Towards the Ability to Make Super Duper LaTeX Documents

Christopher James Forman

Department Of Chemistry

University of Cambridge

This dissertation is submitted for the degree of *Doctor of Philosophy*

November 7, 2010

Declaration

Acknowledgments

Family

Mum. Dad. Pet Dog.

Scientific Collaborations

Dr A.N.Other at uni of blah for samples and instrument time etc. My Amazing Lab Chums.

Scientific Support

My amazing lab chums.

General Assistance

Cleaner, My amazing lab chums.

Financial Support

Bank, Tax payer. My amazing lab chums.

Moral Support

My Amazing Lab chums.

Academic Inspiration

Netwon, My super. My Amazing Lab Chums.

Everyone else

My Amazing Lab Chums.

Summary

One possibility for controlling the high density organisation of enzymes for technological purposes, which could increase human use of sunlight and absorb atmospheric CO_2, would be to decorate amyloid fibres with functional enzymes. Therefore several multi-domain fusion proteins were made which consisted of permutations of the SH3 fibre forming domain and cytochrome b_{562}—an electron transfer domain whose native state is stabilised when it binds the electron transfer cofactor ferrous protoporphyrin IX (also known as haem). It was anticipated that these fusion proteins would form fibres displaying the electron transfer enzymes and would be capable of transporting charge along the complete length of the fibre. To understand the organisation of the proteins in a fibre, at a molecular level, a variety of experimental and theoretical approaches were required.

Most permutations of the fusion proteins successfully formed amyloid fibres, whose ribbon-like morphology varied on a transition between twisted ribbons and spiral ribbons, with notable exceptions. It is known that the helical fibres are formed from helical filaments consisting of beta strands that are perpendicular to the fibre axis. A simple model was created based on maximising the projection of adjacent beta strands, modelled as short rigid rods, on a helical space curve. Such maximisation occurs when the beta strands are perpendicular to the helical space curve and parallel to an arbitrary global reference plane, a condition satisfied by the normal vector of the Frenet frame for a helix. Filaments formed in this way can be coaxially aligned at varying displacements to recreate both twisted and spiral ribbon morphologies.

The variation in morphology, driven by the variation in sequence of the fusion proteins, was quantified by low resolution TEM and could be explained in the context of the model by distortion of the position of the beta strands due to excess material which was not involved in the core of the fibre. Low resolution AFM was used to confirm these trends and discovered that surface adsorption flattens the fibres against the substrate such that filament crossover points form features which enumerate the number of filaments in the fibre. Adsorption can also disintegrate the fibres into segments of characteristic length due to strain on the filaments at the crossover points.

High resolution AFM under liquid found that the surface roughness of the fibre depended on the species under consideration, the quantity of haem bound by the cytochromes and also confirmed the dimensions and number of filaments within each fibre species.

A key observation from preliminary work—that only half of the fibre-borne cytochromes can bind haem—was confirmed and extended to all the fibre forming proteins studied. The haem binding improves the imaging properties of the fibre and changes the morphology in a systematic way, thus yielding small molecule control over the fibre morphology which, in principle, could be used to bring the displayed cytochromes into adequate proximity for electron transfer to occur.

The improved imaging arising from haem binding allowed the high resolution AFM tip to penetrate deeper into the fibre. The existence of an inner filament linked to surface moeities was confirmed and the best information possible was used to create a model of the fibre.

The fibres were probed with STM and found to be insulating regardless of the presence of haem. Some conducting features were observed but could not be identified as fibres unequivocally. The formation of an ionic aqueous film combined with the rapid cycling of fusion proteins between the fibres and the solution inhibited the STM analysis of the fibres.

Abbreviations

ADP	Adenosine Diphosphate
(nc/c)AFM	(non-contact/conducting) Atomic Force Microscopy
ATP	Adenosine Triphosphate
BSS	Cytochrome b_{562}-SH3-SH3 fusion protein
BSSB	Cytochrome b_{562}-SH3-SH3-cytochrome b_{562} fusion protein
CD	Circular Dichroism
E. coli	*Escherichia coli*
DNA	Deoxyribonucelic Acid
EPR	Electron Paramagnetic Resonance
ET	Electron Transfer
FTIR	Fourier Transform Infrared Spectroscopy
GST	Glutathione-s-transferase
HMC	High Molecular weight Cytochrome
I-V	Current-Voltage
LD	Linear Dichroism
MD	Molecular Dynamics
(ss)NMR	(solid state) Nuclear Magnetic Resonance
PBS	Phosphate Buffered Saline
(Fe)PPIX	(Ferrous) Proto Porphyrin IX
RNA	Ribonucleic Acid
SEM	Scanning Electron Microscopy
SPIP	Scanning Probe Image Processing

SPM	Scanning Probe Microscopy
SS	SH3-SH3 fusion protein
SSS	SH3-SH3-SH3 fusion protein
SSB	SH3-SH3-Cytochrome b_{562} fusion protein
SSSB	SH3-SH3-SH3-Cytochrome b_{562} fusion protein
STEM	Scanning Tranmission Electron Microscopy
STM	Scanning Tunnelling Microscopy
STS	Scanning Tunnelling Spectroscopy
(cryo)TEM	(cryogenic) Transmission Electron Microscopy
UHV	Ultra High Vacuum
UV-Vis	Ultra Violet to Visible Light Spectroscopy
XSB	General term for the SB fusion proteins
YFP	Yellow Fluorescent Protein

Contents

List of Tables

List of Figures

Chapter 1

Overview

1.1 LaTeX: The answer to everything.

LaTeX is typsetting program that takes an input file of marked up text and then, with a user defined set of rules, typesets a beautifully crafted output file that can be easily printed (e.g. a pdf).

1.1.1 Advantages

The advantages of this method are manifold:

- Use any text editor to view the source document.
- More time can be spent working on the content and not worrying about how text or figures interact in the layout until later on.
- LaTeX uses consistent rules throughout a document
- LaTeX sorts out basic typesetting automatically
- Changes can be introduced globally with very little effort
- Document structure is explicit
- Documents can be professionally typeset and look great
- You are forced to structure your documents correctly.
- Mathematical equations, like $E = mc^2$ or $\imath\hbar\frac{\partial}{\partial t}\Phi(x,t) = \hat{H}\Phi(x,t)$ can be produced almost as fast as typing (if you know the commands!).

1.1.2 Disadvantages

The disadvanatges of this method are also manifold:

- You don't see the output as you go.
- Steep learning curve.
- Documents are harder to edit by a second author (unless they are adept at LaTeX too). This can be mitigated using version control, which makes group authoring processes superior to word.
- The program never works quite the way you want it to and learning how to influence it can be problematic and subtle.
- Although intended to save work the principle of 'conservation of work' means that you simply transform problems associated with WYSISYG approaches to problems associated with WYSIWYM approaches!
- You can go blind trying to determine the difference between wiggly and smooth brackets if your editors font isn't large enough.

1.1.3 On balance?

If it's so rubbish, why use LaTeX?

- Large documents are much more easily handled.
- In general it is quicker to debug a LaTeX document than typeset an entire thesis manually.
- Technical information such as tables, equations and figures are integrated much more smoothly than with word.

1.2 How does it work?

The raw text is interspersed with commands, preceded by a \, which tell LaTeX what to do with the text. For example you can **make it bold**, *italic*, or underlined with the commands \textbf{}, \emph{} or \underlined{}. More on commands in a moment...

Chapter 2

My First Page

2.1 The Very Beginning

The very simplest LaTeX document might look like this:

```
\documentclass[a4paper,12pt]{article}
\begin{document}
Hello World.
\end{document}
```

Hello World.

2.1.1 Break Down

What's all the gobbledegook around my simple message???

In LaTeX we intersperse text and commands. Commands are preceded by a \. For example the first line in a LaTeX document ***must*** be:

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

Where the word "class" may be substituted for one of many things such as: article, proc, minimal, report, book, letter, memoir, slides, beamer.

Similarly, and completely generally in LaTeX speak, the square brackets denote the existence of optional parameters. Each individual command can take its own paramters and for the "\documentclass" command there are options for controlling font size, font family, landscape, oneside, twosided, page size and so on. These options will persist throughout the entire document. For example the document class command for this document, (which is likely to be like the one you would use for a thesis), would be:

```
\documentclass[12pt, oneside, a4paper]{book}
```

Other options include:

- 10pt, 11pt, 12pt (default is 10pt).
- letterpaper, legalpaper, a4paper, executivepaper, a5paper, b5paper

and so on.

2.2 Can I start typing please?

After we have set up the document we can start actual work on our document. So we must tell LaTeX that what follows is to be interpreted as a document. The command \begin can take many different parameters and is a command to enter what is known, in LaTeX speak, as 'an environment'. Thus the commands:

```
\begin{document}

\end{document}
```

tells LaTeX to enter and leave the document environment, and thus constitute the outer limits of our document file. Other environments include the equation environment, the itemize environment, the figure environment and so on. These will be encountered in due course.

Every part of the LaTeX file is therefore within an environment of specific type and the content within each environment consists of commands or text.

That's basically it.

So let's get on with it shall we...

2.3 The ground rules

Hold on, hold on, hold on my son.

First the lessons.

Then the fun!

Dr Seuss.

2.3.1 Spaces

Whitespace characters, such as blank or tab, are treated uniformly as space by LaTeX. Several consecutive whitespace characters are treated as one single space. Whitespace at the start of a line is generally ignored, and a single line break is treated as whitespace. An empty line between two lines of text defines the end of a paragraph. Several empty lines are treated the same as one empty line. The text below is an example.

```
It does not matter whether you
enter one or several             spaces
after a word.




An empty line starts a new
paragraph.
```

It does not matter whether you enter one or several spaces after a word.

An empty line starts a new paragraph.

2.3.2 Special Characters

The symbols

$ % ˆ & _ { } ˜ \

are reserved characters that either have a special meaning under LaTeX or are unavailable in all the fonts. If you enter them directly in your text, they will normally not print, but rather make LaTeX do things you did not intend.

To overide the special meanings of these symbols and allow them to produced within your text you may use the following sequences:

```
\# \$ \% \textasciicircum{} \& \_ \{ \} \~{} \textbackslash
```

Other symbols and many more can be printed with special commands in mathematical formulae or as accents.

The backslash character ’\’ cannot be entered by adding another backslash in front of it (\\); because this sequence means ”linebreak”.

The command \˜{} produces a tilde which is placed over the next letter. For example \˜{n} gives ñ. To produce just the character ˜, use \˜{} which places a ˜ over an empty box.

Similarly, the command \ˆ produces a hat over the next character, for example \ˆ{o} produces ô. If you need in text to display the ˆ symbol you have to use ‘\textasciicircum{}’.

2.4 Preamble

After the \documentclass command we add ”preamble”. In this section which we load special features that we will use throughout our document. These are contained in units called ”packages” which we can tell our LaTeX compiler to download by using the command ”\include{packageName}”. There are many repositories of such packages on the web. Your compiler generally knows where to look and most standard packages are included with any install. For example, we can grow our simple document like this:

```
\documentclass[a4paper,12pt]{article}
\usepackage[version=3]{mhchem}
\begin{document}
Hello World!

We all need \ce{H2O}.

I'm less fussed about \ce{^{235}_{92}U+}.

\end{document}
```

Hello World!
We all need H_2O.
I'm less fussed about $^{235}_{92}U^{+}$.

Here we have loaded a package called: "'mhchem"' which took the option "[version=3]". This is a package for drawing chemical equations easily and it has it's own instruction manual which you can follow easily. It is included in the bundle of files for this course.

Other things we can do in the premable within LaTeX are to redefine existing commands or create our own personal commands. These can be stored in a file called the 'style file' which we can load at the beginning of our document, in place of our documentclass. More about this later on...

2.4.1 Comments

It is often useful to comment your LaTeX documents. You can leave yourself amusing, sarcastic messages that won't get printed out in the final document.

To get a comment use the % command, which tells LaTeX to ignore the rest of the line, the line break and all the white space at the beginning of the next line, for example, we may add to our continually evolving document...

```
\documentclass[a4paper,12pt]{article}
\usepackage[version=3]{mhchem}
\begin{document}
Hello World!

%All humans need water and I would like
%to include this concept in my arguments.
We all need \ce{H2O}.

%Uranium 235 is toxic, which is why I don't want to consume it...
I'm less fussed about \ce{^{235}_{92}U+}.

\end{document}
```

Hello World!
We all need H_2O.
I'm less fussed about $^{235}_{92}U^+$.

2.5 Compiling

Once the document is finished you can compile it. Your compiler will depend on the platform that you use. In the PWF we are using winEDT to edit the documents (a LaTeX front end) and texlive2008 which is the compiler itself. I use MiKTex and the front end texCenter. There is no need to use a front end, you can use a simple text editor and the command line.

the recommended distributions are:

- MiKTeX or TeX Live for Windows
- TeX Live for Unix/Linux
- MacTeX or TeX Live for Mac OS.

Once you have installed your software and got it working, written your source code and tried to compile it then will nearly always be something wrong with your file. These erros will be highlighted in the console window or error output box of your front end. Some front ends dump the running commentary which LaTeX produces into a text file for easy reading afterwards.

In a compile attempt (successful or otherwise) LaTeX may produce the following files:

projectname.aux
projectname.bbl
projectname.lof
projectname.lot
projectname.txt
projectname.toc
projectname.dvi

These are interim files (toc= table of contents, bbl = bibliography etc). To be honest I have no idea what half of these things contain. You only need the .tex file and a compiler with the right packages installed to produce them again.

Some version of LaTeX only produce DVI files and you then need to convert the dvi file to a PDF or download a dvi viewer. You can also convert DVI files to PS files and then view them. The good thing about front ends is that you can set them up to produce PDFs directly.

OK. So we have now produced a silly document but we understand it. Now lets get on with producing our thesis template!

Chapter 3

Lists

Lists are great. The command \begin{} can be used to enter a list environment. For example:

```
\begin{itemize}
\item cat
\item dog
\item horse
\end{itemize}
```

Produces:

- cat
- dog
- horse

We can also replace the bullet points with numbers using the enumerate keyword.

```
\begin{enumerate}
\item cat
\item dog
\item horse
\end{enumerate}
```

1. cat
2. dog
3. horse

and we can use the description keyword which does this:

```
\begin{description}
\item[Cat] a lovely furry creature with a cute nose and whiskers.
\item[Dog] Another furry creature that smells rather well;
           its olfactory power stems from its nasal dampness.
\item [Horse] A large stinky creature with sideways facing eyes.
\end{description}
```

Cat a lovely furry creature with a cute nose and whiskers.

Dog Another furry creature that smells rather well; its olfactory power is proportional to its nasal dampness.

Horse A large stinky creature with sideways facing eyes.

Chapter 4

Maths Equations

4.1 Producing Beautiful Looking Mathematics

One of the best features about LaTeX is 'maths mode'. For example the schrodinger equation can be produced as follows:

$$\imath\hbar\frac{\partial}{\partial t}\Phi(x,t)=\hat{H}\Phi(x,t)$$

```
\begin{math}
\imath\hbar\frac{\partial}{\partial t}\Phi(x,t)=\hat{H}\Phi(x,t)
\end{math}
```

There are a number of ways to switch on maths mode. The first, as above, is with the \begin{math} environment. You can also enter mathmode inline using the $ symbol. For example typing in $y=axˆ2+bx+c$ yields $y = ax^2 + bx + c$. Another way is to enter the equation environment which enables you to number equations so you can then refer to them later in the text.

$$y(t)= \sin \left(\frac{{\alpha}t}{2\pi} + \phi_0\right) \tag{4.1}$$

```
\begin{equation}
y(t)= \sin \left(\frac{{\alpha}t}{2\pi} + \phi_0\right)
\end{equation}
```

.

4.2 Basic Maths Mode

Once in maths mode there is a kind of text based code for writing down your equations. Here are the most basic symbols to get you going.

Final Result	LaTeX Code
$a + b$	a+b
$a - b$	a-b
ab	ab
$a * b$	a*b
$a \times b$	a \times b
$a \cdot b$	a \cdot b
$\frac{a}{b}$	\frac{a}{b}
a^b	aˆb
a_b	a_b
$\sin a$	\sin a (same for cos, tan)
$sin a$	sin a
$\sqrt{a}$	\sqrt{a}
$\left(a \right)$	\left(a \right)
$\left[a \right]$	\left[a \right]
α	\alpha
π	\pi

A full treatise on maths mode is not practical here. There are lots of online tutorials and summaries of symbols. It just takes a bit of practice and you can build up equations really easily. It's straight forward to learn new stuff once you've done it a few times.

4.3 Equation Arrays

Sometimes you need to arrange several equations vertically, referencing individual lines separately and aligning the equations on the = sign. This can be achieved with equation arrays as follows:

$$\begin{aligned} A\left(x\right) &= \frac{x^2+2x+1}{x+1} && (4.2)\\ &= \frac{\left(x+1\right)\left(x+1\right)}{x+1} \\ &= x+1 \\ B(x,t) &= \frac{e^{\left(\imath\omega_0 t+kx\right)}}{4\pi\epsilon_0} && (4.3) \end{aligned}$$

```
\begin{eqnarray}
A\left( x\right) & = & \frac{x^2+2x+1}{1+x} \\
& = & \frac{\left(x+1\right)\left(x+1\right)}{1+x} \nonumber\\
& = & x+1 \nonumber\\
B(x,t) & = & \frac{e^{\left(\imath\omega_0 t + kx\right)}}{4\pi\epsilon_0}
\end{eqnarray}
```

- Note the & symbols. This tells LaTeX where to align the equations. There must be the same number of & symbols in each line.
- Note the \\ at the end of each line except the last one. This symbol tells LaTeX to add another row in the array. If you put it on the last line you get an empty row at the bottom of the array.
- Note the \nonumber command which suppresses line numbering for that line.
- Note that equation number carries on from equation 4.1 in the previous section.

4.4 Maths Packages

Maths mode comes as standard in LaTeX, however you can download packages that buff up your maths symbol set. For example neat vector notation comes in the package ‘vector’. e.g. \uuvec{T} yields $\underline{\hat{T}}$.

```
\usepackage{amssymb}
\usepackage{amsmath}
\usepackage{vector}
```

Chapter 5

Chemical Equations

The mhchem package means you can do basic stuff very easily using \ce{}. For example:

$$\mathrm{CO_2 + C \longrightarrow 2\,CO}$$
$$\mathrm{CO_2 + C \longleftarrow 2\,CO}$$
$$\mathrm{CO_2 + C \rightleftharpoons 2\,CO}$$
$$\mathrm{A{-}B{=}C{\equiv}D{-}E{=}F{\equiv}G}$$

```
\ce{CO2 + C -> 2CO}
\ce{CO2 + C <- 2CO}
\ce{CO2 + C <=> 2CO}
\ce{A-B=C#D\sbond E\dbond F\tbond G}
```

You can also use math mode within chemical equations.

$$x\,\mathrm{Na(NH_4)HPO_4} \xrightarrow{\Delta} \mathrm{(NaPO_3)}_x + x\,\mathrm{NH_3}\uparrow + x\,\mathrm{H_2O}$$

```
\ce{$x\,$ Na(NH4)HPO4 ->[\Delta](NaPO3)_{$x$} + $x\,$ NH3 ^ + $x\,$ H2O}
```

And you can number chemical reactions as well by using the math mode equation environment.

$$\mathrm{CO_2 + C \rightleftharpoons 2\,CO} \tag{5.1}$$

```
\begin{equation}
\ce{CO2 + C <=> 2CO}
\end{equation}
```

However, this will number mathematical and chemical equations using the same number system. There is a discussion in the mhchem pdf which shows you how to number chemical and mathematical equations independently. It's a bit complex and involves you creating your own type of environment... bit beyond the intro level of this course...

Chapter 6

Tables

Tables are big business in Latex. I use these packages to help me.

```
\usepackage{multirow}
\usepackage{booktabs}
\usepackage{dcolumn}
```

Here's an example table

Fibre Type	N	M	L_t (nm)	R_t (nm)	d_t (nm)	ΔZ_t (nm)	
SS Twisted	5	4	313 (122)	5.7 (1.6)	5.4 (1.0)	150.5 (63.1)	
			L_s (nm)	R_s (nm)	W_s (nm)	ΔZ_s (nm)	ΔZ_e (nm)
SS Spiral	32	7	123 (31.6)	11.4 (3.3)	11.1 (2.6)	32.9 (22.2)	26·9
SSB Spiral	64	18	106 (22.9)	10.8 (2.7)	9.8 (1.8)	28.2 (8.1)	22·7
BSS Spiral	37	19	110 (44.9)	13.4 (3.0)	12.9 (1.8)	27.1 (9.1)	21·5
SSSB Spiral	26	6	104 (23.9)	13.3 (2.6)	16.4 (3.4)	29.9 (11.1)	22·5

Table 6.1: A funky table from my thesis.

Here's what the code looks like:

```
\begin{table}[!hb]
 \centering
\begin{tabular}{@{}rccc@{~}r@{.}l*3{r@{.}l@{~}r@{.}l}D{.}{\cdot}{2,1}}
\toprule
\multicolumn{1}{c}{Fibre Type}
& N
& M
&\multicolumn{3}{c}{$L_t$ (nm)}
&\multicolumn{4}{c}{$R_t$ (nm)}
&\multicolumn{4}{c}{$d_t$ (nm)}
&\multicolumn{4}{c}{${\Delta}Z_t$ (nm)}
&\multicolumn{1}{c}{~}\\
\cmidrule(lr){1-1}
\cmidrule(lr){2-2}
\cmidrule(lr){3-3}
\cmidrule(lr){4-6}
\cmidrule(lr){7-10}
\cmidrule(lr){11-14}
\cmidrule(lr){15-18}
  SS Twisted & 5 & 4 & 313 &\multicolumn{2}{c}{(122)} & 5&7 &(1&6)  & 5&4 &(1&0)  &
  \multicolumn{19}{c}{~}\\
  %\cmidrule(lr){3-18}
  &
  &
  &\multicolumn{3}{c}{$L_s$ (nm)}
  &\multicolumn{4}{c}{$R_s$ (nm)}
  &\multicolumn{4}{c}{$W_s$ (nm)}
  &\multicolumn{4}{c}{${\Delta}Z_s$ (nm)}
  &\multicolumn{1}{c}{${\Delta}Z_{e}$ (nm)}\\
\cmidrule(lr){4-6}
\cmidrule(lr){7-10}
\cmidrule(lr){11-14}
\cmidrule(lr){15-18}
\cmidrule(lr){19-19}
  SS Spiral& 32 & 7 &123&(31&6)&11&4&(3&3)&11&1&(2&6)&32&9&(22&2)&26.9\\
  SSB Spiral& 64&18 &106&(22&9)&10&8&(2&7)&9&8&(1&8)&28&2&(8&1)&22.7\\
%\midrule
  BSS Spiral&37 &19 &110&(44&9)&13&4&(3&0)&12&9&(1&8)&27&1&(9&1)&21.5\\
%\midrule
  SSSB Spiral&26& 6 &104&(23&9)&13&3&(2&6)&16&4&(3&4)&29&9&(11&1)&22.5\\
\bottomrule
\end{tabular}
\caption[Basic Fibre Dimensions by TEM]{A funky table from my thesis.}
\label{tab:BasicXSBFibreDimensionsTEM}
\end{table}
```

6.1 Tables Made Easy

Here is a simple table followed by the code that produced it.

anchovy	banana	carrot
dog	apple	fennel
goat	strawberry	potato

```
\begin{tabular}{lcr}
anchovy & banana & carrot \\
dog & apple & fennel \\
goat & strawberry & potato
\end{tabular}
```

The tabular environment is a special case of the ”array” environment for distributing content uniformly across a region of the page. This ability can be exploited in mathematical equations which we’ll see that later on. For now though look at the first line.

```
\begin{tabular}{lcr}
```

This command tells LaTeX to enter the tabular environment. The letters l, c and r in the curly braces tell LaTeX to create a table with three columns in which the first column is left justified, the second column is centered and the third column is right justified. Lets add a fourth column and this time center justify all the columns.

anchovy	banana	carrot	Johnny
dog	apple	fennel	Pete
goat	strawberry	potato	

```
\begin{tabular}{cccc}
anchovy & banana & carrot & Johnny\\
dog & apple & fennel & Pete\\
goat & strawberry & potato &
\end{tabular}
```

Each row in the table is a list of items separated by the & symbol. The end of each row is denoted by \\. The last row in the table doesn’t have a \\. You do not have to have data between the ampersands but you must have the right number of ampersands to match the number of columns that LaTeX is expecting.

6.1.1 Adding Borders To Tables

Tables should never have vertical lines. No professionally typeset table contains vertical lines. Do not put vertical lines in your tables. That said it is easy to do.

anchovy	banana	carrot	Johnny
dog	apple	fennel	Pete
goat	strawberry	potato	

```
\begin{tabular}{|c|c|c|c|}
anchovy & banana & carrot & Johnny\\
dog & apple & fennel & Pete\\
goat & strawberry & potato &
\end{tabular}
```

Tables should have neatly headed columns with the heading for each field separated from the data by horizontal lines. The \toprule, \cmidrule and \bottomrule commands from the booktabs package are useful for controlling horizontal lines.

Ingredient 1	Ingredient 2	Ingredient 3	Source
anchovy	banana	carrot	Johnny
dog	apple	fennel	Pete
goat	strawberry	potato	

```
\begin{tabular}{cccc}
\toprule
Ingredient 1 & Ingredient 2 & Ingredient 3 & Source \\
\cmidrule(){1-4}
anchovy & banana & carrot & Johnny\\
dog & apple & fennel & Pete\\
goat & strawberry & potato & \\
\bottomrule
\end{tabular}
```

Note that when using the \bottomrule command you must add the \\ symbol to the last line of data. The last line of the table is now buried within the \bottomrule command.

6.1.2 The \cmidrule Command

This useful and versatile command takes a bunch of options to control subtleties like only putting lines across some of the columns, or not quite making them cross the full width of the column. The (lr) option trims the left and right ends of the lines off. For example:

Recipe Version	Ingredient 1	Ingredient 2	Ingredient 3	Source
10.1	anchovy	banana	carrot	Johnny
1.34	dog	apple	fennel	Pete
709.23	goat	strawberry	potato	

```
\begin{tabular}{ccccc}
\toprule
Recipe Version & Ingredient 1 & Ingredient 2 & Ingredient 3 & Source \\
\cmidrule(lr){1-1}
\cmidrule(l){2-2}
\cmidrule(){3-3}
\cmidrule(r){4-4}
\cmidrule(lr){5-5}
10.1 & anchovy & banana & carrot & Johnny\\
1.34 & dog & apple & fennel & Pete\\
709.23 & goat & strawberry & potato & \\
\bottomrule
\end{tabular}
```

6.1.3 Aligning Decimal Points Trick One

Note that the decimal points don't line up in the new column "Recipe Version" in the previous section. There are some tricks to deal with this.

Recipe Version	Ingredient 1	Ingredient 2	Ingredient 3	Source
10.1	anchovy	banana	carrot	Johnny
1.34	dog	apple	fennel	Pete
709.23	goat	strawberry	potato	

```
\begin{tabular}{r@{.}lcccc}
\toprule
\multicolumn{2}{c}{Recipe Version} & Ingredient 1 & Ingredient 2 & Ingredient 3 & S
\cmidrule(lr){1-2}
\cmidrule(lr){3-3}
\cmidrule(lr){4-4}
\cmidrule(lr){5-5}
\cmidrule(lr){6-6}
10&1 & anchovy & banana & carrot & Johnny\\
1&34 & dog & apple & fennel & Pete\\
709&23 & goat & strawberry & potato & \\
\bottomrule
\end{tabular}
```

My version numbers now use two columns. One is right justified and the other is left justified. The sequence of symbols @{.} tells LaTeX to replace the border between the columns with a period[1]. If I place the part of the number before the period in the first right justified column and the part of the number after the period in the second left justified column then, with the border between the adjacent table cells delineated by a period, it has the appearance of a decimal number, but all the numbers in the table are now aligned at the decimal point.

I also added the command \multicolumn{n}{j}{text} to span the heading across both columns. The parameters of the \multicolumn command are n = number of columns to span, j = justification (l, c or r) and "text" is the data to appear in the column. There is a similar \multirow command for stretching information over several rows in a table which works in a similar way. Look it up.

[1]Notice that the vertical lines in the tables in section 6.1.1 are created by placing the | symbol at the border between the columns.

6.1.4 Aligning Decimal Points Trick Two

The previous trick for aligning decimal points does not centre the number under the heading with looks rubbish. You could have a shorter heading to reduce this effect or you can use the DColumn package which is a better more general solution for this problem. (I taught the first trick to you so you came across the multicolumn command and learned more about the subtleties of how tables work, and to get you used to thinking in Tablese.).

$RecipeVersion$	Ingredient 1	Ingredient 2	Ingredient 3	Source
10·1	anchovy	banana	carrot	Johnny
1·34	dog	apple	fennel	Pete
709·23	goat	strawberry	potato	

```
\begin{tabular}{D{.}{\cdot}{4,4}cccc}
\toprule
Recipe Version & Ingredient 1 & Ingredient 2 & Ingredient 3 & Source \\
\cmidrule(lr){1-1}
\cmidrule(lr){2-2}
\cmidrule(lr){3-3}
\cmidrule(lr){4-4}
\cmidrule(lr){5-5}
10.1 & anchovy & banana & carrot & Johnny\\
1.34 & dog & apple & fennel & Pete\\
709.23 & goat & strawberry & potato & \\
\bottomrule
\end{tabular}
```

The DColumn package defines a new type of column which can be invoked placing a capital D in the \tabular command which defines the table. If defined using a D, then the column is placed in mathmode. D takes four parameters: D{a}{b}{c,d} where a is the symbol which is to be aligned, b is the symbol with which to replace the aligning character, and c,d must be integers which indicate LaTeX should have up to c white space characters before the aligning character and d afterwards, thereby defining the position of the number within the column. The command \cdot prints a special type of mathmode symbol which is a dot that is vertically shifted and larger than a normal period e.g.: ·.

6.1.5 Final Table Trick

DColumn forces the column to be in mathmode which is why the heading ”Recipe Version” was typeset in italics in the previous section. Indeed LaTeX tried to interpret the heading of the table as a number to be aligned. We can over ride this behaviour by using the \multicolumn command to locally impose a different type of justification and temporarily disable mathmode as follows:

Recipe Version	Ingredient 1	Ingredient 2	Ingredient 3	Source
10·1	anchovy	banana	carrot	Johnny
1·34	dog	apple	fennel	Pete
709·23	goat	strawberry	potato	

```
\begin{tabular}{D{.}{\cdot}{4,4}cccc}
\toprule
\multicolumn{1}{c}{Recipe Version} & Ingredient 1 & Ingredient 2 & Ingredient 3 & S
\cmidrule(lr){1-1}
\cmidrule(lr){2-2}
\cmidrule(lr){3-3}
\cmidrule(lr){4-4}
\cmidrule(lr){5-5}
10.1 & anchovy & banana & carrot & Johnny\\
1.34 & dog & apple & fennel & Pete\\
709.23 & goat & strawberry & potato & \\
\bottomrule
\end{tabular}
```

6.2 Numbering Tables

To tell LaTeX to assign a number to a table and add it to the list of tables you must use the \begin{table} command to tell LaTeX to create a table environment as follows:

Recipe Version	Ingredient 1	Ingredient 2	Ingredient 3	Source
10·1	anchovy	banana	carrot	Johnny
1·34	dog	apple	fennel	Pete
709·23	goat	strawberry	potato	

Table 6.2: Recipes that ought to be banned.

```
\begin{table}[!bh]
\centering
\begin{tabular}{D{.}{\cdot}{4,4}cccc}
\toprule
\multicolumn{1}{c}{Recipe Version}& Ingredient 1 & Ingredient 2 & Ingredient 3 & So
\cmidrule(lr){1-1}
\cmidrule(lr){2-2}
\cmidrule(lr){3-3}
\cmidrule(lr){4-4}
\cmidrule(lr){5-5}
10.1 & anchovy & banana & carrot & Johnny\\
1.34 & dog & apple & fennel & Pete\\
709.23 & goat & strawberry & potato & \\
\bottomrule
\end{tabular}
\caption[Table of Banned Recipes]{Recipes that ought to be banned.}
\label{tab:Recipes}
\end{table}
```

In the table environment the \caption[text1]{text2} command adds a caption, where text1 appears in the list of tables at the beginning of the document and text2 is the local caption. The label command creates a label with which to reference the table e.g. Table 6.2 is a table of recipes that have been made up to illustrate how to use tables in LaTeX. We also use the \centering command to center the table and caption within the table environment. We could also use the \begin{center} and \end{center} commands.

Chapter 7

Adding Figures To Your Document

7.1 My First Figure

Adding figures is easy in LaTeX. You just create a figure environment which is much the same as the table environment. For example:

```
\begin{figure}[!th]
\centering
\includegraphics[width=137.4mm]{Images/haemStructure.png}
\caption[Haem Structure]{A fancy image from Chris' Thesis.}
\label{fig:haemStructure}
\end{figure}
```

In this example the command \includegraphics tells LaTeX to look in the directory 'Images' and incorporate the file called 'haemStructure.png' into the final document while setting the width to 137.4mm. This command can be used to resize a graphical file using the height or width parameter as shown. You must specify the units which can be pt, ex, em, mm, cm and so on. LaTeX recognises a very wide variety of standard units and graphics formats. The command is part of the graphicx package and so in the preamble you must include the command \usepackage{graphicx}.

Many packages have been written to provide new commands. For example the 'subfigure' package enables you to place separate images or files in the same figure side by side while giving them their own label. However, in this document we only concentrate on the very basics.

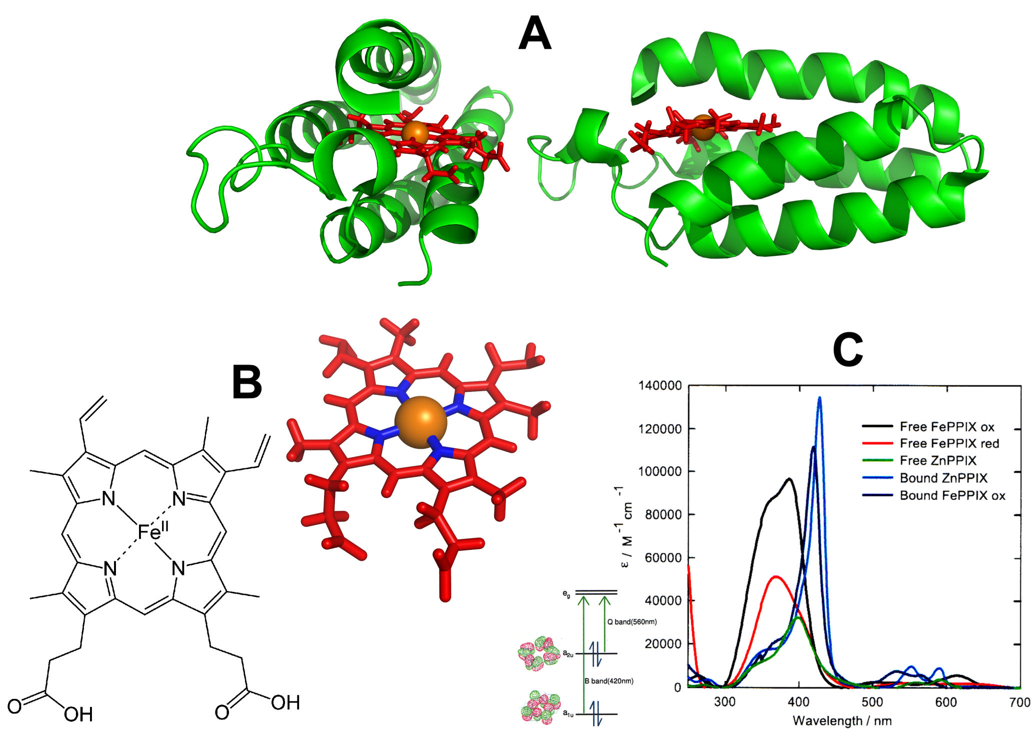

Figure 7.1: A fancy image from Chris' Thesis.

7.2 Floating Environments

Both tables and figures are examples of a 'floating environments' which means LaTeX decides where to put them.

The mysterious [!tbh] letters appearing everywhere give guidance to LaTeX about roughly where the figure or table is allowed to be, but in general they can move a long way from where you position them within your text file. The letters in the square bracket can be t, h or b which stand for top, here or bottom. If you don't specify any letters LaTeX defaults to [t].

LaTeX will always preserve the order in which figures appear. If it cannot find a type setting solution, then it may move the float to it's own page, and combine it with other figures. If it still can't fit that in for some reason then it moves the float to the end of the document. This has the effect of pushing all the remaining floating environments to the end of the document also.

The exclamation mark instructs LaTeX to "try harder" at putting the float where you told it to. Often you must play around to ensure the float positioning is acceptable, but usually this can be achieved by stretching or shrinking the image slightly using the size parameter, re-ordering text, judicial applications of the \pagebreak command or by shouting loudly and slapping the computer monitor about.

Chapter 8

The Chapter on Making Chapters

Right. LaTeX forces you to structure your document. There a series of simple commands for achieving this:

```
\chapter{chapterName}
\section{sectionName}
\subsection{subsectionName}
\subsubsection{subsubsectionName}
```

Easy enough. At the start of each chapter or section just issue one of these commands to name the section and LaTeX will present the chapter heading in the right style of font, which will be the same style of all the other heading names at that level of hierarchy throughout the document. In addition LaTeX assigns a number to that section. For example, this chapter was created using the command:

```
\chapter{The Chapter on Making Chapters}
```

LaTeX assigns the correct chapter number to each chapter in turn and then puts it in the table of contents, as you can see by looking at the table of contents! Easy.

8.1 The Section About Sections

I think you're getting the hang of this. This section was created using the command \section{The Section About Sections}. It appears in the table of contents as section 8.1.

8.1.1 The Sub-Section About Sub-Sections

Now you're really getting the hang of this. This subsection was created using the command \subsection{The Sub-Section About Sub-Sections}.

The Sub-Sub-Section About Sub-Sub-Sections

Now you're really getting the hang of this. but.. caught you out! This sub-subsection doesn't have a number! Ha Ha! It was created using the command \subsubsection{The Sub-Sub-Section About Sub-Sub-Sections} in the same vein as all the other subsections/chapters etc. Here we have stopped the depth of the section numbering at the second level with the command:

```
\setcounter{secnumdepth}{2}
```

Easy. WYSIWYM. LaTeX does what you tell it to do. This is both a blessing and a curse.

The Section About Sections That Don't Appear in the Table of Contents

Ok. Now we're throwing a spanner in the works. This section was created using the command:

\section*{The Section About Sections That Don't Appear in the Table of Contents}.

Notice how LaTeX has not assigned a number to the section and it doesn't appear in the table of contents. The effect of the * is to suppress the inclusion of a chapter, section or subsection in the automatic numbering. Easy. You can do this at any level. This is useful for things like prefaces, tables of contents or acknowledgements which you may or may not wish to have an entry in the main contents table. Up to you. It's your thesis. Don't just copy me.

8.2 A Small Point About Numbering

In this section notice how the numbering starts from where it left off before we suppressed the numbering on the previous section. Easy. WYSIWYM.

8.3 The Section About Internal Referencing

There are two related commands:

\label{labelName}
\ref{labelName}

The \label{} command allows you to create a label in a particular environment[1]. The label won't appear in the final document. It's just a label which makes it easy to refer back to any particular environment elsewhere in the document. The \ref{} command enables you to insert a reference anywhere in the document to any label in the document. For example this is section 8.3. The names you use in a label can be anything you like but musn't contain whitespace or special characters. I used the two commands:

\label{sec:InternalReferencing}
\ref{sec:InternalReferencing}.

The astute among you will realise that LaTeX has to read the document several times. Once to find the labels and then again to populate the references with the correct numbers. So you have to compile a latex document twice to get the referencing right. If there is a missing label or you refer to something that doesn't exist then latex inserts a convenient ? at that point. So hunting for queries is useful way of finding broken references. Latex issues warnings when it finds broken references.

As you create a document you will find yourself putting labels in all over the place so choose a sensible naming convention to help you remember the label names.

Each type of environment (equations, figures, tables, sections etc) has its own independent numbering system. So when you choose your label name it's a good idea to have an identifier for that type of environment. I have my own convention for label names which I use to help me remember references. E.g. sec:SectionAboutCats. Eqn:EquationAboutCats, Fig:FigureAboutCats and so on. This means you can differentiate between referring to the section or the figure more easily, even though they are about the same thing.

So Internal Referencing is a doddle. Easy!

[1]That's right. Chapters, Sections and sub-sections etc are environments!

Chapter 9

Organising Your Files

One of the drawbacks of word is that the figures and tables are included within the document which becomes very very clunky quite quickly and once it reaches around 100Mb it becomes unwieldy and hard to export. One of that advantages of LaTeX is that the files are all text files and so are very light on resources and are easy to email around the place.

However, sometimes it can be difficult to navigate through a single text file, so if you had a huge document then it becomes difficult to find your way around. One way of circumnavigating this problem is to break the file up into smaller files. There is no need to have all the instructions in one file as long as LaTeX knows where to look when it compiles the output. So you can have file for each logical sub-division of your document and then tell LaTeX how to link the files together in the final compilation.

Therefore it is worth spending a bit of time at the beginning of a large project deciding how you wish to break up your LaTex Files and organise them neatly and tidily. You can then tell LaTex where to look for the files using the following command:

```
\include{fileName}
```

9.1 Linking a File

We are now going to copy part of the tutorial file into a separate files and link it the main file. Because this could go wrong it is sensible to back up tutorial.tex as something else. So save a copy called 'tutorialOriginal.tex' for safe keeping. Then follow the following instructions.

1. Create a new file called MyFirstPage.tex
2. Select and Cut out the first chapter of Tutorial.Tex
3. Paste the first chapter into MyFirstPage.tex
4. Save both tutorial.tex and MyFirstPage.tex
5. Then add \include{MyFirstPage} in tutorial.tex where the text used to be before you cut it out. We don't need to add the '.tex' because LaTeX can only include '.tex' files.
6. compile tutorial.tex and view the PDF

The final PDF should look exactly the same as it was! except we now have a separate file for one of the chapters.

9.2 The Structure of this Document

Lets now look at how I've organised this document. It is identical to my thesis because it started out life as my thesis. I usurped the structure and simply replaced the headings and content to make a kind of reference manual for this introductory course. This document therefore is three things:

1. An example thesis structure to work from.
2. The LaTeX files which were used to create it can be used as a template for a thesis.
3. It is also a convenient introductory latex manual!

Groovy huh?

9.3 The Master File

There is a master file where LaTeX begins the compilation procedure. This contains the \documentclass command needed to set the ball rolling. Within the file there are a series of \include{} commands, one for each Chapter, which tell LaTeX where to find the files containing each chapter.

The subfiles cannot be compiled on their own because they do not contain the \documentclass, \begin{document} or \end{document} commands. Lets us look in more detail at the master file and we will learn some new LaTeX commands.

```
\documentclass{Style/LatexCourseStyle}

\begin{document}
\setcounter{secnumdepth}{2}
\setcounter{tocdepth}{1}

\frontmatter
\onehalfspacing
\include{Frontmatter/Titlepage}
\include{Frontmatter/Declaration}
\include{Frontmatter/Acknowledgements}
\include{Frontmatter/Summary}
\include{Frontmatter/Abbreviations}
\tableofcontents
\listoftables
\listoffigures

\mainmatter
\renewcommand{\sectionmark}[1]{\markright{\thesection \ #1}{}}
\include{Chapters/Overview}
\include{Chapters/MyFirstPage}
\include{Chapters/Lists}
\include{Chapters/mathsEquations}
\include{Chapters/ChemicalEquations}
\include{Chapters/Tables}
\include{Chapters/Figures}
\include{Chapters/Sectioning}
\include{Chapters/ThesisTemplateMainFile}
%\include{Chapters/External Referencing}

\appendix
%\include{Appendices/UsefulWebsites}

\singlespacing

\backmatter
%\addcontentsline{toc}{chapter}{References}
%\bibliographystyle{Bibliography/cjfthesisv1}
%\renewcommand{\bibname}{References}
%\bibliography{Bibliography/LatexCourseBib}

\end{document}
```

9.4 Analysing the Master File

Let us go through this file and observe the new commands. Most of it is self-explanatory.

```
\documentclass{Style/LaTeXCourseStyle}
```

This command tells latex that the document class is the one defined in the the style file "LaTeXCourseStyle", which is in the sub directory "style". Easy.

```
\setcounter{secnumdepth}{2}
\setcounter{tocdepth}{1}
```

The first of these commands tells latex that the section depth number is 2. I.E. throughout the document only bother numbering sections, subsections and subsub-sections down to the second level (such as section 1.2.3). The second command tells latex to only build the table of contents from section numbers down to the first level i.e. 1.1. Easy enough.

The command \frontmatter is exclusive to the book class, on which this style is based, and tells latex that within this section to number the pages using roman numerals, and not to include these numbers in the table of contents, just like in a book. The \mainmatter and \backmatter commands similarly delineate the other sections of the document.

Childs play. How long would it take you to figure out how to do that in word???? No idea. I've never tried.

Note that these rather cool commands are not available in every single class. So for a paper you would use the article class where such compartmentalisation of your document is not appropriate. In fact you can often download latex templates from the journal to wish to submit your paper, which is why I concentrate on the thesis template in this course.

```
\onehalfspacing
```

Is obvious. "From this point forward use line spacing of one and a half lines please. Thank you." How many hours have you spent clicking on paragraph in-dentations for each separate paragraphs in word? Nightmare. Want to change the standard in the whole document? Not a chance in word. In latex, you can do it with a single word, carefully placed in the master file. Grace personified.

In fact in general in latex you can override global settings locally. This is v. useful. Just switch it on and off at will as required.

```
\include{Frontmatter/Titlepage}
\include{Frontmatter/Declaration}
\include{Frontmatter/Acknowledgements}
\include{Frontmatter/Summary}
\include{Frontmatter/Abbreviations}
```

The \include{} command tells latex to include the specified .tex file at this point. Thus it is a simple matter to break up a large document into sub components. I have written each part of the front matter in a separate file and then grouped all the front matter into one directory called: Frontmatter.

See how the directory structure of my files reflects the structure of my book?

This is so easy and flexible it's unbelievable that humans are smart enough to come up with this stuff...

```
\tableofcontents
\listoftables
\listoffigures
```

Again. This is easy. These commands mean: "Please LaTeX, at this point in the creation of my document could you scan through the whole document and build for me a table of contents, a list of tables, a list of figures and then write down the entries along side their section numbers (as I have them ordered in this version of the document), as well as the page numbers on which they appear and create a nice little summary table and add them to the front matter of my document. Thanks.".

How does latex do this? remember the idea of environments? Well each table, figure, equation etc is a defined environment. LaTeX just counts them up and assigns numbers starting wherever you want and builds a table. You can tell latex to ignore individual sections if you want to.

You can also control the look and feel of your table of contents with subtley and grace. Do you want to fill the white space with lines of dots to aid readability, even though the chapter titles are all different lengths? page numbers? How deep should the TOC been? etc etc etc. How to do this is beyond the scope of this course, but now you know about it, go look it up. easy as pie.

Ok.

Main matter we've done.

```
\renewcommand{\sectionmark}[1]{\markright{\thesection \ #1}{}}
```

This command tells latex to 'renewcommand', i.e. change the meaning of the following command from this point in the document forward, over-riding previous definitions.

```
\sectionmark}[1]{\markright{\thesection \ #1}{}
```

Is latex speak for "please write the section heading at the top right hand side of each page. thanks". This works in conjunction with a command in the style file which we'll talk about later. You have to tell latex to switch on headings as well as where to put them.

```
\include{Chapters/Overview}
\include{Chapters/MyFirstPage}
\include{Chapters/Lists}
\include{Chapters/Tables}
\include{Chapters/Figures}
\include{Chapters/Maths Equations}
\include{Chapters/Chemical Equations}
\include{Chapters/ThesisTemplateMainFile}
\include{Chapters/Sectioning}
%\include{Chapters/User Defined Commands}
%\include{Chapters/External Referencing}
```

This is the include command again. You will notice that some of the include lines are commented out. This is because I haven't written those files yet, if I tried to compile and the file wasn't there then latex would politely inform me that it was unable to find the file. In general I can comment out lines in the main file to suppress their inclusion in a shorter version of my thesis. Say I wish to build just one chapter, all I need to do is put % signs everywhere except the chapter that I want to include. This will recreate the table of contents with only the chapters that are included.

```
\appendix
%\include{Appendices/UsefulWebsites}

\singlespacing

\backmatter
%\addcontentsline{toc}{chapter}{References}
%\bibliographystyle{Bibliography/cjfthesisv1}
%\renewcommand{\bibname}{References}
%\bibliography{Bibliography/LaTeXCourseBib}
```

Finally the rest is easy. \Appendix means we are in the appendix now and start labelling sections with A.1.2 instead of 1.2.3.

\addcontentsline means forcibly insert a contents line about the references into the contents table.

\{bibliographystyle{Bibliography/cjfthesisv1}} means use the biblography style defined in the file cjfthesisv1. Defining a bibliography style will be covered later if we have time. This is where you specify how the references will appear (names, journals, full stops, commas etc).

\biblography means add the following bibliography. You can reference entries in this bibliography file very easily.

The astute among you will realise that LaTeX will have to read the document several times before it can be compiled with all the references. Hey ho. No problem. You can go and have a cuppa while LaTeX does your references for you in the style of your favorite journal... nice.

9.5 Creating A Class File

You can also use the \LoadClass{} command to allow your preamble to be saved in a separate file. This makes your main document much easier to understand.

1. Create another new file and save it as TutorialStyle.cls
2. Copy the preamble of tutorial.tex into the new file. Starting with the \documentclass command and copy right up to, but not including, the \begindocument command. Delete the preamble in the original document.
3. Save TutorialStyle.cls and tutorial.tex
4. Modify the first line of TutorialStyle.cls by replacing the word 'document' with the word 'Load'.
5. Modify the first line of tutorial.tex to become \documentclass{TutorialStyle}
6. Save both files.

9.5.1 Sanity Check

The first line of TutorialStyle.cls should now be:

```
\LoadClass[12pt, oneside, a4paper]{book}
```

The first line of tutorial.tex should be:

```
\documentClass{TutorialStyle}
```

We have just created what is called a style file. You can use this style file as a generic preamble for other documents as well. Compile tutorial.tex. Nothing should have changed in the final PDF.