to do!) but you are welcome and encouraged to do more: read it and get additional insights into how the program works and its capabilities.

Once we download and install LaTeX you will find additional documentation and tutorials with the distribution. These can also be of great help.

Finally, you can find answers to many conundrums (as in 'how do I typeset "$\mathcal{A}$"') by searching through the web. Type into your search engine, for example, '`LaTeX superimpose symbols`.' When I do this on Google the first entry that comes up is a link to the TeX Stack Exchange that asks and answers the question *How do you superimpose two symbols over each other?*. Try it!

1.1 Installation

The engine for LaTeX, together with many of the packages that make it all work and the links/permissions/directories setups, makes installation by hand a big chore. Installers are available for all platforms that make it easy by automating the process. But you must be patient, its a big download! The desktops in class all have LaTeX installed, so you do not need to worry about this if you do not plan to install this on your computer (be it a laptop or a desktop). It is likely that you will want to use LaTeX from now on to write-up your technical reports (homework, projects, thesis, etc) and you probably would find it easier to do that on your own computer: the software is free and this would be the right time to install it.

1.2 Editors, Command Line Terminal Windows and Viewers

Before we install LaTeX, we will install TeXworks. The reason for doing it in this order is simply so that you don;t forget to install it before the first class. TeXworks is a very small program that provides you an interface to LaTeX — without LaTeX installed, TeXworks does next to nothing. You can download it (for free) from www.tug.org/texworks. Downlaod the installer for Windows, or disk images for Mac, or the appropriate version for GNU/Linux (Ubuntu, openSUSE, Debian, Fedora, Arch Linux). Note that some of teh installers of the LaTeX package (see below) icnlude and install TeXworks. So you may try installing LaTeX frist and check for installation of TeXworks; come back here if it was not installed.

TeXworks is the editor we will use by default. That means it's the editor I will use to demonstrate stuff in class. It comes with an integrated viewer making the process of previewing the output to your documents straightforward.

However, students with some coding experience may be accustomed to and prefer some other text editor. If taht si your case, you may use that isntead. By a text editor I mean one that can produce and modify plain text files. Microsoft Word and OpenOffice Writer are not good for this (unless you insist on saving as plain text file). But Emacs, Atom and Vim are, for example. If you know these or something like these, and your are well versed in how to use them, feel free to stick

to them. If you do that you will have to run LaTeX commands from a command line window:

- Windows: Open the Command Prompt window by clicking the Start button Picture of the Start button, clicking All Programs, clicking Accessories, and then clicking Command Prompt. Alternatively, Here's another way to open a Command Prompt window: Click the Start button Picture of the Start button. In the Search box, type Command Prompt, and then, in the list of results, double-click Command Prompt.
- Mac OS X: From the Finder Go>Utilities then double click Terminal (or Shift-Command-U will open the Utilities window and then double-click Terminal)
- Unix. No explanation needed.

If you are not using TeXworks you will also need a pdf viewer. Everyone has Acrobat Reader. But it's ok to use anything else (like Skim and Preview on the Mac or Nitro and Sumatra on Windows).

My favorite is Emacs. The main reason is that I have used it for ages. But it is ultra-customizable and has a myriad bells and whistles. Aquamacs is a Mac version which I like. A windows version is available from https://ftp.gnu.org/gnu/emacs/windows/.

1.3 Mac

Go to `http://tug.org/mactex/` home to the MacTeX-2019 Distribution. If your Mac is running Sierra (Mac OS 10.12) or higher and runs on Intel processors scroll down to below "The current distribution is MacTeX-2019", and click the link MacTeX Download; from the next page simply click on MacTex.pkg. This is a 3.9G download: unless you have great connection at home you should do this on campus (but not in Spring 2020!). And even then, you must be patient! I have done this many times and never had a problem, it is straightforward.

If you are running an older version of the operating system (pre OS 10.12), from `http://tug.org/mactex/` go down to the last line that starts "To Obtain Older Versions of ..." and click on "click here". Follow instructions.

Click the Donate link on the title bar to donate some money. Giving them ten bucks helps a lot and it is certainly much cheaper than buying a textbook or comercial software.

If space is at a premium, or you don't have the time or the connection speed to deal with the large download above you can try BasicTeX: scroll down to find Smaller Download. This is a 110M download which gives you all you need to get

started. It has a very limited set of packages, but these can be added as needed using TeXLive Utility (see below). **I strongly recommend you install the full MacTeX-2019 distribution.**

The editor we will use is TeXworks, which you can get from `http://www.tug.org/texworks/#Getting_TeXworks`. This is not included in either the full nor basic distributions. So go ahead and download and install this.

1.4 Windows

Go to `http://miktex.org/` home to the MiKTeX project page. I recommend you first read Howto: Install MiKTeX on your Windows computer; from the home page click on Downloads and then scroll down to tutorial. One important thing to note, on the last step of the installation wizard you will be asked to select "Preferred paper" and "Install missing packages on the fly"; for the former select "letter" (it may be called "letterpaper" or "lettersize") and for the latter select "Always". and Click on Download on the title bar and in that page find the Recommended Download. This does not download MikTeX but rather an installer. It should be obvious how to run it (and hopefully you have read the detailed instructions under "Howto: ...". The installation Wizard asks for where to install MiKTeX (the default, `C:\Program Files (x86)\MiKTeX 2.9` is fine) and then, to repeat, for "Preferred paper" (default `A4`) and "Install missing packages on the fly" (default `Ask me first`). Change the preferred paper to *letter* and for on the fly choose *Always*.

You should have now a new collection of apps, among them the MikTeX Package Manager. Things are easier if you follow the detailed instructions for working with the package manager. Open this app (either find the app or Start>Programs>MiKTeX 2.9>Browse Packages). Select all packages then click the plus sign on the top left, right under "File." You can also accomplish this from entries under the menu bar (for example, under "File" you will see an "Install All Packages"). If you cannot install all packages at once, try installing first just one, then try again.

The editor we will use is TeXworks, which is included in the MiKTeX distribution.

Click the Give Back link on the title bar to donate some money. Giving them ten bucks helps a lot and it is certainly much cheaper than buying a textbook or comercial software.

1.5 Unix

If you have a unix system I have to assume you are at ease with getting into the guts of your computer and I can just direct you to the Tex Live -Quick Install at http://www.tug.org/texlive/quickinstall.html or to the MikTeX downloads page.

Click the How you can help link on the home page http://www.tug.org/texlive/ to donate some time and/or money. Giving them ten bucks helps a lot and it is certainly much cheaper than buying a textbook or comercial software.

1.6 Compilation

So why do we need an editor? You can think of LaTeX as a markup language (and if you do not know what that means, do not worry, read on). A compiler takes an input file (*e.g.,* an input document) which contains a set of instructions for the computer and produces from it an output file. This output file can be, for example, a pdf file that you can print or read with a pdf viewer, like Acrobat reader. So in order to produce the desired output you must first create the input that contains the instructions. The editor is used to create and change (edit) the input file.

The input file must have extension `.tex`. For example, you may create a file called `myfirstfile.tex`. The name of the file is anything that your computer system accepts as a name for a document, with, of course, extension `.tex`. Some systems have a limit on the number of characters you can use in the name, and some characters may not be allowed.

The output file that is produced by the compiler will have extension `.pdf` if you compiled so that a pdf file is produced. Another common extension for an output file is `.dvi` that stands for *device independent*. A dvi file has to be processed to produce something readable. The TeX distribution includes commands like `dvips` that produces a postscript file (extension `.ps`) out of the dvi file, and `dvipdf` that produces a pdf file out of the dvi file.

You will get used to the process pretty quickly. It takes two steps. First you produce your input `.tex` file and then you process it, or "LaTeX it" to produce an output file, which is typeset quality. If your output file is a pdf file you are done. If your output is a dvi file you need one more step, to produce the ps or pdf file.

Chapter 2

Basics

2.1 First Project

Open TeXworks. Type the following

```
\documentclass{article}

\begin{document}

Hello world!

\end{document}
```

Then save the file. In TeXworks this can be done by clicking on the icon for *save file*, by going to the File menu and selecting *Save* or *Save as ...* or by using the shorthand for the latter, something like command-S on the Mac or command-S on Windows. I saved mine as `first_project.tex`. The extension is added automatically: I only needed to type *first_project* and the file is saved as `first_project.tex`.

To produce an output file you click on the green button with a triangle, on the top-left of TeXworks. Make sure that the drop-down menu next to the green compile button has *pdfLaTeX selected*. If all goes well a console window will pop up momentarily, then disappear and a second window called *first_project.pdf* appears. It should be a mostly blank page with the words Hello world! displayed about a quarter distance for the top.

Congratulations, you have completed your first project!

2.1.1 Debugging

Now, let's insert a mistake into the input file and learn how to deal with it. Go back to your first project input file and type a line right below `Hello world!` as

follows:

```
\documentclass{article}

\begin{document}

Hello world!
\parr

\end{document}
```

Now save your work (always save your work before you compile!) and compile. The green button turn red with a cross, to indicate a problem. The console window *Console output* will appear below your typed text, but this time it will stay up. It should look like this:

```
This is pdfTeX, Version 3.14159265-2.6-1.40.15 (TeX Live 2014) (preloaded
 format=pdflatex)
 restricted \write18 enabled.
entering extended mode
(./first_project.tex
LaTeX2e <2014/05/01>
Babel <3.9l> and hyphenation patterns for 79 languages loaded.
(/usr/local/texlive/2014/texmf-dist/tex/latex/base/article.cls
Document Class: article 2014/09/29 v1.4h Standard LaTeX document class
(/usr/local/texlive/2014/texmf-dist/tex/latex/base/size10.clo))
(./first_project.aux)
! Undefined control sequence.
l.6 \parr

?
```

The important thing to note is, first, the compiler stopped because there was an `! Undefined control sequence.` and, second, that the problem is `l.6 \parr`, meaning it is on line 6 of the file and that the problem is with `\parr`.

The question mark at the end of the console's output indicates the program has stopped and is awaiting further instructions. It is saying, man, I got stuck, now what? You will notice that below the console window there is a one-line text box into which you can type your instruction to the compiler. There is a variety of thing you can tell the compiler, we will go through three options.

Option 1: Help In the text box type `?-ret`, that is a question mark and hit return. You should see

```
? ?
Type <return> to proceed, S to scroll future error messages,
R to run without stopping, Q to run quietly,
I to insert something, E to edit your file,
1 or ... or 9 to ignore the next 1 to 9 tokens of input,
H for help, X to quit.
?
```

The first question mark was there before, the second one is the one you typed. The next few lines remind you of your options. The last question mark tells you that the program is waiting for your input again.

Option 2: Proceed Now click in the text box and hit return without typing anything. The program runs through the end, and the output is produced. It looks exactly as before, with "Hello World!" near the top. The following is added to the console window

```
[1{/usr/local/texlive/2014/texmf-var/fonts/map/pdftex/updmap/pdftex.map}]
(./first_project.aux) )</usr/local/texlive/2014/texmf-dist/fonts/type1/public/a
msfonts/cm/cmr10.pfb>
Output written on first_project.pdf (1 page, 11915 bytes).
SyncTeX written on first_project.synctex.gz.
Transcript written on first_project.log.
```

There is useful information in this, but let's not get bogged down with all of it now. Notice only that an output file, `first_project.pdf`, was written. You can find it in the directory (folder) where you saved your project. So the output is not just on the output screen. It is in a file that you can share with others and print.

Option 3: Quit Sometimes you know that something has gone horribly wrong and you just want the compiler to stop immediately, aborting the compilation. To see how this works, hit the green Typeset button again and when compilation stops, type `x-ret` into the type-box. The console then adds below the question mark the lines:

```
No pages of output.
Transcript written on "first_project.log".
```

Since you have told the compiler to stop, there in no output.

2.1.2 Fixing it

Go back tou your project. Remove one `r` from `\parr`. Your input file reads:

```
\documentclass{article}

\begin{document}

Hello world!
\par
\end{document}
```

Typeset this. No error messages any more. Output is produced. The sequence of characters `\par` is an allowed sequence. We will study this in detail later. For now what matters is that we know what to do if we get stuck! We use the compilation window to find where the error is, fix it by editing the input document and Typeset again.

2.2 Syntax Basics

From here on these notes are more like bullet points. I will basically list the topic and we will work with it in class. I will include form time to time specific instructions, but most instruction will come verbally and on the blackboard, and we will continuously experiment with the knowledge on our laptops.

- Characters: A–Z, a–z, 0–9, some symbols (comma, period, etc) excluding reserved characters (see bellow)
- White space
- Empty lines
- Reserved characters: # $ % ^ & _ { } ~ \
- Latex Groups:

  ```
  {
  \bf This is bold.
  }
  This is no longer bold.
  ```

- LaTeX commands: start with \ followed by letters only, the `name` of the command.

  - Commands with arguments and options. Example

    ```
    \hspace{1in}
    ```

    The \hspace command takes as an argument a distance measure, in this case 1 inch. It leaves 1 inch of space in your text. Try this in your Hello World! document, forcing 1 inch of space between the two words:

    ```
    \documentclass{article}

    \begin{document}

    Hello \hspace{1in} world!

    \end{document}
    ```

  - Switches. In the example above, \bf is the boldface command. It is a switch. It changes the typesetting into boldface characters. The switch is limited to the group enclosed in curly brackets { and }

- Comments: anything following % in the current line is ignored. Add a comment to your first project now! Example:

  ```
  \documentclass{article}

  \begin{document}

  Hello \hspace{1in} world! %This won't show

  % neither will this
  \end{document}
  ```

- LaTeX Environments:

  ```
  \begin{environmentname}
  text to be influenced
  \end{environmentname}
  ```

  Let's try this with the *center* environment:

  ```
  \documentclass{article}
  \begin{document}
  \begin{center}
  Hello world!
  \end{center}
  \end{document}
  ```

2.3 Under the hood

Look in the folder where you saved `first_project.tex` or whatever you called your LaTeX file. In addition to `first_project.tex` you will find a number of other files, among them:

`first_project.pdf` Is the fruit of your hard work. It is a pdf file (which , by the way, stands for Portable Document Format). You can view this file with Acrobat Reader, or other viewers, *e.g.*, Preview or Skim for the Mac, Nitro or Sumatra for Windows. You can email it to your buddies. You can post it for downloading from your web page. Texworks has an embedded pdf viewer which is what allowed you to see this file within that application.

`first_project.log` Is a plain text file, and there are many apps that allow you to view plain text files. It logs the output of the LaTeX engine (more about this below), including warnings and error messages. Useful for debugging.

`first_project.aux` This is an auxiliary file. The LaTeX engine uses this to keep track of some information. We will go back to this later. Th eLaTeX engine needs to keep track of items that are numbered, like equations, figures and bibliographical references, and this is the place in which it does it. Don't worry, until you become a LaTeX expert you don't have to look inside this file.

So here's the thing. Texworks is not LaTeX. It is an app that uses many other apps. In particular it uses a LaTeX app that exists in your computer system, but that you cannot open by double clicking. LaTeX sits there in your computer waiting for some other app to use it. (Actually, for those of you who are techies, you can access it from a Terminal(Mac)/Command(Windows) window: you simply type the command `pdflatex` followed by the name of your project and hit return). Texworks does have a primitive editor: an "editor" is an app that allows you to create a plain text file and type text into it — that is, to create your project and save it as `*.tex`. (Techies: you can use your favorite editor, such as emacs, vim, gEdit or nano, and then run latex from the terminal/command window, without ever using Texworks). And Texworks has a pdf viewer to see the output of LaTeX, which is a pdf file. In fact there are many apps that do the same tasks as Texworks, and some have more bells and whistles and have a neater appearance. But Texworks runs almost identically on windows and mac, so its good for this course.

Chapter 3

My First Full Article

3.1 Document classes

We have seen we always start with `\documentclass`. This command takes two arguments:

```
\documentclass[options]{class}
```

As you have seen you do not have to specify options. We have used the class `article`. There are other classes, such as `book` and `letter`. See the complete list in the LaTeX Wikibook. We will stick with `article` for now.

There are many options. You can enter them in a comma separated list. Let's try it. Start a new document with

```
\documentclass[letterpaper,notitlepage,11pt]{article}
\begin{document}
Hello world again!
\end{document}
```

Now change `12pt` for `11pt`.

See the complete list of options in the LaTeX Wikibook.

3.2 Using packages

Packages enhance the capabilities of LaTeX. There is a huge number of packages available. We can see a listing of packages installed in our systems with TeXLive Utilities (Package Manager).

Let's do an example:

```
\documentclass[letterpaper,notitlepage,11pt]{article}

\usepackage{color}
\begin{document}
Hello {\color{red} world} again!
\end{document}
```

Packages can take options too:

```
\usepackage[options]{packagename}
```

You can call many packages:

```
\usepackage[options]{packagename1}
\usepackage[options]{packagename2}
\usepackage{packagename3,packagename4,packagename5}
```

Try this:

```
\documentclass[letterpaper,notitlepage,11pt]{article}
\usepackage[margin=2cm]{geometry}
\usepackage{color}
\begin{document}
Hello {\color{red} world} again!
\end{document}
```

3.3 Top Matter

Now we add a title, author, date and abstract to our scientific breakthrough:

```
\documentclass[letterpaper,notitlepage,11pt]{article}
\usepackage[margin=2cm]{geometry}
\usepackage{color}
\begin{document}
\title{The Origin of Species}
\author{Charles Darwin}
\date{December 2014}
\maketitle
\begin{abstract}
  I copy-pasted this from wikipedia in       http://en.wikipedia.org/wiki/On_the_Origi
  want): On the Origin of Species, published on 24 November 1859, is a
  work of scientific literature by Charles Darwin which is considered
  to be the foundation of evolutionary biology. Its full title was On
```

```
  the Origin of Species by Means of Natural Selection, or the
  Preservation of Favoured Races in the Struggle for Life. For the
  sixth edition of 1872, the short title was changed to The Origin of
  Species. Darwin's book introduced the scientific theory that
  populations evolve over the course of generations through a process
  of natural selection.
\end{abstract}

Hello {\color{red} world} again!
\end{document}
```

Try this: change the date to \date{\today}

3.4 Sectioning and Paragraphs

3.4.1 Paragraphs

Now you can add some text to the "article." Let's all do the same. Go to http://en.wikipedia.org/wiki/On_the_Origin_of_Species and select and copy the first three paragraphs, from where it says **On the Origin of Species,** *published ...* until *... become the unifying concept of the life sciences.* Then paste this in your document, and remove the *Hello world again!* line.

You will see that you code contains a blank line between paragraphs. Play with removing the blank line and with inserting extra blank lines and look at the output. Now you know how to make paragraphs. and that extra blank lines are ignored.

3.4.2 Line Breaks

Suppose you want to break a line, without starting a new paragraph.

Insert \\ after three words (make sure you leave at least one blank space after this).

Insert instead \break after three words (make sure you leave at least one blank space after this).

Insert instead \hfil\break after three words (make sure you leave at least one blank space after this). We'll talk about \hfil in more detail later, but obviously it fills up the blank space.

3.4.3 Sections

Your article may need different sections and subsections. Insert before the first paragraph the following:

```
\section{Introduction}
```

Your document should look something like this:

```
\documentclass[letterpaper,notitlepage,11pt]{article}
\usepackage[margin=2cm]{geometry}
\usepackage{color}
\begin{document}
\title{The Origin of Species}
\author{Charles Darwin}
\date{\today}
\maketitle
\begin{abstract}
  I copy-pasted this from wikipedia (copy-paste what you
  want): On the Origin of Species, published on 24 November 1859, is a
  work of ...  blah blah ... a process
  of natural selection.
\end{abstract}

\section{Introduction}
On the Origin blah blah

blah blah  life sciences.
\end{document}
```

You may also need sections of sections, or *subsections*. Type in

```
\subsection{Generalities}
```

right after the `\section{Introduction}` line and again between the first and second paragraphs

```
\subsection{More stuff}
```

Play with leaving blank lines before and after these commands.
There are more sectioning commands. See the Wikibook for the complete list.

3.4.4 Page break

Now insert between the second and third paragraphs the command

```
\newpage
```

My document looks like this:

```
\documentclass[letterpaper,notitlepage,11pt]{article}
\usepackage[margin=2cm]{geometry}
\usepackage{color}
\begin{document}
\title{The Origin of Species}
\author{Charles Darwin}
\date{\today}
\maketitle
\begin{abstract}
  I copy-pasted this from wikipedia (copy-paste what you
  want): On the Origin of Species, published on 24 November 1859, is a
  work of scientific literature by Charles Darwin which is considered
  to be the foundation of evolutionary biology. Its full title was On
  the Origin of Species by Means of Natural Selection, or the
  Preservation of Favoured Races in the Struggle for Life. For the
  sixth edition of 1872, the short title was changed to The Origin of
  Species. Darwin's book introduced the scientific theory that
  populations evolve over the course of generations through a process
  of natural selection.
\end{abstract}

\section{Introduction}
\subsection{Generalities}
On the Origin of Species, published on 24 November 1859, is a work of
scientific literature by Charles Darwin which is considered to be the
foundation of evolutionary biology. Its full title was On the Origin
of Species by Means of Natural Selection, or the Preservation of
Favoured Races in the Struggle for Life. For the sixth edition of
1872, the short title was changed to The Origin of Species. Darwin's
book introduced the scientific theory that populations evolve over the
course of generations through a process of natural selection. It
presented a body of evidence that the diversity of life arose by
common descent through a branching pattern of evolution. Darwin
included evidence that he had gathered on the Beagle expedition in the
1830s and his subsequent findings from research, correspondence, and
experimentation.[3]
\subsection{More stuff}
Various evolutionary ideas had already been proposed to explain new
findings in biology. There was growing support for such ideas among
```

```
dissident anatomists and the general public, but during the first half
of the 19th century the English scientific establishment was closely
tied to the Church of England, while science was part of natural
theology. Ideas about the transmutation of species were controversial
as they conflicted with the beliefs that species were unchanging parts
of a designed hierarchy and that humans were unique, unrelated to
other animals. The political and theological implications were
intensely debated, but transmutation was not accepted by the
scientific mainstream.
\newpage
The book was written for non-specialist readers and attracted
widespread interest upon its publication. As Darwin was an eminent
scientist, his findings were taken seriously and the evidence he
presented generated scientific, philosophical, and religious
discussion. The debate over the book contributed to the campaign by T.
H. Huxley and his fellow members of the X Club to secularise science
by promoting scientific naturalism. Within two decades there was
widespread scientific agreement that evolution, with a branching
pattern of common descent, had occurred, but scientists were slow to
give natural selection the significance that Darwin thought
appropriate. During "the eclipse of Darwinism" from the 1880s to the
1930s, various other mechanisms of evolution were given more credit.
With the development of the modern evolutionary synthesis in the 1930s
and 1940s, Darwin's concept of evolutionary adaptation through natural
selection became central to modern evolutionary theory, and it has now
become the unifying concept of the life sciences.
\end{document}
```

3.5 Color help in LaTeX editing of source

It helps finding stuff out if the document you are editing has the commands and environment delimiters in different color than normal text.

In TeXworks go to the pull down menu from the titlebar *Format>Syntax Coloring>LaTeX*. Your commands should appear in blue and the environments in dark green

Chapter 4

Math

There is a lot of stuff here. We take it one step at a time. Steep learning curve!

4.1 Equations

4.1.1 Inline equations

Our third project starts with (make sure you select *Format>Syntax Coloring>LaTeX* from the title bar form every project from here on!)

```
\documentclass[11pt]{article}

\begin{document}
My fist equation is $F=ma$.
\end{document}
```

Now add to the last line:

```
My fist equation is $F=ma$. My second is \( a=F/m\).
```

4.1.2 Displayed equations

For the basic displayed equation, change the above as follows

```
My fist equation is $F=ma$. My second is \[ a=F/m\].
```

Let's add more text:

```
\documentclass[11pt]{article}

\begin{document}
```

```
My fist equation is $F=ma$. My second is \[ a=F/m\].
The period is in the wrong place. I should place it inside the
displayed equation. I want to write enough that there will be a few
lines after the displayed equation blah blah blah.
\end{document}
```

Place period in correct place.

Another version:

```
\documentclass[11pt]{article}

\begin{document}
My fist equation is $F=ma$. My second is
\begin{equation}
 a=F/m.
\end{equation}
The period is in the right place. I have place it inside the
displayed equation. I want to write enough that there will be a few
lines after the displayed equation blah blah blah.
\end{document}
```

Notice the equation number.

You can't leave a blank line in a displayed equation:

```
\begin{equation}
 a=F/m.

\end{equation}
```

gives an error message `! Missing $ inserted` in the console window.

Equation numbers and labels

I will explain referencing in class. Then, try replacing in the above

```
...
\begin{equation}
\label{accel}
 a=F/m.
\end{equation}
The period is in the right place. I have placed it inside the
displayed equation \ref{accel}. I want to write enough that there will be a few
lines after the displayed equation blah blah blah.
\end{document}
```

You run this and get in place of \ref{accel} a couple of question marks, as in ??. At this point we want to find out what went wrong (nothing really as you will see). We want to look at error messages in the console output, but it has disappeared form the screen in TeXworks. Force it back by selecting from the top menu *Window > Show Console Output*. You should see, among other stuff:

```
LaTeX Warning:  There were undefined references.
LaTeX Warning:  Label(s) may have changed.  Rerun to get cross-references
right.
```

Several comments are in order:

(i) The clue to what must be done is in "Rerun to get cross-references right.." So hit the typeset button again. Now the typeset document will display the number 1 in place of \ref{accel}. The two "LaTeX warning" lines don' come up in the console anymore.

(ii) You can check more detail of the console output in a file created with .log extension, as in third_project.log

(iii) Two passes necessary: first one creates *auxiliary* file third_project.aux, second one reads it. It keeps track of references/labels. Look at file listing in folder.

Notice no parenthesis around "1." So try instead (\ref{accel}).

Now, to make the point of labeling obvious, insert *before* the current equation, another equation:

```
...
\begin{equation}
\label{eq:Newton}
F=ma
\end{equation}
is the same as
\begin{equation}
\label{accel}
 a=F/m.
\end{equation}
The period is in the right place. I have placed it inside the
displayed equation (\ref{accel}). I want to write enough that there will be a few
lines after the displayed equation blah blah blah. I added equation (\ref{eq:Newton}).
\end{document}
```

4.2 amsmath package

Now insert

```
\usepackage{amsmath}
```

after `documentclass` and replace `equation*` for `equation` in the environment name.

The same can be accomplished using `\nonumber` after the equation in the `equation` environment (with or without the `amsmath` package.

Get a copy of the amsmath user's guide. There are lots of goodies there, some of which we explore next.

4.2.1 Other displayed equations

Long displayed equations

If an expression is too long to fit in one line:

```
\begin{multline}
a+b+c+d+e+f+g+h+i+j+k+l\\
+m+n+o+p+q+r+s+t+u+v+w+x+y+z
\end{multline}
```

Now try this with `equation` replacing `multline`. It ignores the line break. If you look atthe console it says `Overfull \hbox (0.15474pt too wide) detected at line 20`.

There is a no-equation-number of this too:

```
\begin{multline*}
a+b+c+d+e+f+g+h+i+j+k+l\\
+m+n+o+p+q+r+s+t+u+v+w+x+y+z
\end{multline*}
```

Many equations

You may want to display several equations at once. One on top of the next (as many as you want):

```
\begin{equation}
\begin{split}
a+b&=c\\
e+f&=g
\end{split}
\end{equation}
```

The ampersand before the equal sign, `&=`, sayas, "align at equal sign." The ampersand is used in many LaTeX constructions for alignment.

Notice this comes with one equation number for the whole block so we give it only one label.

```
\begin{equation}
\label{manyeqs}
\begin{split}
a+b&=c\\
e+f&=g
\end{split}
\end{equation}
The equations (\ref{manyeqs}) whatever.
```

For separate numbered aligned equations,

```
\begin{align}
\label{manyeqs1}
a+b&=c\\
\label{manyeqs2}
e+f&=g
\end{align}
The equations (\ref{manyeqs1}) and  (\ref{manyeqs2}) whatever.
```

Here is a more complicated situation. You want two equations per line, aligning the two sets of equations:

```
\begin{align}
\label{manyeqs1}
a+b&=c & l & = h + n \\
\label{manyeqs2}
e+f&=g& o+p+q &=r
\end{align}
The equations (\ref{manyeqs1}) and  (\ref{manyeqs2}) whatever.
```

Note the additional `&` to separate the two equations in one line.

There are a few other constructions. See the amsmath user's guide.

4.2.2 Parenthesis in equation number references

Getting tired of writing parenthesis around `\ref{eqlabel}`? With amsmath you can use instead `\eqref{eqlabel}`, as in

```
The equations \eqref{manyeqs1} and \eqref{manyeqs2} whatever.
```

4.3 More math

4.3.1 Subscripts and superscripts

We need to raise to powers and so on:

```
\begin{equation}
E=mc^2
\end{equation}
```

Note that

```
\begin{equation}
a=x^12
\end{equation}
```

Does not give $a = x^{12}$, but $a = x^1 2$. Instead we need grouping:

```
\begin{equation}
a=x^{12}
\end{equation}
```

For subscripts,

```
\begin{equation}
a_1=x_{12}
\end{equation}
```

You can combine them:

```
\begin{equation}
a_1=x_{12}^5
\end{equation}
```

4.3.2 Greek

We use a lot of greek in math. So we need:

```
\begin{equation}
\alpha + \beta = \gamma+\delta
\end{equation}
```

and

```
\begin{equation}
A + B = \Gamma+\Delta
\end{equation}
```

There is no `\Alpha` nor `\Beta` because they are the same symbols as `A` and `B`.

For some symbols there is a variant alternative:

```
\begin{equation}
\epsilon+\varepsilon = \theta+\vartheta=\phi+\varphi
\end{equation}
```

See the complete list of math symbols in the Wikibook.

4.3.3 Simple symbols

Some of the common symbols on the keyboard work straightforwardly (in math mode!):

```
a+n!-b/c=[a*(bc)] < d > e'=|g|
```

4.3.4 Not so simple symbols: operators I

But you may want greater-than-or-equal sign, as in $a \geq b$. Try

```
a \ge b \le c
```

There is an equivalent, sometimes easier to remember,

```
a \geq b \leq c
```

We also have

```
a \equiv b \ll c \gg d \sim g \neq h \propto k\approx z\times w
```

You will memorize many of these as you use them. No point in memorizing them all now. Consult the complete list of math symbols in the Wikibook.

4.3.5 Additional symbols: amssymb

Some symbols that cannot be found in the basic set in can be found elsewhere. A good place to start is the `amssymb` package. Load it:

```
\usepackage{amssymb}
```

and try the following (in math mode!):

```
a \gtrsim b \lesssim c
```

Consult the AMS math list of symbols for more symbols. Really, go there and try some. Here are a few:
`\aleph` $\aleph$ `\beth` $\beth$, `\gimel`, $\gimel$
`\otimes` $\otimes$, `\times` $\times$, `\div` $\div$, `\cap` $\cap$, `\cup` $\cup$
`\infty` ∞, `\forall` $\forall$, `\partial` ∂, `\hbar` $\hbar$

4.3.6 Even more symbols

But if you cannot find it in amssymb, then what. Almost anything you can dream of is available. Consult *and play with* the comprehensive LaTeX symbol list. You will find all sorts of weird stuff, and sometimes several packages that do similar things. For example, the packages `wasysym`, `marvosym` and `mathabx` all supply astronomical symbols. Let's try it:

```
...
\usepackage{wasysym}
...
\mercury, \earth
...
```

These work both in text and in math mode. Try both.

Alert student Zaid Mansuri points to another solution: Detexify is a web site that allows you to scribble a symbol, then attempts at recognizing it and offers several possible LaTeX versions. Try it! Typset this: ♀, ♂, ☿, ⊟, ə.

The message is that other than the common symbols that you will memorize from repeated use, other symbols are easy to find in all these resources.

4.3.7 Fractions

Both in inline and in display modes, but with different results:

```
\frac{a}{b}
```

You can also have one in another

```
\frac{\frac{a}{b}}{\frac{c+d}{d+e}}
```

Grouping is not required, so for simple numerical fractions you can use

```
\frac12=\frac24=\frac{a}b=\frac{a}{b}
```

But the following will give an error message

```
\fracab=\fraca{b}
```

because LaTeX think you want the commands `\fracab`, in the first instance, and `\fraca`, in the second, neither one exist.

Big and small: note the different typesetting of fraction in inline vs display modes: `\(\tfrac12\)` vs `\[\tfrac12\]`.

Sometimes we want a small faction in display mode. Use tiny fraction, `\tfrac`. Compare

```
\[\frac12a+b\]
```

with

```
\[\tfrac12a+b\]
```

4.3.8 Integrals, derivatives and all that

Derivatives are just fractions:

```
\frac{df}{dt}
```

For partial derivatives we need a curly "d"

```
\frac{\partial f}{\partial t}
```

For integrals, sums and products we need new commands:

```
\int f(x) dx, \sum x_n, \prod \omega_k
```

These also display differently in inline/display modes. We can have definite integrals, with limits of integration, and we can display explicitly the range over which we sum:

```
\int_0^1 f(x) dx, \sum_{n=0}^7 x_n,  \prod_1^{10} \omega_k
```

Notice the different placement of the limits in display *vs* inline modes.

4.4 Detour: Extra space and text in math

If we want to separate two expression in display mode,

```
a=g,  \quad c=z, \qquad d=0
```

This is particularly useful with a little text, but this looks bad:

```
a=g\quad then implies \quad g=2
```

which gives $a = g \quad thenimplies \quad g = 2$. We need instead

```
a=g\quad \text{then implies} \quad g=2
```

Fine points of typesetting: extra space

It is usually best to let LaTeX decide how much space to leave between symbols in a mathematical expression. It is designed to make it look best. But sometimes it does not understand the context well enough and gives not so good looking results. For this we have, in *math mode*, a full space `~`, a small space `\;`, a smaller space `\:`, and an even smaller space `\,`:

```
a~a\;a\:a\,a a~~~a   ~    a
```

Note that white spaces are ignored!

You can also subtract a bit of space, using `\!`. You can use it repeatedly:

```
a\!a~a\!\!a
```

You can see that normally you want to stay away from this.

Now we can go back to the example above with integrals and re-write:

```
\int_0^1 \!\!f(x)\, dx,\qquad \sum_{n=0}^7 x_n,\qquad  \prod_1^{10} \omega_k
```

In text mode, the special character `~` leaves a blank space. But it is special: in `this~word` it interprets the whole expression as one word (even if there is a blank space between `this` and `word`). So LaTeX tries to avoid a line break in the middles of the expression. It is useful to avoid starting a line with, for example, a number.

For example: Your text says *blah blah in equation 7 we see blah blah.* But after typesetting you may find ... *blah blah in equation 7 we see blah blah*, that is the line ends with *equation* and the next one starts with *7*. To avoid this you type `blah blah in equation~7 we see blah blah`.

It is good practice to do this as you type, so that you don't have to go back and edit your document every time this occurs.

4.5 Even more math

4.5.1 Trigonometry and such

Bad:

```
cos(\theta)=sin(\tfrac{\pi}2-\theta)
```

Good:

```
\cos(\theta)=\sin(\tfrac{\pi}2-\theta)
```

We also have, among others,

```
\exp(x)=e^x,\quad \ln x=\log x,\quad \sinh(z),\quad \theta=\arcsin P
```

By now you should know where to find a more complete list!

4.5.2 Limits and infinity

```
\frac{df}{dt}=\lim_{\epsilon \to 0}\frac{f(t+\epsilon)-f(t)}{\epsilon}
```

and

```
\lim_{x \to \infty}\frac{1}{x}=0
```

4.5.3 Roots

You can write

```
x^{\frac12}=\sqrt{x}
```

and

```
x^{\frac13}=\sqrt[3]{x}
```

Note how it automatically resizes:

```
\sqrt{\frac{y}{x}}=\sqrt{a+b+c+d}
```

4.5.4 Vector stuff

Newton's second law is actually a vector equation:

```
\vec F = m \vec a
```

Operatons between vectors

```
\vec a\cdot \vec b, \quad \vec a\times\vec b,
```

You can combine things freely, as in

```
\vec a\cdot (\vec b+\vec c_1), \quad \vec a\times\vec b,
```

4.5.5 dot-dot-dot

For a list you need *lower* dots

```
x_1,\ldots,x_n
```

but for a product or a sum you need plain dots

```
x_1+\dots+x_n,\qquad y_1y_2\cdots y_n
```

In text dots (called ellipsis) are `\dots` or `\ldots`. You should not use `...`. Compare (in text mode):

```
from here \ldots to there ... and there
```

By the way, there is no space between `\ldots` and `to` in this example. To add space you can use

```
from here \ldots\ to there ... and there
```

or

```
from here \ldots~to there ... and there
```

More on these later.

4.6 Delimiters

Bad:

```
(\frac12 a+\frac{x+y}{z+d})
```

Better:

```
\left(\frac12 a+\frac{x+y}{z+d}\right)
```

Other delimiters:

```
( a ), [ b ], \{ c \}, | d |, \| e \|, \langle f \rangle
```

The last one is very common to denote averages. They all get resized and don't have to match:

```
\left\langle\frac12 a+\frac{x+y}{z+d}\right]
```

4.6.1 Unmatched resized delimiters

You may want

```
\left. \frac{df}{dt} \right|_{t=0}
```

4.6.2 Multiline equations

If you try

```
\begin{multline}
\left(\frac12 a\\
+\frac{x+y}{z+d}\right)
\end{multline}
```

you get the error message

```
! Missing \right. inserted.
<inserted text>
                \right .
l.34 \end{multline}
```

Instead we need

```
\begin{multline}
\left(\frac12 a \right. \\
\left.+\frac{x+y}{z+d}\right)
\end{multline}
```

This works in any environment that goes over many lines. For example:

```
\begin{align*}
\left(\frac12  a \right. & \\
&\left. +\frac{x+y}{z+d}\right)
\end{align*}
```

Note that the alignment symbol cannot be inside the delimiters. The following gives an error message:

```
\begin{align*}
\left(\frac12 & a \right. \\
&\left. +\frac{x+y}{z+d}\right)
\end{align*}
```

4.6.3 Manual Control

Again, LaTeX generally makes the right decision on how to resize delimiters, but sometimes you want to override it:

```
( \big( \Big( \bigg( \Bigg(
```

Now that you have seen it, where would you use it? Here is one example:

```
(a+f(x)),\quad \left(a+f(x)\right), \quad \big(a+f(x)\big)
```

Chapter 5

Alignment: Matrices & Tables

We have seen the alignment character is &.

5.1 Matrices

Start a new LaTeX document, `fourth_project.tex`:

```
\documentclass[11pt]{article}
\usepackage{amsmath}

\begin{document}
\[
\begin{matrix}
  a & b & c \\
  d & e & f \\
  g & h & i
 \end{matrix}
\]
\end{document}
```

gives a matrix without delimiters. For standard delimiters:

```
\begin{pmatrix}
  a & b & c \\
  d & e & f \\
  g & h & i
 \end{pmatrix}
```

You can have several and do stuff with them, eg

```
\[
2A+3\times\begin{pmatrix}
  a & b & c \\
  d & e & f \\
  g & h & i
 \end{pmatrix}
=0
\]
```

For a determinant

```
\begin{vmatrix}
  a & b & c \\
  d & e & f \\
  g & h & i
 \end{vmatrix}
```

You can use other delimiters by use of the \left and \right commands:

```
\left(\begin{matrix}
  a & b & c \\
  d & e & f \\
  g & h & i
 \end{matrix}\right]
```

5.1.1 Arbitrary size matrices

Use of \cdots (horizontal), \vdots (vertical) and \ddots (diagonal):

```
A =
 \begin{pmatrix}
  a_{11} & a_{12} & \cdots & a_{1n} \\
  a_{21} & a_{22} & \cdots & a_{2n} \\
  \vdots  & \vdots  & \ddots & \vdots  \\
  a_{m1} & a_{m2} & \cdots & a_{mn}
 \end{pmatrix}
```

5.1.2 Extra space

Sometimes LaTeX does not calculate space correctly:

```
M = \begin{bmatrix}
       \frac{5}{6} & \frac{1}{6} & 0           \\
       \frac{5}{6} & 0           & \frac{1}{6} \\
```

```
      0           & \frac{5}{6} & \frac{1}{6}
    \end{bmatrix}
```

To fix this add extra space:

```
M = \begin{bmatrix}
      \frac{5}{6} & \frac{1}{6} & 0           \\[0.3em]
      \frac{5}{6} & 0           & \frac{1}{6} \\[0.3em]
      0           & \frac{5}{6} & \frac{1}{6}
    \end{bmatrix}
```

Here `em` is a distance measure, roughly equal to the width of the letter `m`. So the space we have added is `0.3em`, that is 30% of the width of `m`.

The construction `\\[`*distance*`]` can be used also in line breaks. Type the following in text mode:

```
Here is a short line\\[1in]
and the new line of text continues 1 in below.
```

5.1.3 Array

Sometimes we need finer alignement or finer spacing control or insertion of guide lines in a matrix. For this there is

```
\begin{array}{ccc}
  -a & -b & c \\
  d & e & f \\
  g & h & i
 \end{array}
```

The command takes an argument, in this case `ccc` telling LaTeX to center justify the columns. Try instead

```
\begin{array}{rrr}
  -a & -b & c \\
  d & e & f \\
  g & h & i
 \end{array}
```

for right justification of the columns. You can guess `lll` left justifies. You can mix these:

```
\begin{array}{rlc}
  -a & -b & c \\
  d & e & f \\
  g & h & i
 \end{array}
```

Placing a vertical separator line is easy:

```
\begin{array}{r|rr}
  -a & -b & c \\
  d & e & f \\
  g & h & i
 \end{array}
```

You can place several vertical separator lines. Change the above to {|r|r|r|}, or to {|r||r|||r|}, etc.

For horizontal lines use \hline:

```
\begin{array}{r|rr}
  -a & -b & c \\
\hline
  d & e & f \\
\hline
  g & h & i
 \end{array}
```

The spacing is not quite right. A crutch around this is to add an extra blank line with a line break followed by negative space. We adjust the amount of negative space by trial an error:

```
\begin{array}{r|rr}
  -a & -b & c \\
\hline\\[-1em]
  d & e & f \\
\hline\\[-1em]
  g & h & i
 \end{array}
```

It is possible to do this more elegantly; see the Stack Exchange answer to this issue. Alternatively use the tabls package, which fixes the problem automatically. Note however the output may not be precisely what you want and you may need to set some dimension parameters, but we have not learned to do that yet.

5.2 Tables

We often need tables to display information succinctly. Tese are not matrices, they belong in text.But they look like matrices without delimiters, only with text entries. So the construction is similar.

5.2.1 The Tabular Environment

This is basically the same as `array` but with text entries (and in text mode). So in the last example change the environment, add a few vertical and horizontal lines and take it out of the displayed equation:

```
\documentclass[11pt]{article}
\begin{document}
Here is a table:
\begin{tabular}{r||r|r}
  -a & -b & c \\
\hline\hline
  d & e & f \\
\hline
  g & h & i
\end{tabular}
\end{document}
```

A more interesting table has actual text in it and occupies its own paragraph:

```
\documentclass[11pt]{article}
\begin{document}
Here is a table:

\begin{tabular}{c||l|l}
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
\hline
  11 & Didier & Drogba
\end{tabular}
\end{document}
```

If you want it centered you should use the `center` environment. Try it!

Fixed Width Columns: Text Wrapping

Sometimes the text in one of the entries in the column is long. LaTeX adjusts the table by making the column wider. You may not want this:

```
\begin{center}
\begin{tabular}{c||l|l}
  Jersey & First Name & Last Name \\
\hline\hline
```

```
  10 & Cristiano & Ronaldo \\
\hline
  11 & Didier & Drogba\\
  \hline
10 & Edson & Arantes do Nascimento (Pele)
\end{tabular}
\end{center}
```

Replacing the alignment character by p{*dimension*} gives a left-right justified column of fixed width of the given dimension:

```
\begin{center}
\begin{tabular}{c||l|p{1in}}
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
\hline
  11 & Didier & Drogba\\
  \hline
10 & Edson & Arantes do Nascimento (Pele)
\end{tabular}
\end{center}
```

To add a line break within a cell you use `\newline` (obviously `\\` won't work):

```
\begin{center}
\begin{tabular}{c||l|p{1in}}
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
\hline
  11 & Didier & Drogba\\
  \hline
10 & Edson & Arantes\newline do Nascimento  (Pele)
\end{tabular}
\end{center}
```

Partial Horizontal Line

Sometimes you want lines across some columns but not all:

```
\begin{center}
\begin{tabular}{c||l|p{1in}}
```

```
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
\cline{2-3}
 & Edson & Arantes\newline do Nascimento  (Pele)\\
\hline
  11 & Didier & Drogba
\end{tabular}
\end{center}
```

The command `\cline` takes as argument the rangle of columns over which you want the horizontal line. It can be only one column, in which case the start and end of the range of columns is the same. For example, writing `\cline{2-2}` instead of `\cline{2-3}` would have given a line across column 2 only, rather than columns 2 and 3 in the example above.

Breather: Good coding practice

Without change to the output, you can code the above as

```
\begin{center}\begin{tabular}{c||l|p{1in}}Jersey&First Name&Last Name\\
\hline\hline   10&Cristiano&Ronaldo\\\cline{2-3}&Edson&
Arantes\newline do Nascimento (Pele)\\\hline 11&Didier&Drogba
\end{tabular}\end{center}
```

You decide which is clearer. Think possible errors and debugging!

Partial vertical lines

How do we do the opposite, that is, a vertical line that does not span all rows? Use `\multicolumn` with three arguments: the first is the number of columns, the second is alignment, `l, c, r` with possible vertical bars, and the third one is the text:

```
\begin{center}
\begin{tabular}{|c||l|p{1in}|}
\hline
\multicolumn{3}{|c|}{Dream Team}\\
\hline
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
\cline{2-3}
```

```
 & Edson & Arantes\newline do Nascimento  (Pele)\\
\hline
  11 & Didier & Drogba\\
\hline
\end{tabular}
\end{center}
```

Spacing between rows, again

As with `array` we have an issue of separation between horizontal lines and text. We can use our previous trick of adding a line plus negative space. In the previous use `\\[-1em]` after `\hline` or `\cline` as appropriate.

Environments within cells

Within each cell in a table you can use any environment. This only works if the column containing the environment is justified by `p{`*dimension*`}`. This is useful, for example, in typing tables with formulae:

```
\begin{center}
\begin{tabular}{c|p{2in}}
  Newton & \begin{equation*} \vec F=m\vec a\end{equation*} \\
\hline
  Einstein & \begin{equation} E=mc^2\end{equation}
\end{tabular}
\end{center}
```

This is not pretty, but works. For simple equations it is easier to use the inline-text equation. The following looks better:

```
\begin{center}
\begin{tabular}{c|c}
  Newton & $ \vec F=m\vec a$ \\
\hline\\[-1em]
  Einstein &$E=mc^2$
\end{tabular}
\end{center}
```

Alignment at decimal point, and other separators

It does not look pretty:

```
\begin{center}
```

```
\begin{tabular}{c}
2.30\\
100.23\\
0.12345
\end{tabular}
\end{center}
```

Instead use:

```
\begin{center}
\begin{tabular}{r@{.}l}
2&30\\
100&23\\
0&12345
\end{tabular}
\end{center}
```

Here is what's going on. We used two columns, one right justified and the next left justified. The separator is not a vertical line but whatever goes in `@{`*separator symbol*`}`, in this case the separator symbol being a period.

5.3 Advanced Tables

Several packages, like `array`, `tabularx` and `tabulary`, give you many more options for constructing complicated tables. You know where to find more details. Here is just one example, using the switch `\rowcolors` provided by the `xcolor` package with option `table`:

```
\documentclass[11pt]{article}
\usepackage[table]{xcolor}
...
\begin{document}
...

{ \rowcolors{1}{yellow}{pink}
\begin{tabular}{|c||l|p{1in}|}
\hline
\multicolumn{3}{|c|}{Dream Team}\\
\hline
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
```

```
\cline{2-3}
 & Edson & Arantes\newline do Nascimento  (Pele)\\
\hline
  11 & Didier & Drogba\\
\hline
\end{tabular} }
...
```

Chapter 6

Floats and Graphics

Think of a *float* as a frame into which you can put something, like a table or a picture (an image). It floats in your document, in the sense that it finds a place to land in your document. LaTeX decides where, although you can nudge it into placing it where you want. Again, LaTeX is designed to make a good decision for you, so give it a chance.

6.1 The table environment

Use this in documents to (i) automatically place the table, (ii) automatic table number referencing, (iii) include a caption. Example, using something from `fourth_project.tex`:

```
Here is some text. In table \ref{mytable} we show our Dream Team
\begin{table}[b]
\begin{center}
\begin{tabular}{|c||l|p{1in}|}
\hline
\multicolumn{3}{|c|}{Dream Team}\\
\hline
  Jersey & First Name & Last Name \\
\hline\hline
  10 & Cristiano & Ronaldo \\
\cline{2-3}
 & Edson & Arantes\newline do Nascimento  (Pele)\\
\hline
  11 & Didier & Drogba\\
\hline
\end{tabular}
```

```
\end{center}
\caption{
This table shows blah blah \label{mytable}
}
\end{table}
```

The environment `table` takes one *positioning* argument, `b` in this example. It stands for, *place this table at the bottom of a page, as close as possible to where it was inserted in the code*. You can also insert it `h` for *here*, and `t` for *at the top of the page*. The top of the page is the most commonly used.

The command `\caption` takes an argument text that constitues the caption to the table. You can insert a label and refer to it just as we did with equations. The counter for table numbers is independent of the counter for equation numbers.

6.1.1 List of tables

Sometimes in a very long document you want a list of tables showing where to find each table. Just add `\listoftables`. Add this now to the previous example, just before your `\end{document}`:

```
\newpage
\listoftables
```

6.2 Inserting Graphics

The command for inserting a graphic (a picture) into your document is `\includegraphics[`*attributes*`]{`*filename*`}`. For an example we first need a graphics file. Let's download one from the web. I go to *Google images* and search for "UCSD logo." I save an image named `gl-5-triton.png`. Notice the extension is `.png`. More on this later. Now I insert into my file a line

```
\includegraphics{gl-5-triton.png}
```

and get an error message. The reason is that LaTeX is happy with a particular format for images, namely `.eps` (encapsulated postsript) images. We need a package that helps LaTeX with other formats. One common package for this is `graphicx`. Use this (put `\usepackage{graphicx}` in the preamble) and typeset your document with the figure.

Next try

```
\includegraphics[width=2in]{gl-5-triton.png}
```

The attribute `width=`*dimen* sets the width of the image to the desired dimension. This and `height=`*dimen* are the most common, but there are a few others, like `scale=`*factor* which amplifies the image by the factor (reduces it if factor is smaller than 1) and `trim=l b r t` which is used for cropping by lengths l, b, r and t, from the left, bottom right and top, respectively. For `trim` to work you have to add the attribute `clip` (or rather set `clip=true` but adding just `clip` works):

```
\includegraphics[trim= 8cm 0cm 0cm 0cm,clip ]{gl-5-triton.png}
```

See the complete list of attributes in the WikiBook.

Try this:

```
\includegraphics[width=2in, height=4in]{gl-5-triton}
```

The extension is not necessary: LaTeX knows to look for `.png` files as graphic files. In fact, if you are typesetting with *pdfLaTeX*, and we are, and are using the *graphicx* package, and we are, then LaTeX looks for files with extensions `.jpg`, `.png` and `pdf`, and assumes that if you did not write an extension that it will find the full file with extension in the current directory (and looks for all three extensions).

6.2.1 Graphics folder

It is convenient to have a folder (ie, a directory or subdirectory) containing your graphics files, particularly when there are many of them. Often we use a folder `images` in the current folder. One can then insert the graphic file by specifying the path to the file from the local folder, eg, if we put our Triton image in the folder `images` located in the folder with our `fourth_project.tex` then we include it with

```
\includegraphics{images/gl-5-triton}
```

or with

```
\includegraphics{./images/gl-5-triton}
```

Create a folder within the current one called `images` (or anything else), move the graphics file there and try the above (with the obvious modification if you named the folder differently). There are subtleties arising in including blank spaces in the name of the folder or the document, so better don't!

Note: if your folder name has blank spaces, as in `my pictures`, you will have trouble using this.

6.3 Figure Environment

The `figure` environment is analogous to the table environment. It is also a float. Without further ado:

```
\begin{document}
Some text here.

\begin{figure}[t]
\begin{center}
\includegraphics[width=2in]{images/gl-5-triton}
\end{center}
\caption{\label{tritonLogo} This is a UCSD logo that displays the
  mascot, a Triton.}
\end{figure}

In figure \ref{tritonLogo} one finds a Triton. This is the symbol of \dots
\end{document}
```

6.3.1 List of figures

Just as with tables:

```
\listoffigures
```

You can put this anywhere in your document, but it is often found eiher at the very beginning together with the table of contents, or at the end.

6.4 More...

We will not do this in the course but you may want to play with putting boxes around images, text wrapping, inserting text on top of an image, including full pdf pages, and more. Consult the Wikibook. Play with it! There is also a lot of information there about converting among different formats abd editing graphics.

Chapter 7

Lists and other useful environments

7.1 Itemized lists

Here is how to make one:

```
\begin{itemize}
\item The first entry here
\item Then the second
\item etc
\end{itemize}
```

To change the bullet into something else you have two options. First, line by line:

```
\begin{itemize}
\item[-] The first entry here
\item[*] Then the second
\item[$>$] etc
\end{itemize}
```

If you want to change the item symbol for the whole document insert the following before the first `{\begin{itemize}` (can go in the preamble):

```
\renewcommand{\labelitemi}{\textgreater}
```

You can change this as many times as you please.

7.1.1 Nested itemized lists

You can nest lists:

```
\begin{itemize}
\item The first entry here
\item Then the second, which leads to
\begin{itemize}
\item The first sub-entry here
\item Then the second sub-entry
\item etc
\end{itemize}
\item Return to original list, etc
\end{itemize}
```

If you want to change the symbols of these lists do as before, inserting before `\begin{document}` your choices:

```
\renewcommand{\labelitemi}{\textgreater}
\renewcommand{\labelitemii}{$\star$}
```

7.2 Numbered lists

You can use

```
\begin{enumerate}
\item The first entry here
\item Then the second
\item etc
\end{enumerate}
```

You can also nest these:

```
\begin{enumerate}
\item The first entry here
\item Then the second, which leads to
\begin{enumerate}
\item The first sub-entry here
\item Then the second sub-entry
\item etc
\end{enumerate}
\item Return to original list, etc
\end{enumerate}
```

7.2.1 Controlling the numbered list

For this we use the `enumerate` package.

```
\usepackage{enumerate}
...
\begn{document}
...
\begin{enumerate}[(i)]
\item The first entry here
\item Then the second
\item etc
\end{enumerate}
```

The option we used is one of `A, a I, i` and `1`. The parenthesis, and anything else, is treated as text. If you want to use any of `A, a I, i` and `1` in the literal text you should enclose it in curly brackets, as in {`I`}. So for example

```
\usepackage{enumerate}
...
\begn{document}
...
\begin{enumerate}[Exerc{i}se 1)]
\item The first entry here
\item Then the second
\item etc
\end{enumerate}
```

works, but try leaving out the curly brackets (really, try!).

This is all that the package `enumerate` does. Just a simple way of handling the labels. The more sophisticated package `enumitem` gives you much more control. For example, you can modify the layout of the list. But it is more difficult to use. For example, the same output as above is obtained by using

```
\usepackage{enumitem}
...
\begn{document}
...
\begin{enumerate}[label=Exercise \arabic*)]
\item The first entry here
\item Then the second
\item etc
\end{enumerate}
```

So this is more complicated. But now we can do more. For example,

```
\usepackage{enumitem}
...
\begn{document}
...
\begin{enumerate}[label=Exercise \arabic*), itemsep=1in]
\item The first entry here
\item Then the second
\item etc
\end{enumerate}
```

There are many such controls, `topsep`, `leftmargin`, `rightmargin` and `labelwidth` among others. You should play with these. Get the full power of this package from the enumitem package documentation. It allows for referencing entries in the item lists, making inline lists and more.

7.3 Description Lists

This is sometimes useful:

```
\begin{description}
\item[First] The first entry here
\item[Second] Then the second
\item[Last] Then the last
\end{description}
```

The argument of `item` is optional, but why would you not use it, and the output would look peculiar.

Chapter 8

Bibliography

A painful aspect of composing a technical or academic document is the bibliography. We would like to have a simple way of citing papers or books, and to list them accordingly at the back of our document. Sometimes we are asked to have the listing of references alphabetized, sometimes the publisher (or prof) wants us to list them in the order in which they are used/cited in the document. The style of the references may change from publisher to publisher too. What a pain!

Fortunately LaTeX can take care of all this for you.

8.1 The easy way: `thebibliography` environment

So we need to refer to, say,

On the Relation between the Expansion and the Mean Density of the Universe, A. Einstein, W. de Sitter, Proc.Nat.Acad.Sci. 18 (1932) 213-214

and

On Gravitational waves, Albert Einstein, N. Rosen, J.Franklin Inst. 223 (1937) 43-54

And the text should read something like

Einstein discussed the relation between the expansion and the mean density of the universe[1], and then he went on to say something about gravitational waves[2], whatever they are

To do this in LaTeX we first write text into our document:

```
Einstein discussed the relation between the expansion and the mean
density of the universe\cite{albert1}, and then he went on to say something
about gravitational waves\cite{einsteinRosen}, whatever they are.
```

Note that the labels are completely arbitrary text tokens. You choose! Now add before the \end{document} the following references:

```
\begin{thebibliography}{99}

\bibitem{albert1}
On the Relation between the Expansion and the Mean Density of the Universe,
A. Einstein, W. de Sitter, Proc.Nat.Acad.Sci. 18 (1932) 213-214

\bibitem{einsteinRosen}
On Gravitational waves,
Albert Einstein, N. Rosen, J.Franklin Inst. 223 (1937) 43-54
\end{thebibliography}
```

The blank lines between bibitems are ignored by LaTeX: you can add more or have none at all. Blank lines within a bibitem produce unwanted line breaks.

As with other referencing in LaTeX the first time you run this you get question marks in place of citing references, as in *Einstein discussed ... of the universe[?], and then he went on to say something about gravitational waves[?], whatever they are.* The Console output contains,

```
LaTeX Warning: Citation 'albert1' on page 1 undefined on input line 15.

LaTeX Warning: Citation 'einsteinRosen' on page 1 undefined on input line 16.
```

and then lower down, near the end,

```
LaTeX Warning: There were undefined references.

LaTeX Warning: Label(s) may have changed. Rerun to get cross-references right.
```

The last line tell you how to fix the problem. Just run LaTeX again. But the Warning messages are more generally useful for debugging problems, as we will discuss in lecture. So typeset again. Now the question marks are replaced by `[1]` and `[2]`, and the Console output gives no Warning messages.

Try this: List the two references in the bibliography in the reverse order. What happens? Make sure you look at the typeset output and the Console output after each time you run LaTeX now.

The mandatory argument of the `thebibliography` environment indicates the width of the field used for enumerating. It counts the digits. So if the argument is 1 or 7 or 9 it means *one* digit which is useful for a list with no more than 9 entries. Similarly if the argument is 34 or 87 or 99 it tells LaTeX you will use two digits for the list. That's 99 references maximum. So it is common to use as argument 9, or 99, or 999. It is never a bad idea to overestimate the number of references in your work.

8.2 The hard way: BibTeX

BibTeX is more flexible, but more difficult to use. It has the great advantage that bibliography files are stored and can be reused in many documents. They are databases. Only the references you need are used.

We need to

1. Make one (or more) plain text bibliography files (databases), extension `.bib`
2. Include `\cite{bibitemlabel}` in text as before
3. Include

   ```
   \bibliographystyle{plain}
   \bibliography{samplebib1}
   ```

   before `\end{document}`
4. Typeset with *pdfLaTeX*. Then run *BibTeX*. Then typeset twice with *pdfLaTeX*.

To run *BibTeX* choose *BibTeX* from the drop-down menu in *TeXworks* (the one that normally says *pdfLaTeX*).

This is a lot to swallow at once. Let's go one step at a time

8.2.1 The `bib` file(s)

The bibliography files are plain text files. You can create them with any plain text editor. We can use TeXworks for this. From the File menu choose New. Enter the bibliography information (bellow), and when you are ready *Save as ...*. In the *Save* dialogue box make sure

1. to save in the same folder as you LaTeX document
2. to choose *BibTeX databases (*.bib)* from the *Files of Type:* drop-down menu.

For our example above write into the new file

```
@article{albert1,
  author = "Einstein, A.,  de Sitter, W.",
  year = "1932",
  title = "On the Relation between the
   Expansion and the Mean Density of the Universe",
  journal = "Proc.Nat.Acad.Sci.",
  volume = "18",
  pages = "213--214"
}

@article{einsteinRosen,
  author = "Einstein, Albert,  Rosen, N.",
  year = "1937",
  title = "On Gravitational waves",
  journal = "J.Franklin Inst.",
  volume = "223",
  pages = "43--54"
}
```

I called my database `samplebib1.bib`. I will make other bib files below and will give them different names.

Building the database made easy

In your browser go to *Google Scholar*, and search for *einstein rosen on gravitational waves*. The firs hit will be the paper in question. Under it click *Cite* and rom the dialog box clik *BibTeX*. You can then copy and paste into a database file, which in my computer I will call `samplebib2.bib`:

```
@article{einstein1937gravitational,
  title={On gravitational waves},
  author={Einstein, Albert and Rosen, Nathan},
  journal={Journal of the Franklin Institute},
  volume={223},
  number={1},
  pages={43--54},
  year={1937},
  publisher={Elsevier}
}
```

This is almost what we want. The label for the reference, `einstein1937gravitational` is not what I use in my LaTeX file to cite this reference. So I edit the file to change the first line to read `@article{einsteinRosen,`

Next I look for *einstein de sitter On the Relation between the Expansion and the Mean Density of the Universe* and paste into my file

```
@article{einstein1932relation,
  title={On the Relation between the Expansion and the Mean Density of the Universe},
  author={Einstein, Albert and De Sitter, Willem},
  journal={Proceedings of the National Academy of Sciences of the United States of Ame
  volume={18},
  number={3},
  pages={213},
  year={1932},
  publisher={National Academy of Sciences}
}
```

and edit the first line to read `@article{albert1,`

Many journals give you the full reference in BibTeX format. Example:

1. Go to www.sciencemag.org
2. Navigate: *Science Journals* > *Science* > *Science Home* > *Table of Contents* > *Reports*
3. For any report navigate *Abstract* > *Download Citation* (found under Article Tools on left margin) > *BibTeX*
4. Copy and paste to your database

You are out of luck with Nature. But fine with many journals, including American Physical and Chemical Societies journals, IEEE journals, and more.

BibTeX templates: fields and entry types

Sometimes you have to write the entry into the database form scratch. It is useful to have a template which contains all possible *fields*:

```
@article{Xarticle,
    author    = "",
    title     = "",
    journal   = "",
    volume    = "",
    number    = "",
    pages     = "",
    year      = "XXXX",
    month     = "",
    note      = "",
}
```

Some comments: Author names are `Last, First` and multiple names are separated by `and`. Year has to be four digits. Fields that are not used should not appear (do not use % to comment unwanted fields).

`article` is an *entry type*. There are many others. One that you may use frequently is

```
@book{Xbook,
    author    = "",
    title     = "",
    publisher = "",
    volume    = "",
    number    = "",
    series    = "",
    address   = "",
    edition   = "",
    year      = "XXXX",
    month     = "",
    note      = "",
}
```

Here is an example copied from an entry in Google Scholar:

```
@book{weinberg1993first,
  title={The first three minutes: a modern view of the origin of the universe},
  author={Weinberg, Steven},
  year={1993},
  publisher={Basic Books}
}
```

8.2.2 BibTeX styles

Edit your LaTeX document.

1. Remove `thebibliography`
2. Include in its stead:

   ```
   \bibliographystyle{plain}
   \bibliography{samplebib1}
   ```

   before `\end{document}`
3. Typeset with *pdfLaTeX*. Then run *BibTeX*. Then typeset twice with *pdfLaTeX*.

To run *BibTeX* choose *BibTeX* from the drop-down menu in *TeXworks* (the one that normally says *pdfLaTeX*).

The argument `plain` in `\bibliographystyle` is a BibTeX *style*.
Now change that for `alpha` and typeset again (four steps!). And again with `abbrv`. Then try it with `samplebib2`. Notice that the names are abbreviated, even if the full first name was given in the database. A list of styles with samples can be found in https://www.sharelatex.com/learn/Bibtex_bibliography_styles

Finally, change the bibilography:

```
\bibliography{samplebib1,samplebib2}
```

This is how you specify more than one databases. In this case there are repeated entries in the databases. How does BibTeX deal with this? When you run BibTeX you will see in the Console Output a number of messages:

```
Repeated entry---line 1 of file samplebib2.bib
 : @article{albert1
 :                  ,
I'm skipping whatever remains of this entry
Repeated entry---line 12 of file samplebib2.bib
 : @article{einsteinRosen
 :                        ,
I'm skipping whatever remains of this entry
(There were 2 error messages)
```

You should clean up your databases to avoid this! But BibTeX is smart enough that you can still typeset the document. It will use the first instance of the repeated reference found. Try it!