A Suggested Thesis or Dissertation Layout

Original document by Nathan Rountree
(Department of Computer Science), amended
for Physics in 2020 by Annika Seppälä

submitted in partial fulfilment of the degree of

Bachelor of Science, with Honours

at the University of Otago, Dunedin,
New Zealand.

9 February 1998

Abstract

An abstract should be somewhere between 50 and 200 words long—the absolute maximum at Otago is 500 words, but that is almost certainly too long. Remember, you will probably get at least one article out of your thesis and it would be nice to be able to recycle the abstract without having to snip it.

All you need to do in the abstract is give a statement of the problem, a *brief* explanation of the method and procedures used, and a summary of conclusions. You do **not** have to give away your conclusions entirely; save that for the concluding chapter. Bear in mind that the abstract is all that most people will bother to read, so it had better be good!

Acknowledgements

This is the *acknowledgements* environment, defined in `otagothesis.sty`. If you have any preface material, this is where to put it.

I would like to thank the following people:

- The Department of Computer Science, for employing me over the Summer break in 1997/8;
- My colleagues in the Lab, who *still* haven't complained about the mess;
- Everyone who puts up with me.

Thank you all.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Preamble

The `otagothesis` package consists a style file and document template for writing theses and dissertations at the University of Otago. It was intended only for computer science students, but seems to have found wider popularity. Its purpose is to take the hard work of constructing a large document out of your hands.

Earlier versions of this package used Latex to produce a DVI file which could be converted to PostScript and printed. There are several advantages to using Pdflatex instead: the document goes straight to PDF, and may include graphics files of many types instead of only EPSF. This package now uses Pdflatex instead of Latex. If you wish to use it in place of an earlier version, you may need to convert any EPSF diagrams to PDF with the `epstopdf` program, and you should remove the `[dvips]` option from the `\usepackage{graphicx}` command.

This document is a quick tour of the `otagothesis` package and a very brief comment on how to organise a thesis in Computer Science. For an even more brief introduction to the package, read the README file. Please bear in mind that this is not a style guide, nor an example of good content—although you will find some ideas on how to cite other authors' work, and how to include tables and figures in your document. These are all specific to the Systems Research Lab at the Computer Science Department; all other students should seek advice from their supervisors before following *any* of the advice contained herein!

It should be very easy to convert the file `thesis.tex` to your own use. The `otagothesis.sty` file is not quite as flexible, but feel free to hack around with it if you wish—just be sure to pass on the original to others (or take my name off the changed version). The idea is that if you are an Otago University student you shouldn't need to change `otagothesis.sty` at all.

1.2 Files You Need

There are several files you need to have a copy of before you will be able to compile this document for yourself. These files are included in the original set of files. They are:

- `otagothesis.sty`: this is the file which provides the `otagothesis` package. It contains some useful commands and options, and formats your pages according to University library restrictions.
- `logo.pdf`: the Otago University crest, required for your library declaration page.
- `natbib.sty`: this is the file which provides Harvard style (author, date) citations. It is provided in TeTeX, the distribution of LaTeX that comes with Redhat Linux.
- `plainnat.bst`: this provides Author-date style bibliographies. It is also standard in TeTeX.

The easiest way to use the otagothesis package is to put `otagothesis.sty` and `logo.pdf` in the same directory as your document. At its simplest, put `\usepackage{otagothesis}` just after the `\documentclass{report}` declaration, and you're away. For other usage options, follow the pattern in the `thesis.tex` file.

1.3 Compiling

Just to see if everything is working, try compiling this document by unzipping the distribution, changing into the `example_document` directory, and typing 'make'. You should see a whole lot of messages printed to the screen, followed by a preview window showing you a title-page. If you don't get this, try reading Subsection 1.3.1 on troubleshooting the distribution. If everything seems to be OK, go on to Section 1.4 which tells you a bit about using these files.

If you don't like previewing your document every time you recompile it, read Subsection 1.3.2 which has some suggestions for changing the makefile.

1.3.1 Troubleshooting

Here are some of the things that might go wrong on the initial compilation and how to fix them.

- `LaTeX Error: File 'otagothesis.sty' not found`:
  The simplest solution is to put the relevant file(s) into the same directory as

your document. The files `otagothesis.sty` and `logo.pdf` are in this distribution, and `natbib.sty`, `plainnat.bst`, and `moreverb.sty` are all available on `http://www.ctan.org`. This can also happen if your latex distribution doesn't have the `natbib` package installed, in which case it will complain about not having the files `plainnat.bst` and `natbib.sty`. Other files that could be missing are `logo.pdf` or `moreverb.sty`.

- The makefile doesn't work:
  This makefile was written for GNUmake, and isn't guaranteed for any other version of make. If you need to compile without make, type the following:

  ```
  pdflatex thesis
  bibtex thesis
  pdflatex thesis
  pdflatex thesis
  ```

  (also known as the LaTeX stanza). The extra `pdflatex` at the end is to get all the cross-references right.

- I printed it up, and it's all off to one side!
  It's supposed to be. The bindery wants you to have a big binding margin on the left of recto pages and on the right of verso pages. If you are printing up using the one-sided option, then all of the pages should have a wider margin on the left.

- Any other problems are caused by one of two things: either you have jumped the gun and edited a file, or your installation of LaTeX doesn't look like mine. You need the standard LaTeX 2_ε packages for it all to work; in particular the `moreverb`, `graphicx`, and `natbib` packages are absolutely crucial.

- Many people in the department use LaTeX, ask around for help! Internet is also great for finding solutions to LaTeX problems.

1.3.2 The Makefile

The makefile for this package is called (appropriately enough) "Makefile". You will find detailed instructions for its use in the header comments. A makefile tries to figure out what needs to be done every time you compile your thesis, depending on what was last changed. It can save you a lot of typing.

Briefly: typing `make` will create a pdf file viewable with Acrobat reader, MacOsX's Preview, or Linux's gv. Typing `make clean` will remove all auxilliary files, and `make`

`touch` will prepare for a complete re-compile. If you want to suppress automatic viewing, find the `AUTOVIEW` variable in the makefile and set it to `no`.

1.4 The `otagothesis` Package: Overview

The `otagothesis` package has been designed using the guidelines in the booklet "Notes on the Preparation of Theses", available from the reference desk at the Central Library. I strongly suggest that you obtain a copy of this and read it.

Open up the `thesis.tex` file to see how the `otagothesis` package has been invoked and what sorts of options it gives you. There are lots of comments to explain what each part is for.

The package has the following features:

- A 30mm binding margin. If you are printing up single sided, this will be on the left of every page. If you are printing up double sided, it will be on the *left* of *recto* (right-hand-side) pages, and on the *right* of *verso* (left-hand-side) pages;
- Automatic title-page and contents page generation;
- Automatic insertion of abstract and acknowledgements files;
- A wide margins option for proofreading purposes (so your supervisor has room for the red pen!);
- Inclusion of the bibliography in the table of contents, renamed "References".

The following subsection lists all of the package's options in detail.

1.4.1 Package Options

To use the package, all you need to do is include the line

```
\usepackage[option1,option2, ... ]{otagothesis}
```

in your main document. With no options at all, this will give you the commands available in the package (see the next subsection) and will format your document with the correct margins. It assumes you are doing a PhD. The available options are:

`nofigures` Suppresses the production of a "list of figures" page.

`notables` Suppresses the production of a "list of tables" page.

`nolibrary` Suppresses the production of a library declaration page.

thesistype Enter a thesistype to produce the correct title-page. Available thesistypes are: `phd`, `msc`, `ma`, `mcom`, `interimsci`, `interimarts`, `interimcom`, `dipsci`, `diparts`, `dipcom`, `bschons`, `bahons`, `bcomhons`. The package will default to `phd`.

1.4.2 Package Commands

There is only one command provided with the package, and it is designed to help you print draft copies. The command is

```
\frontstuff
```

and it places all the preface material before Chapter 1. If you don't issue this command in the document, you won't get a title-page, table of contents, etc.—your thesis will just print from Chapter 1.

1.5 Other packages included in the template

Physics department has added some other useful package options.

1.5.1 `amssymb` and `amsmath`

These are the essential packages for writing math and physics formulas. For a short guide to usage in your latex document see http://mirror.aut.ac.nz/CTAN/info/short-math-guide/short-math-guide.pdf

For example (this is an example based on the above document):

```
\begin{equation}\label{first}
\alpha=b+c
\end{equation}
some intervening text
\begin{subequations}\label{grp}
\begin{align}
\alpha&=b+c\label{second}\\
d&=e+f+g\label{third}\\
h&=i+j\label{fourth}
\end{align}
\end{subequations}
```

Produces the following:

$$\alpha = b + c \tag{1.1}$$

some intervening text

$$\alpha = b + c \tag{1.2a}$$
$$d = e + f + g \tag{1.2b}$$
$$h = i + j \tag{1.2c}$$

1.5.2 siunitx

A great package for typing units in and out of equation mode! A comprehensive guide is available at http://tug.ctan.org/macros/latex/exptl/siunitx/siunitx.pdf. For example

```
\si{kg.m.s^{-1}} \\
\si{\kilogram\metre\per\second} \\
\si[per-mode=symbol]{\kilogram\metre\per\second} \\
\si[per-mode=symbol]{\kilogram\metre\per\ampere\per\second}
```

Produces:
$\mathrm{kg\,m\,s^{-1}}$
$\mathrm{kg\,m\,s^{-1}}$
$\mathrm{kg\,m/s}$
$\mathrm{kg\,m/(A\,s)}$

`siunitx` is particularly useful when typing unit within an equations: The standard equation mode will italicise letters:

$$1m + 1m = 2m$$

`siunitx` prevents that:

```
\[ 1\si{.m} + 1\si{.m} = 2\si{.m} \]
```

$$1\,\mathrm{m} + 1\,\mathrm{m} = 2\,\mathrm{m}$$

1.5.3 hyperref

The `hyperref` package generates clickable links within the compiled pdf file. Anytime you reference literature using `\citet{} ,\citep{}` etc. or reference a Chapter, Figure or Table using `\ref{}` http://otago.ac.nz

1.6 Using the Thesis Template

The main aim of this package is to allow you to write completely vanilla LaTeX and have everything come out in the right place. All you need to do is know how to create each chapter file, how to include them in the main document, and how to print up a draft as well as the main copy.

So go ahead and write chapters which have LaTeX formatting commands in them, and `\include` them into the `thesis.tex` file. If you are using `make`, add the name of the chapter file to the makefile (including the `.tex` suffix). Type `make` and bingo: one perfectly formatted thesis!

1.6.1 Printing a Final Copy

Make sure you have issued the command `\frontstuff` directly after `\begin{document}`. You should also comment out the line beginning with `\includeonly`, which restricts which chapters you are going to print up. If you want to include an abstract and acknowledgements, you should have files `abstract.tex` and `acknowledgements.tex` in the same directory as `thesis.tex` and they will be included automatically.

Now all you have to do is type `make` to preview your document. If you are using the `twosided,openright` option, there will be quite a few blank pages because new chapters will always start on a recto (right-hand-side) page.

The final document is in PDF format, so you can print it either through a pdf viewer such as Acrobat, or by using the `pdftops` program to produce a postscript version that you can send to the printer with `lpr`.

1.6.2 Printing Specific Chapters

Let's suppose that you have 4 chapters done and are working on Chapter 5. Compile time is getting pretty long; it would be nice if you could generate only Chapter 5, but retain correct pagination and references. Here's how:

1. Comment out the `\includeonly` line and type `make` to create a complete set of `.aux` files.

2. Uncomment the `\includeonly` line and enter the names of the chapters you wish to print up (without the `.tex` extensions).

3. Type `make` to preview the document. Only the desired chapters and their references should be printed up, but the page numbers and cross-references should all be correct.

1.7 More to Come

In Chapter 2 we will look at how to use the Harvard style citations and references provided by the `natbib` package. Then, in Chapter 3, we discuss how to include figures in your document. Chapter 4 has some examples of how to include code snippets in the body of your text, and how to include code in the appendices. Next, in Chapter 5, some examples of LaTeX tables are presented. Finally, Chapter 6 gives some dos and don'ts regarding style and content.

Chapter 2

Literature Survey

Since this chapter would normally contain your literature review and background sections, it seems like a good time to discuss citations and bibliographies. The file `thesis.tex` is set to use Harvard style citations, using the `natbib` package. The following sections describe how to use this package effectively.

2.1 Reference Lists

The `otagothesis` package automatically renames your selected bibliography to "References", since in Computer Science we only list works that are actually referred to in the text. We use the Harvard style of citations as implemented by the `natbib` package, which is part of the TeTeX distribution of LaTeX (i.e. the one found on most versions of Linux).

To get all references and citations correct, you need to perform the LaTeX stanza (see Subsection 1.3.1): `pdflatex, bibtex, pdflatex, pdflatex`. The makefile does this for you when you type `make`.

Every time you use a `\citet` or `\citep` command, a bibliographic entry will be added to your reference list. On the whole, BibTeX will get the styles correct, but you should definitely check this before submitting a final copy; BibTeX bibliographies are notoriously difficult to spell-check and some strange things can happen with capitalisation.

2.2 Creating a Bibliography File

If you look at the file `thesis.bib` you will find some instructions on how to create a BibTeX database, and some sample entries. The makefile and thesis template are set up so as to *expect* to see a file called `thesis.bib` (your bibliographic database file)

and will generate an error message if there isn't one. You can find a large number of Computer Science citations in BibTeX format on the Collection of Computer Science Bibliographies Homepage, at `http://liinwww.ira.uka.de/bibliography/`.

A bibliographic entry looks something like this:

```
@InProceedings{amir97,
  author =   {Amihood Amir and Ronen Feldman and Reuven Kashi},
  title =   {{A New and Versatile Method for Association Generation}},
  booktitle =   {{Principles of Data Mining and Knowledge Engineering;
                  First European Symposium, PKDD'97}},
  OPTcrossref =  {},
  OPTkey =   {},
  editor =   {Jan Komorowsk and Jan Zytkow},
  OPTvolume =   {},
  OPTnumber =   {},
  OPTseries =   {},
  year =   {1997},
  OPTorganization = {},
  publisher = {Springer-Verlag},
  address =   {Trondheim, Norway},
  month =   {June},
  pages =   {221--231},
  OPTnote =   {},
  OPTannote =   {}
}
```

Separate authors with the word "and". The title is in double braces to maintain the capitalisation exactly as written, as is the title of the conference proceedings. The very first part of the entry (amir97) must be a unique key—this is what you will use when referencing the work in your document, so make it something easy to remember. Most people use something like "first_author_year" with a letter tacked on the end to avoid duplicates (e.g. amir97a, amir97b).

Here are some suggestions for creating your own database:

- Leave optional fields in place, even when they are empty. You never know when that information might turn up, so all you have to do is add it into the correct field and remove the OPT prefix.

- Use double braces to maintain your own capitalisation. You can always remove them later using a search-and-replace, but it is a real pain to go through the database putting them in.

- Use the emacs macro to create each new entry (M-x bib-TAB-TAB for a list of options). You will be thankful for the consistency as the list grows.

- Use "@InProceedings" for an article in conference proceedings, not "@Proceedings". Use "@InCollection" for an article which forms a stand-alone chapter in a book where each chapter is written by a different author, not "@InBook".

2.3 Citation Styles

The most usual form of citation is an "aside" (in parentheses, like this). The time to use it is when you have made a statement which might be questionable, and you wish to give a source for it. For instance:

> Writing a thesis is easy (Rountree, 1998).

This type of citation is made using the `\citep` command. It should be used sparingly, because it lends itself most to sweeping, general statements (of which you don't want too many!).

The `natbib` package gives you another option with the `\citet` command. This enables you to refer directly to a piece of work, using it as a noun in your sentence. For instance:

> The article by Amir, Feldman, and Kashi (1997) describes a method of preprocessing a database into a trie, so as to make association generation much quicker.

This style is more versatile, and should be favoured over the use of `\citep`.

When there are multiple authors, all subsequent citations of that work will just list the first author followed by "et al.". This will happen automatically; for instance:

> Here is a second citation of Amir *et al.* (1997).

If for some reason you want the whole list of authors, you can use the "starred" form of the command, i.e. `\citep*` or `\citet*`.

Sometimes, it will be inelegant to include the year with the citation; for example when you are referring to a particular article very frequently in a single section, (perhaps when a chapter forms a criticism of another piece of work). For those purposes, use the command `\citeauthor`. This will allow a sentence like "the book is of no use to anyone, but Rountree published it anyway."

The command `\citeyear` is similar, but gives just the year, allowing you to occasionally break style if you wish—and `\citeyearpar` will give the year with parentheses.

This will let you write a sentence like "Amir *et al.*'s article, published in 1997, is the most up-to-date treatment of this topic."

If you wish to add text to the citation (for instance the letters e.g., or some page numbers), you may do so either at the beginning or the end. For the end (as with page numbers), simply add an optional argument; `\citep[pp97--100]{rountree98}` will produce

> The section titled "Are You Bored Yet?" is particularly useless (Rountree, 1998, pp97–100).

However if you want to add text at the beginning of the citation, you should use a combination of the `\citetext` and `\citealp` commands, like this:

```
Several works have been cited in this document
\citetext{e.g., \citealp{rountree98,amir97}}.
```

producing:

> Several works have been cited in this document (e.g., Rountree, 1998; Amir *et al.*, 1997).

Citing your current thesis is not allowed. It is acceptable to refer to previous work of your own.

Chapter 3

A New Approach

In your thesis you will be expected to present some sort of new treatment of your subject. Since this may well involve the use of diagrams, I will take this opportunity to explain the use of "floating" figures and including graphics files.

3.1 Floats

Because LaTeX forces the writer to concentrate on content rather than formatting, you may never be sure where pages will break or where figures will be placed. LaTeX uses a system of "Floats" for figures and tables which ensure that they will be put in the most convenient place.

Figure 3.1 is an example of a floating figure. Note that even though the code for it is placed before this paragraph, the figure is appearing after this text; it has been "floated" into a more convenient place. The code for the figure looks like this:

```
\begin{figure}[htb]
\begin{center}
\includegraphics[height = 110mm]{graph1.pdf}
\caption[An example of a floating figure.]{An example of a floating
         figure.  Captions for diagrams generally go {\em underneath}
         the figure itself.}
\label{fig:graph1}
\end{center}
\end{figure}
```

Both width and height can be scaled using the optional part of the `\includegraphics` command. If you scale just one or the other, the aspect ratio will be maintained. If you try to scale both, you run the risk of your picture being "squished".

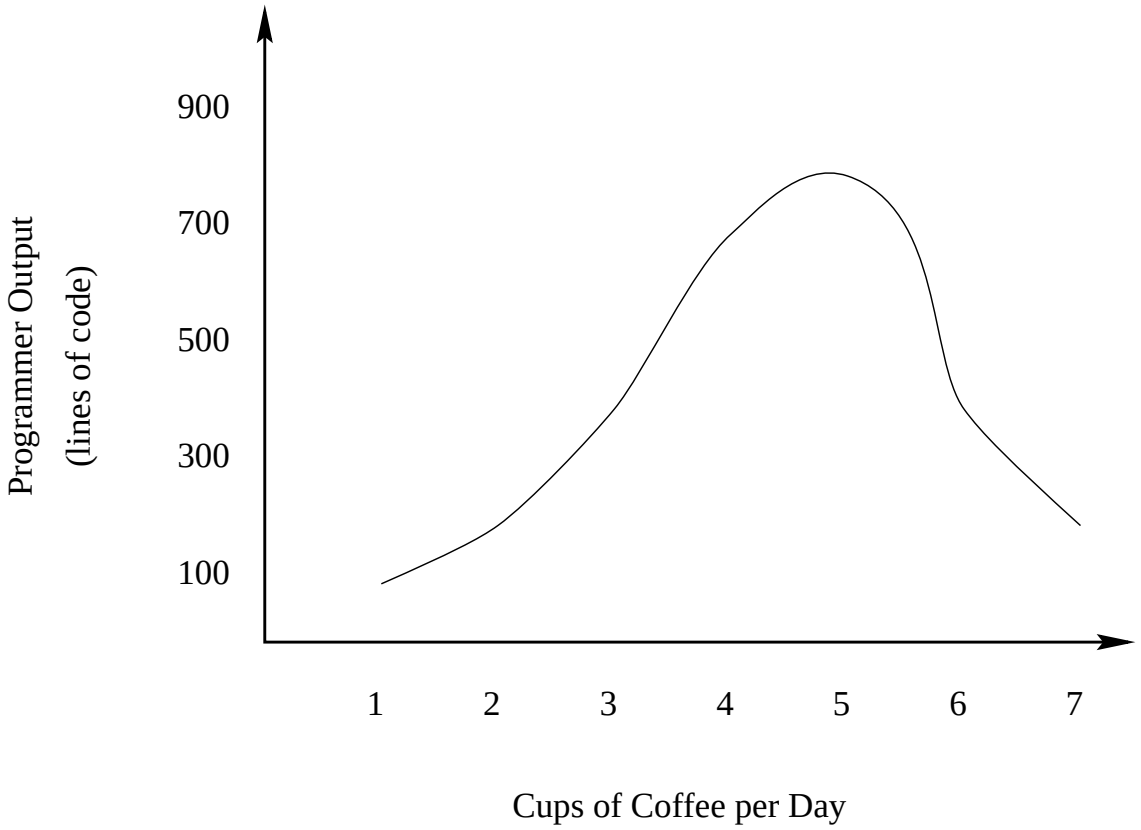


Figure 3.1: An example of a floating figure. Captions for diagrams generally go *underneath* the figure itself.

Under Pdflatex, you may use GIF, JPEG, TIFF, PNG, or PDF graphics. All will work the same way. Bear in mind that if you are drawing a *diagram*, it is best to use a vector format that will scale (e.g. as PDF), whereas photographs and screenshots may as well be in JPEG or PNG format.

The [htb] part gives LaTeX the options you want to use to try placing the figure. First, it will try to place the float at the exact position in the text. Next, it will try to place it at the top, then bottom of the current page. Finally, if [p] is also specified, it will try to place it on a page dedicated only to floats. The order of [htb] after `\begin{figure}` is irrelevant—it will always try to place figures in this order. The options just tell it *what* it is allowed to try, not the order it tries it.

The caption has two parts: the part inside the square brackets is what appears on your List of Figures page, and the part inside the braces is what appears under the figure itself.

The `\label` part gives you a label so you can reference your figure. The command

`\ref{fig:graph1}` will be replaced in your text by the actual number of the figure; e.g. `Figure~\ref{fig:graph1}` will come out as "Figure 3.1". For this number to be correct, the `\label` part *must* come *after* the `\caption` part. Also don't forget that if you have a figure with a caption you are required to refer to it somewhere in your text.

Here is the figure again; this time we will make LaTeX try *really hard* to put the figure right *here*.

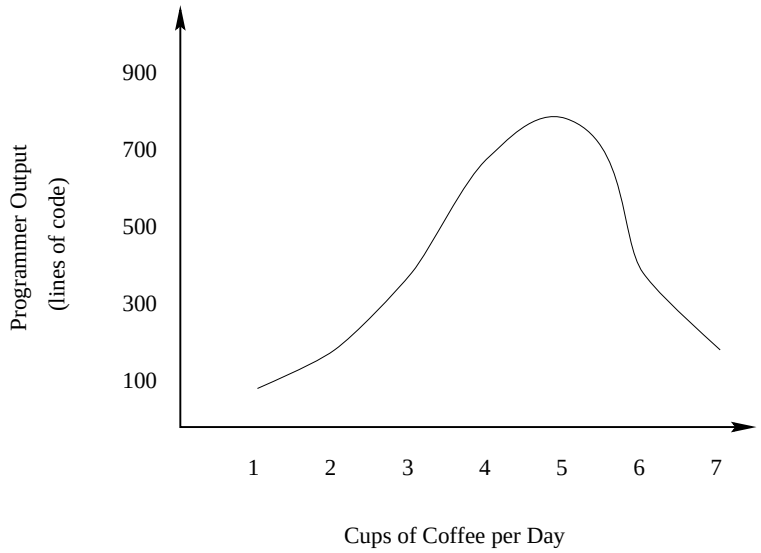


Figure 3.2: This figure has been forced into position by using the [!h] option.

Figure 3.2 has been placed right where it is included in the document, using the [!h] instead of [htb]. If [!h] is not strong enough, it can be made even stronger by using the [H] option.

You can also scale figures based on their original sizes; Figure 3.3 has been scaled by a factor of 50% using the option [scale = 0.5].

This is one of the most obvious reasons to use PDF graphics when creating diagrams—PDF vector diagrams will scale perfectly and print out crisply and clearly, with no "jaggies". Just about every graphics program in common use will output PDF graphics for you.

If you use GNU programs like `xfig`, `dia` or `gimp` to do your diagrams or screenshots, you will have no problem exporting files as PDF.

Figure 3.4 is an example of the point above. It is the contents page of this very document; first, the document was compiled by typing `make`. Next, the contents page

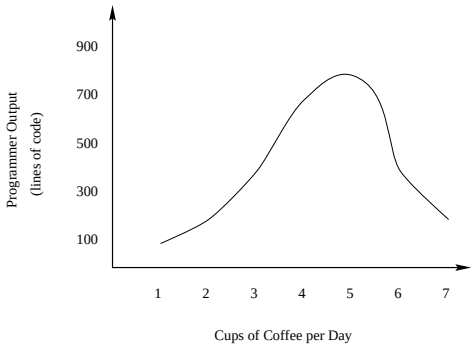


Figure 3.3: This figure has been been scaled from its original size.

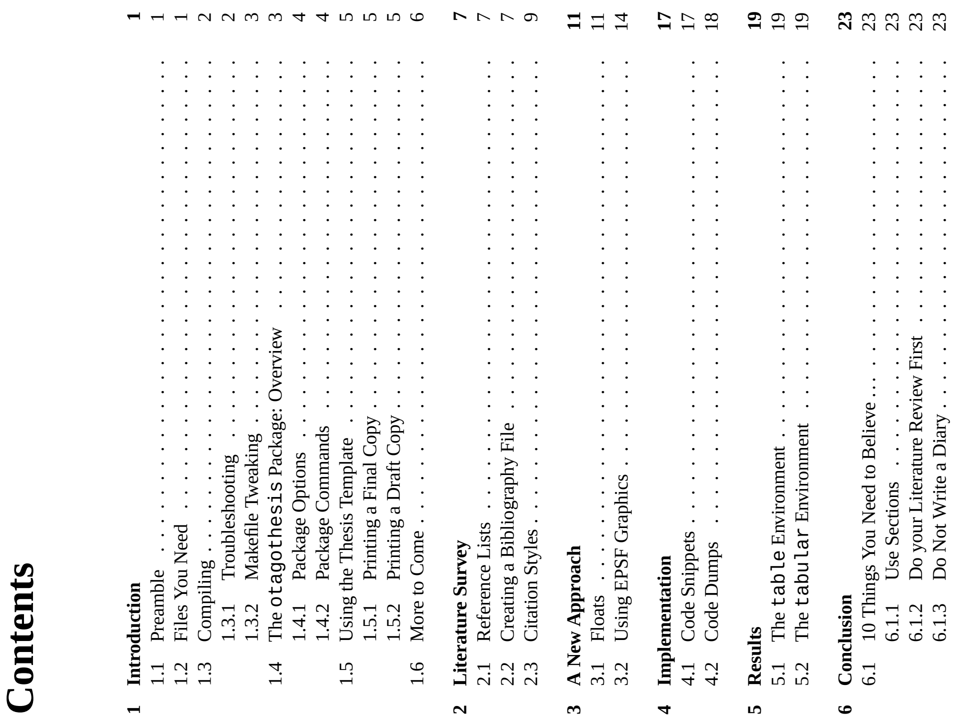
was viewed using `gv` and saved as a separate PDF file. It was rotated about 90° by using the [angle=90] option in the `\includegraphics` command.

The `otagothesis` Makefile has a feature especially for xfig users. If you uncomment the line that starts with the word `FIGFILES`, put your xfig figures in the same directory as your thesis, and put their filenames in place of `ex1.fig ex2.fig` etc., the Makefile will call `fig2dev` on them to convert them all to pdf format. As a result, you don't have to export your figures from xfig as pdf; just save them as regular `.fig` format in your thesis directory. NOTE: this requires installation of the `xfig/fig2dev` software on MacOsX.

A final word about figures, from Goosens, Mittelbach, and Samarin (1994, p141 ff):

> Floats are often problematical in the present version of LaTeX, since the system was developed at a time when the amount of graphical material in a document was considerably less than it is now ... If ... a lot of floating material is present ... then it is often the case that all material from a certain point onwards floats to the end of the document.

In fact, this can cause a real problem—not only is it ugly, but if too many floats remain unprocessed before the end of the document is reached, LaTeX will die with a "too many unprocessed floats" error. This can be fixed by periodically issuing the command `\clearpage`, which forces all as-yet unprocessed floats to be printed.



Contents

vii

Figure 3.4: This is the contents page of this document, included as a floating figure and scaled to 60% of its original size

Chapter 4

Implementation

In this chapter, a Computer Science student would be expected to explain the design-decisions made regarding the program they have written. Therefore, some way of including code snippets is required, as well as some way of printing out all of the code as an appendix.

4.1 Code Snippets

The best package for printing out code snippets is `moreverb`, by Angus Duggan. This lets you print out code listings, with numbered lines. For code snippets, I suggest you use the `listing` environment, like so:

```
\begin{listing}[1]{1}
  ... some code here ...
\end{listing}
```

The example above will produce a code listing with numbers every line, starting from line 1, like this:

```
1   #include <stdio.h>
2   int main(void) {
3     printf("hello world!\n");
4   }
```

I also prefer to issue the command `\linespread{1}\small` before presenting code snippets, to condense the example somewhat. If you do this, start a new paragraph just before the command, otherwise it will be applied to the paragraph above. Also, remember to go back to normal spacing after the code snippet by issuing the command `\linespread{1.3}\normalsize`.

4.2 Code Dumps

If you look at the file `appendices.tex`, you will see how the code for this document has been included as an appendix. The command `\listinginput` has been used—it works just like `\begin{listing}` except that it reads in a file.

For the purposes of reading your code into a document, I strongly suggest that you make a directory under your `thesis` directory called `src_links`. Then, make symbolic links to all your code in this directory. Next, use those links as the arguments to `\listinginput`. Now if you change some code in a file, those changes will be reflected in your code dump.

Chapter 5

Results

Chapters containing experimental results often contain a number of tables. These are "floated" in LaTeX, just like figures. The short caption is automatically added to the "List of Tables" page, and—like a figure—the table can contain almost anything (graphics, text, tabular material, code snippets or whatever).

The most important thing to establish is the difference between the `table` and `tabular` environments. A table is just a floating body, but the tabular environment is used to create *actual* tables. The following examples show what you can do with both environments.

5.1 The `table` Environment

Table 5.1 is an example of a table with graphical content. Note one convention: unlike figures, tables are usually captioned at the **top**, not the bottom.

As with figures, the best place to put the `\label` is directly after the `\caption`, otherwise you will end up with the section label instead. You can find the code for Table 5.1 on page 42, starting at line 106.

5.2 The `tabular` Environment

Most likely what you will want to put inside a `table` environment is a table of data. This is done using the `tabular` environment, as in the following example. This code:

```
\begin{tabular}{l|c|c}
Day & Cups of Coffee & Lines of Code \\
\hline
Mon & 3 & 200 \\
```

```
Tue & 5 & 500 \\
Wed & 5 & 300 \\
Thu & 4 & 200 \\
Fri & 3 & 100 \\
\end{tabular}
```

will produce this table:

Day	Cups of Coffee	Lines of Code
Mon	3	200
Tue	5	500
Wed	5	300
Thu	4	200
Fri	3	100

The vertical bars in the `{l|c|c}` argument tell LaTeX to put lines between each column. The argument `{|l|c|c|}` would have resulted in lines around the outside as well. The letters indicate the text formatting in the table (l for left, c for center, etc.). The command `\hline` is used to add horizontal lines (including at the top and bottom of the table) and `\\` is used to end each row. You do not need to use `\\` at the end of `\hline`. Finally, the `&` character is used to separate entries within rows.

Table 5.2 is a slightly more complex example of tabular material, included as a table in the document. Here is the code which produced it:

```
1   \begin{table}
2   \hrulefill
3   \caption[A tabular table]{An example of a table whose contents are
4   formatted using the {\tt tabular} environment.}
5   \label{tab:ex2}
6   \hrulefill
7   \begin{center}
8   \begin{tabular}{|l|l|r|} \hline\hline
9   {\em type} & \multicolumn{2}{c|}{\em style} \\ \hline
10  smart & red & short \\
11  rather silly & puce & tall \\ \hline\hline
12  \end{tabular}
13  \end{center}
14  \par
15  \bigskip
16  \hrulefill
17  \end{table}
```

The `\multicolumn` command on line 8 allows the spread of data over several columns, and overrides the normal vertical bar placement. Lines 2, 6 and 14–16 produce the horizontal lines which separate the table from the rest of the text.

Table 5.1: An example of a table which uses graphical input as its content. This is once again the contents page of this document, saved as PDF.

Contents

Table 5.2: An example of a table whose contents are formatted using the `tabular` environment.

type	*style*	
smart	red	short
rather silly	puce	tall

Chapter 6

Conclusion

Here are some final comments about putting together a thesis. They are mostly opinion (and certainly opinionated) so take on board what you will. This is free advice after all (and may be worth as much) but it is meant to make life easier for you.

6.1 10 Things You Need to Believe . . .

6.1.1 Use Sections

Chapters are BIG things. Write an opening paragraph to your chapter, then mark in each section you are going to cover. Fill in the actual content AFTER you have worked out just what the sections are going to be.

6.1.2 Do your Literature Review First

Lots of people do their literature review after they have written their program/done their experiment. Don't fall into this trap—you think the write-up will only take six weeks, but it won't, because the lit. review will take four. Create a BibTeX file while you do the review, so you can cite as you go. You would be amazed at how many people don't do this.

6.1.3 Do Not Write a Diary

Nobody is interested in how you went about writing your program. The academic community is interested in how you have integrated the existing theory with your own ideas, and the department is interested in why you made particular design decisions.

Remember that we are a Humanities department as well as a Science department, so we want to see some sort of convincing argument for your ideas.

6.1.4 Avoid Visual Formatting

Try not to use commands like `\vspace, \pagebreak` or `\enlargethispage`. The whole point of LaTeX is that it provides a markup language that works perfectly 90% of the time. This means that when you have **finished**, the last thing you should do is print up a draft, then go through and mark any formatting you don't like (there will probably be about one thing every ten pages). That is the time to insert things like `\pagebreak` commands.

6.1.5 Learn LaTeX Early

The learning curve for LaTeX is steep but short. The idea is that some poor sod like me does all the dirty work, and all you have to do is *fill in the content*. However you may spend about a week learning all the fiddly bits (like tables and figures), and you don't want to be taking time away from the actual thesis writing.

6.1.6 Read Some Documentation

In the Systems Lab, we are going to make every attempt to have printed documentation lying around. If you need quick help, check out the file `essential.dvi` on any Linux LaTeX installation. It is probably the best short guide to LaTeX around, and even contains lots of tricky maths examples.

6.1.7 Save Paper

On a mac, use the layout options to print two-up and double-sided. On a linux machine, use `psutils` to print your drafts. This way you use a quarter of the paper. Here's what to do if you don't have a duplex printer:

1. Use the twosided option and make the PDF file.
2. Use `pdftops` to convert `thesis.pdf` to `thesis.ps`.
3. Use `psbook` to arrange the pages for booklet printing;

   e.g. `psbook thesis.ps thesis.bk.ps`.

4. Use `psnup` to get the book to a booklet;

   e.g. `psnup -2 thesis.bk.ps thesis.2up.bk.ps`.

5. Use `psselect` to print up the even pages first (in reverse), then the odd pages;

   e.g.
   type `psselect -e -r thesis.2up.bk.ps | lpr`
   Put the result into the sheet feeder, blank side up
   type `psselect -o thesis.2up.bk.ps | lpr`

That's the theory; in practice it goes something like this:

`psbook thesis.ps | psnup -2 | psselect -e -r | lpr`
Now open the laserjet side-door and put the paper which came out after the first command on the tray. Don't change its orientation or anything: just pick it up, move it to the tray and drop it. Then:
`psbook thesis.ps | psnup -2 | psselect -o | lpr`

6.1.8 Use LaTeX, not Word

Microsoft Word is not your friend. Word is not *anybody's* friend. It is possible to write large documents in Word, but you have to be **so** strict on yourself (using heading styles, TOC entries, etc.) that you may as well have used LaTeX in the end anyway. Once a problem is nailed in LaTeX, it stays nailed. The same is not true of Word. Pdflatex produces standard PDF which can be printed anywhere, and is fast becoming a standard for on-line article publication. Ask anyone how many problems they have had printing Word documents to PostScript printers.

6.1.9 The First Copy is Not the Final Copy

Well, unless you're Isaac Asimov, or Mozart. Print up draft copies of chapters and give them to your supervisor to read, then implement any changes when they get returned. Then, a week later, reread the chapter and change anything that makes you cringe.

6.1.10 Spell Check your Work

If you are using UNIX (as I have assumed throughout this document) then the `ispell` program is a pretty good spell checker. Also get someone to check your punctuation and grammar, and be on the lookout for spellchecker errors (like "fro" instead of "for", which will not be picked up at all). Even the *worst* writer of novels who gets published has a good grasp of grammar, because it is unpleasant to read work which doesn't make sense. There is no point in putting the examiner in a bad mood by turning the reading of your thesis into an unpleasant chore.

6.2 General Comments

Traditionally, academic works are written using the passive voice—"this was done" rather than "we did this". Also, the use of the personal pronoun "I" is avoided in favour of "we"; but here you must be careful. There are two ways of using "we": either *in*clusively or *ex*clusively. Inclusively is pretty much alright as it makes the reader feel like part of the story; e.g. "Reducing Equation 2.3 to Equation 2.4, we can see ...". Exclusively sounds pompous—"we now present a new algorithm which will solve this problem ..." and should be avoided unless writing a paper with more than one author.

There are some other things worth remembering:

- When referring to another chapter or section, Use a capital letter; e.g. "see Chapter 3" not "see chapter 3". Use a tilde character (~) to put a non-breaking space before the number. That way you will never begin a new line with the number.

- The layout suggested in this document (introduction, lit. review, new ideas, implementation, results, conclusion) is pretty generic. If you stick to it, you will complete a thorough but boring thesis. You will almost certainly need to digress occasionally, and perhaps integrate the literature review more with your own work. While academic writing isn't usually noted for its racey prose, it doesn't have to be boring.

Finally, at the end of your thesis, don't be afraid to blow your own trumpet. If you have done something new and original, restate just what that is. The last part of the concluding chapter is what most people read straight after the abstract, so it has to be just as pithy and imagination-capturing.

References

Amir, A., Feldman, R., and Kashi, R. (1997). A New and Versatile Method for Association Generation. In J. Komorowsk and J. Zytkow (Eds.), *Principles of Data Mining and Knowledge Engineering; First European Symposium, PKDD'97*, Trondheim, Norway, 221–231. Springer-Verlag.

Goosens, M., Mittelbach, F., and Samarin, A. (1994). *The LaTeX Companion* (2nd ed.). Addison-Wesley.

Rountree, N. (1998). *How to do Anything.* Dunedin, New Zealand: Non-existent Publishers.

Appendix A

Source Code for thesis.dvi

A.1 thesis.tex

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% This is a document template for an Otago thesis (Masters, PhD, etc).
%% A skeleton chapter layout is also suggested.
%%
%% Look in the directory example_document for filled-out chapters
%% that show you how to do figures, bibliographies, and tables.
%%
%% Since this was written for Computer Science at Otago University,
%% Harvard (author, date) style citations are used.
%%
%% All Physics and EMAN students should consult their advisors as
%% to what style of citations to use!
%%
%% Nathan Rountree 9/2/98
%%
%% Edited for Physics use by Annika Seppala, Feb 2020
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%
%% In the style of a technical report, in 12pt and one sided.
%% Start chapters on right hand side pages only.
%%
\documentclass[12pt]{report}
%\documentclass[12pt,twoside,openright]{report} %% Use this for twosided.

%%
%% Load packages.
%%
\usepackage[bschons,nolibrary]{otagothesis}     %% Use Otago page layout
%% Use [bschons] for BScHons thesis
%%     [msc]     for MSc
%%     [phd]     for PhD
%%     [dipsci]  for PGDipSci
%%  nolibrary-option omits a library declaration form.
```

```
\usepackage[longnamesfirst,round]{natbib} %% Use Natural Sciences bibliography

\usepackage{graphicx}             %% jpg, gif, tiff, and pdf graphics
\usepackage{moreverb}             %% Verbatim Code Listings

% Standard Physics additions
\usepackage{amssymb,amsmath}
\usepackage{siunitx}
\usepackage[colorlinks=true,pdfstartview=FitV,linkcolor=blue,
            citecolor=blue,urlcolor=blue]{hyperref}

%%
%% Set title, author and date.
%%
\title{A Suggested Thesis or Dissertation Layout}
\author{Original document by Nathan Rountree (Department of Computer Science), amended for Phys
\date{9 February 1998}

%\title{Your thesis title here} % <-- Your thesis title here
%\author{Your name here} % <-- Add your name here
%\date{\today} % <-- Submission date here.
%% \date{\today} prints the date you compiled the document.
%% This changes automatically every time you run LaTeX!

%%
%% The library want to know all sorts of personal stuff!
%% Can be left out if you don't use the \frontstuff command
%%
%\fullname{Your full name here} % <-- Add your name here
%\department{Department of Physics}
%%\dob{1 January 1900} %% date-of-birth, only needed for library declaration
%\address{730 Cumberland Street, Dunedin, NZ}
\fullname{Nathan Rountree}
\department{Department of Computer Science}
\dob{1 January 1900} %% date-of-birth
\address{111 North Road, Dunedin, NZ}

%%
%% Uncomment to just print up a few chapters.
%%
%%\includeonly{literature,conclusion}

%%
%% Go!
%%
\begin{document}

%%
%% Put in titlepage and contents, etc...
%%
 \frontstuff
%% If you comment this out the pdf file will start from Chapter 1.
%% This is can be useful while you are still working on the text.

%%
```

```
%% Set to one-and-a-half line-spacing
%%
\linespread{1.2} \normalsize %% adjust this for final thesis printing!

%%
%% Include each chapter as a separate file.
%% These lines assume there are files called intro.tex, literature.tex etc.
%%
\include{intro}

\include{literature}

\include{new_ideas}

\include{implementation}

\include{results}

\include{conclusion}

%%
%% Make certain the ``references'' section begins on a recto page when
%% document is double-sided.
%% The ``bibliography'' line assumes that there is a file called
%% ``thesis.bib'' and that somewhere in the chapter material you have
%% cited something from it.
%%
\cleardoublepage
\bibliographystyle{otago}
\bibliography{thesis}

\include{appendices}

\end{document}
%% All Done!
```

A.1.1 abstract.tex

```
An abstract should be somewhere between 50 and
200 words long---the
absolute maximum at Otago is 500 words, but that is almost certainly
too long.  Remember, you will probably get at least one article out of
your thesis and it would be nice to be able to recycle the abstract
without having to snip it.

All you need to do in the abstract is give a statement of the problem,
a {\em brief} explanation of the method and procedures used, and a
summary of conclusions.  You do {\bf not} have to give away your
conclusions entirely; save that for the concluding chapter.  Bear in
mind that the abstract is all that most people will bother to read, so
it had better be good!
```

A.1.2 acknowledgements.tex

```
This is the {\em acknowledgements} environment, defined in
```

```
{\tt otagothesis.sty}.  If you have any preface material, this is where to
put it.

I would like to thank the following people:
  \begin{itemize}
  \item The Department of Computer Science, for employing me over the
  Summer break in 1997/8;
  \item My colleagues in the Lab, who {\em still} haven't
  complained about the mess;
  \item Everyone who puts up with me.
  \end{itemize}
Thank you all.
```

A.1.3 intro.tex

```
\chapter{Introduction}

\section{Preamble}

The {\tt otagothesis} package consists a style file and document
template for writing theses and dissertations at the University of
Otago.  It was intended only for computer science students, but seems
to have found wider popularity.  Its purpose is to take the hard work
of constructing a large document out of your hands.

Earlier versions of this package used Latex to produce a DVI file
which could be converted to PostScript and printed.  There are several
advantages to using Pdflatex instead: the document goes straight to
PDF, and may include graphics files of many types instead of only
EPSF.  This package now uses Pdflatex instead of Latex.  If you wish
to use it in place of an earlier version, you may need to convert any
EPSF diagrams to PDF with the \verb|epstopdf| program, and you should
remove the \verb|[dvips]| option from the \verb|\usepackage{graphicx}|
command.

This document is a quick tour of the {\tt otagothesis} package and a
very brief comment on how to organise a thesis in Computer Science.
For an even more brief introduction to the package, read the README
file.  Please bear in mind that this is not a style guide, nor an
example of good content---although you will find some ideas on how
to cite other authors' work, and how to include tables and figures in
your document.  These are all specific to the Systems Research Lab at
the Computer Science Department; all other students should seek advice
from their supervisors before following {\em any} of the advice
contained herein!

It should be very easy to convert the file {\tt thesis.tex} to
your own use.
The {\tt otagothesis.sty} file is not quite as flexible, but feel
free to hack around with it if you wish---just be sure to pass on
the original to others (or take my name off the changed version).
The idea is that if you are an Otago University student
you shouldn't need to change {\tt otagothesis.sty} at all.

\section{Files You Need}
```

```
\label{sec:files}
There are several files you need to have a copy of before you will be
able to compile this document for yourself. These files are included in the original set of fil

\begin{itemize}
\item {\tt otagothesis.sty}: this is the file which provides the
{\tt otagothesis} package.  It contains some useful commands and
options, and formats your pages according to University library restrictions.
\item {\tt logo.pdf}: the Otago University crest, required for your
library declaration page.
\item {\tt natbib.sty}: this is the file which provides Harvard style
(author, date) citations.  It is provided in TeTeX, the distribution
of LaTeX that comes with Redhat Linux.
\item {\tt plainnat.bst}: this provides Author-date style bibliographies.
It is also standard in TeTeX.
\end{itemize}

The easiest way to use the otagothesis package is to put {\tt
otagothesis.sty} and {\tt logo.pdf} in the same directory as your
document.  At its simplest, put \verb|\usepackage{otagothesis}|
just after the \verb|\documentclass{report}| declaration, and you're
away.  For other usage options, follow the pattern in the
\verb|thesis.tex| file.

\section{Compiling}
Just to see if everything is working, try compiling this document by
unzipping the distribution, changing into the \verb|example_document|
directory, and
typing 'make'.  You should see a whole lot of messages printed to the
screen, followed by a preview window showing you a title-page.  If you
don't get this, try reading Subsection~\ref{subsec:trouble} on
troubleshooting the distribution.  If everything seems to be OK, go on
to Section~\ref{sec:overview} which tells you a bit about using these files.

If you don't like previewing your document every time you recompile
it, read Subsection~\ref{subsec:makefile} which has some suggestions
for changing the makefile.

\subsection{Troubleshooting}
\label{subsec:trouble}
Here are some of the things that might go wrong on the initial
compilation and how to fix them.
\begin{itemize}
\item {\tt LaTeX Error: File 'otagothesis.sty' not found}: \\
The simplest solution is to put the relevant
file(s) into the same directory as your document.  The files
\verb|otagothesis.sty| and \verb|logo.pdf| are in this distribution,
and \verb|natbib.sty|, \verb|plainnat.bst|, and \verb|moreverb.sty|
are all available on \verb|http://www.ctan.org|.
This can also happen if your latex distribution doesn't have
the \verb|natbib| package installed, in which case it will complain
about not having the files \verb|plainnat.bst| and \verb|natbib.sty|.
Other files that could be missing are \verb|logo.pdf| or
\verb|moreverb.sty|.
```

```
\item The makefile doesn't work: \\
This makefile was written for GNUmake, and isn't guaranteed for any
other version of make.  If you need to compile without make, type the
following:
\begin{verbatim}
    pdflatex thesis
    bibtex thesis
    pdflatex thesis
    pdflatex thesis
\end{verbatim}
(also known as the \LaTeX\ stanza).  The extra {\tt pdflatex} at the end
is to get all the cross-references right.

\item I printed it up, and it's all off to one side!\\
It's supposed to be.  The bindery wants you to have a big binding
margin on the left of recto pages and on the right of verso pages.
If you are printing up using the one-sided option, then all of the
pages should have a wider margin on the left.
\
\item Any other problems are caused by one of two things: either you
have jumped the gun and edited a file, or your installation of \LaTeX\
doesn't look like mine.  You need the standard \LaTeXe\ packages for
it all to work; in particular the {\tt moreverb}, {\tt graphicx}, and
{\tt natbib} packages are absolutely crucial.

\item Many people in the department use \LaTeX , ask around for help! Internet is also great fo

\end{itemize}

\subsection{The Makefile}
\label{subsec:makefile}
The makefile for this package is called (appropriately enough)
``Makefile''.  You will find detailed instructions for its use in the
header comments.  A makefile tries to figure out what needs to be done
every time you compile your thesis, depending on what was last
changed.  It can save you a lot of typing.

Briefly: typing {\tt make} will create a pdf file viewable with
Acrobat reader, MacOsX's Preview, or Linux's gv.
Typing {\tt make clean} will remove all auxilliary
files, and {\tt make touch} will prepare for a complete re-compile.
If you want to suppress automatic viewing, find the \verb|AUTOVIEW|
variable in the makefile and set it to \verb|no|.

\section{The {\tt otagothesis} Package: Overview}
\label{sec:overview}

The {\tt otagothesis} package has been designed using the guidelines in
the booklet ``Notes on the Preparation of Theses'', available from the
reference desk at the Central Library.  I strongly suggest that you
obtain a copy of this and read it.

Open up the {\tt thesis.tex} file to see how the {\tt otagothesis}
package has been invoked and what sorts of options it gives you.  There
are lots of comments to explain what each part is for.
```

```
The package has the following features:
\begin{itemize}
\item A 30mm binding margin.  If you are printing up single sided,
this will be on the left of every page.  If you are printing up double
sided, it will be on the {\em left} of {\em recto} (right-hand-side)
pages, and on the {\em right} of {\em verso} (left-hand-side) pages;
\item Automatic title-page and contents page generation;
\item Automatic insertion of abstract and acknowledgements files;
\item A wide margins option for proofreading purposes (so your
supervisor has room for the red pen!);
\item Inclusion of the bibliography in the table of contents, renamed
``References''.
\end{itemize}
The following subsection lists all of the package's options in detail.

\subsection{Package Options}
To use the package, all you need to do is include the line
\begin{quote}
\verb|\usepackage[option1,option2, ... ]{otagothesis}|
\end{quote}
in your main document.  With no options at all, this will give you
the commands available in the package (see the next subsection) and
will format your document with the correct margins.  It assumes you
are doing a PhD.
The available options are:
\begin{description}
\item[{\tt nofigures}] Suppresses the production of a ``list of figures''
page.
\item[{\tt notables}] Suppresses the production of a ``list of
tables'' page.
\item[\tt nolibrary] Suppresses the production of a library
  declaration page.
\item[{\em thesistype}] Enter a thesistype to produce the correct
title-page.  Available thesistypes are: {\tt phd, msc, ma, mcom,
interimsci, interimarts, interimcom, dipsci, diparts, dipcom, bschons,
bahons, bcomhons}.  The package will default to {\tt phd}.
\end{description}

\subsection{Package Commands}
There is only one command provided with the package, and it is
designed to help you print draft copies.  The command is
\begin{quote}
\verb|\frontstuff|
\end{quote}
and it places all the preface material before
Chapter~1.  If you don't issue this command in the document, you won't
get a title-page, table of contents, etc.---your thesis will just
print from Chapter~1.

% Physics
\section{Other packages included in the template}
Physics department has added some other useful package options.

\subsection{{\tt amssymb } and {\tt amsmath}}
```

```
These are the essential packages for writing math and physics formulas.
For a short guide to usage in your latex document see
\url{http://mirror.aut.ac.nz/CTAN/info/short-math-guide/short-math-guide.pdf}

For example (this is an example based on the above document):
\begin{quote}
\begin{verbatim}
\begin{equation}\label{first}
\alpha=b+c
\end{equation}
some intervening text
\begin{subequations}\label{grp}
\begin{align}
\alpha&=b+c\label{second}\\
d&=e+f+g\label{third}\\
h&=i+j\label{fourth}
\end{align}
\end{subequations}
\end{verbatim}
\end{quote}

Produces the following:

\begin{equation}\label{first}
\alpha=b+c
\end{equation}
some intervening text
\begin{subequations}\label{grp}
\begin{align}
\alpha &=b+c\label{second}\\
d&=e+f+g\label{third}\\
h&=i+j\label{fourth}
\end{align}
\end{subequations}


\subsection{{\tt siunitx}}
A great package for typing units in and out of equation mode!
A comprehensive guide is available at
\url{http://tug.ctan.org/macros/latex/exptl/siunitx/siunitx.pdf}.
%
For example
\begin{quote}
\begin{verbatim}
\si{kg.m.s^{-1}} \\
\si{\kilogram\metre\per\second} \\
\si[per-mode=symbol]{\kilogram\metre\per\second} \\
\si[per-mode=symbol]{\kilogram\metre\per\ampere\per\second}
\end{verbatim}
\end{quote}
Produces:\\
%
\si{kg.m.s^{-1}} \\ \si{\kilogram\metre\per\second} \\
\si[per-mode=symbol]{\kilogram\metre\per\second} \\
\si[per-mode=symbol]{\kilogram\metre\per\ampere\per\second}
```

```
\verb|siunitx| is particularly useful when typing unit within an equations:
The standard equation mode will italicise letters:
\[ 1m + 1m = 2m\]
\verb|siunitx| prevents that:
\begin{quote}
\begin{verbatim}
\[ 1\si{.m} + 1\si{.m} = 2\si{.m} \]
\end{verbatim}
\end{quote}

\[ 1\si{.m} + 1\si{.m} = 2\si{.m}\]



\subsection{{\tt hyperref}}
The {\tt hyperref} package generates clickable links within the compiled pdf file.
Anytime you reference literature using \verb| \citet{} ,\citep{} | etc.
or reference a Chapter, Figure or Table using \verb|\ref{} |
\url{http://otago.ac.nz}



\section{Using the Thesis Template}
The main aim of this package is to allow you to write completely
vanilla \LaTeX\ and have everything come out in the right place.  All
you need to do is know how to create each chapter file, how to include
them in the main document, and how to print up a draft as well as the
main copy.

So go ahead and write chapters which have \LaTeX\ formatting commands
in them, and \verb|\include| them into the {\tt thesis.tex} file.
If you are using \verb|make|, add the name of the chapter file to the
makefile (including the \verb|.tex| suffix). Type {\tt make} and
bingo: one perfectly formatted thesis!

\subsection{Printing a Final Copy}
Make sure you have issued the command \verb|\frontstuff| directly
after \verb|\begin{document}|.  You should also comment out the line
beginning with \verb|\includeonly|, which restricts which chapters
you are going to print up.  If you want to include an abstract and
acknowledgements, you should have files {\tt abstract.tex} and {\tt
acknowledgements.tex} in the same directory as {\tt thesis.tex} and
they will be included automatically.

Now all you have to do is type {\tt make} to preview your document.
If you are using the {\tt twosided,openright} option,
there will be quite a few blank pages because new chapters will always
start on a recto (right-hand-side) page.

The final document is in PDF format, so you can print it either
through a pdf viewer such as Acrobat, or by using the \verb|pdftops|
```

```
program to produce a postscript version that you can send to the
printer with \verb|lpr|.

\subsection{Printing Specific Chapters}
Let's suppose that you have 4 chapters done and are working on
Chapter~5.  Compile time is getting pretty long; it would be nice if
you could generate only Chapter~5, but retain correct pagination and
references.  Here's how:
\begin{enumerate}
\item Comment out the \verb|\includeonly| line and type {\tt make} to
create a complete set of {\tt .aux} files.
\item Uncomment the \verb|\includeonly| line and enter the names of
the chapter{s} you wish to print up (without the {\tt .tex} extensions).
\item Type {\tt make} to preview the document.  Only the
desired chapters and their references should be printed up, but the
page numbers and cross-references should all be correct.
\end{enumerate}

\section{More to Come}
In Chapter~\ref{chap:bib} we will look at how to use the Harvard style
citations and references provided by the {\tt natbib} package.
Then, in Chapter~\ref{chap:diagrams}, we discuss how to include
figures in your document.
Chapter~\ref{chap:code} has some examples of how to include code
snippets in the body of your text, and how to include code in the
appendices. Next, in Chapter~\ref{chap:tables}, some examples of
\LaTeX\ tables are presented.  Finally, Chapter~\ref{chap:final}
gives some dos and don'ts regarding style and content.
```

A.1.4 literature.tex

```
\chapter{Literature Survey}
\label{chap:bib}

Since this chapter would normally contain your literature review and
background sections, it seems like a good time to discuss citations
and bibliographies.  The file {\tt thesis.tex} is set to use Harvard
style citations, using the \verb|natbib| package.
The following sections describe how to use this package effectively.

\section{Reference Lists}

The {\tt otagothesis} package automatically renames your selected
bibliography to ``References'', since in Computer Science we only list
works that are actually referred to in the text.  We use the Harvard
style of citations as implemented by the \verb|natbib| package, which
is part of the TeTeX distribution of LaTeX (i.e.\ the one found on
most versions of Linux).

To get all references and citations correct, you need to perform the
\LaTeX\ stanza (see Subsection~\ref{subsec:trouble}): {\tt pdflatex,
bibtex, pdflatex, pdflatex}.  The makefile does this for you when you type
{\tt make}.

Every time you use a \verb|\citet| or \verb|\citep| command, a
```

```
bibliographic entry
will be added to your reference list.  On the whole, BibTeX will get
the styles correct, but you should definitely check this before
submitting a final copy; BibTeX bibliographies are notoriously
difficult to spell-check and some strange things can happen with
capitalisation.

\section{Creating a Bibliography File}

If you look at the file {\tt thesis.bib} you will find some
instructions on how to create a BibTeX database, and some sample
entries.  The makefile and thesis template are set up so as to
{\em expect} to see a file called {\tt thesis.bib} (your bibliographic
database file) and will generate an error message if there isn't one.
You can find a large number of Computer Science citations in
BibTeX format on the Collection of Computer Science Bibliographies
Homepage, at {\tt http://liinwww.ira.uka.de/bibliography/}.

A bibliographic entry looks something like this:

\linespread{1} \small
\begin{quote}
\begin{verbatim}
@InProceedings{amir97,
  author =       {Amihood Amir and Ronen Feldman and Reuven Kashi},
  title =        {{A New and Versatile Method for Association Generation}},
  booktitle =    {{Principles of Data Mining and Knowledge Engineering;
                   First European Symposium, PKDD'97}},
  OPTcrossref =  {},
  OPTkey =       {},
  editor =       {Jan Komorowsk and Jan Zytkow},
  OPTvolume =    {},
  OPTnumber =    {},
  OPTseries =    {},
  year =         {1997},
  OPTorganization = {},
  publisher = {Springer-Verlag},
  address =      {Trondheim, Norway},
  month =        {June},
  pages =        {221--231},
  OPTnote =      {},
  OPTannote =    {}
}
\end{verbatim}
\end{quote}

\linespread{1.3} \normalsize
Separate authors with the word ``and''.  The title is in double
braces to maintain the capitalisation exactly as written, as is the
title of the conference proceedings.  The very first part of the entry
(amir97) must be a unique key---this is what you will use when
referencing the work in your document, so make it something easy to
remember.  Most people use something like ``first\_author\_year'' with
a letter tacked on the end to avoid duplicates (e.g. amir97a, amir97b).
```

```
Here are some suggestions for creating your own database:
\begin{itemize}
\item Leave optional fields in place, even when they are empty.  You
never know when that information might turn up, so all you have to do
is add it into the correct field and remove the OPT prefix.
\item Use double braces to maintain your own capitalisation.
You can always remove them later using a search-and-replace, but it is
a real pain to go through the database putting them in.
\item Use the emacs macro to create each new entry (M-x bib-TAB-TAB
for a list of options).  You will be thankful for the consistency as
the list grows.
\item Use ``@InProceedings'' for an article in conference proceedings, not
``@Proceedings''.  Use ``@InCollection'' for an article which forms a
stand-alone chapter in a book where each chapter is written by a
different author, not ``@InBook''.
\end{itemize}


\section{Citation Styles}

The most usual form of citation is an ``aside'' (in parentheses, like
this). The time to use it is when you have made a statement which
might be questionable, and you wish to give a source for it. For
instance:
\begin{quote}
Writing a thesis is easy \citep{rountree98}.
\end{quote}
This type of citation is made using the \verb|\citep| command.  It
should be used sparingly, because it lends itself most to sweeping,
general statements (of which you don't want too many!).

The {\tt natbib} package gives you another option with
the \verb|\citet| command.  This enables you to refer directly
to a piece of work, using it as a noun in your sentence.  For
instance:
\begin{quote}
The article by \citet{amir97} describes a method of preprocessing
a database into a trie, so as to make association generation much quicker.
\end{quote}
This style is more versatile, and should be favoured over the use of
\verb|\citep|.

When there are multiple authors, all subsequent citations of that work
will just list the first author followed by ``et al.''.  This will happen\
automatically; for instance:
\begin{quote}
Here is a second citation of \citet{amir97}.
\end{quote}
If for some reason you want the whole list of authors, you can use the
``starred'' form of the command, i.e. \verb|\citep*| or
\verb|\citet*|.

Sometimes, it will be inelegant to include the year with the
citation; for example when you are referring to a particular article
very frequently in a single section, (perhaps when a chapter forms a
```

```
criticism of another piece of work).  For those purposes, use the command
\verb|\citeauthor|.  This will allow a sentence like ``the
book is of no use to anyone, but \citeauthor{rountree98} published it
anyway.''

The command \verb|\citeyear| is similar, but gives just the year,
allowing you to occasionally break style if you wish---and
\verb|\citeyearpar| will give the year with parentheses.  This will
let you write a sentence like ``\citeauthor{amir97}'s article, published
in \citeyear{amir97}, is the most up-to-date treatment of this topic.''

If you wish to add text to the citation (for instance the letters
e.g., or some page numbers), you may do so either at the beginning or
the end.  For the end (as with page numbers), simply add an optional
argument; \verb|\citep[pp97--100]{rountree98}| will produce
\begin{quote}
The section titled ``Are You Bored Yet?'' is particularly useless
\citep[pp97--100]{rountree98}.
\end{quote}
However if you want to add text at the beginning of the citation, you
should use a combination of the \verb|\citetext| and \verb|\citealp|
commands, like this:
\begin{verbatim}
Several works have been cited in this document
\citetext{e.g., \citealp{rountree98,amir97}}.
\end{verbatim}
producing:
\begin{quote}
Several works have been cited in this document
\citetext{e.g., \citealp{rountree98,amir97}}.
\end{quote}

Citing your current thesis is not allowed.  It is acceptable to refer to
previous work of your own.
```

A.1.5 new_ideas.tex

```
\chapter{A New Approach}
\label{chap:diagrams}

In your thesis you will be expected to present some sort of new
treatment of your subject.  Since this may well involve the use
of diagrams, I will take this opportunity to explain the use of
``floating'' figures and including graphics files.

\section{Floats}
Because \LaTeX\ forces the writer to concentrate on content rather
than formatting, you may never be sure where pages will break or where
figures will be placed.  \LaTeX\ uses a system of ``Floats'' for
figures and tables which ensure that they will be put in the most
convenient place.

\begin{figure}[htb]
\begin{center}
\includegraphics[height = 110mm]{graph1.pdf}
```

```
\caption[An example of a floating figure.]{An example of a floating
         figure.  Captions for diagrams generally go {\em underneath} the
         figure itself.}
\label{fig:graph1}
\end{center}
\end{figure}

Figure~\ref{fig:graph1} is an example of a floating figure.  Note that
even though the code for it is placed before this paragraph, the
figure is appearing after this text; it has been ``floated'' into a
more convenient place. The code for the figure looks like this: \par
\linespread{1} \small
\begin{quote}
\begin{verbatim}
\begin{figure}[htb]
\begin{center}
\includegraphics[height = 110mm]{graph1.pdf}
\caption[An example of a floating figure.]{An example of a floating
         figure.  Captions for diagrams generally go {\em underneath}
         the figure itself.}
\label{fig:graph1}
\end{center}
\end{figure}
\end{verbatim}
\end{quote}
\linespread{1.3} \normalsize

Both width and height can be scaled using the optional part of the
\small
\verb|\includegraphics| \normalsize
command.  If you scale just one or the
other, the aspect ratio will be maintained.  If you try to scale
both, you run the risk of your picture being ``squished''.

Under Pdflatex, you may use GIF, JPEG, TIFF, PNG, or PDF graphics.  All
will work the same way.  Bear in mind that if you are drawing a {\em
diagram}, it is best to use a vector format that will scale (e.g.\
as PDF), whereas photographs and screenshots may as well be in JPEG or
PNG format.

The [htb] part gives \LaTeX\ the options you want to use to try placing
the figure.  First, it will try to place the float at the exact
position in the text.  Next, it will try to place it at the top, then
bottom of the current page.  Finally, if [p] is also specified,
it will try to place it on a page dedicated only to floats.  The order
of [htb] after \verb|\begin{figure}| is irrelevant---it will always
try to place figures in this order.  The options just tell it
{\em what} it is allowed to try, not the order it tries it.

The caption has two parts: the part inside the square brackets  is
what appears on your List of Figures page, and the part inside the
braces is what appears under the figure itself.

The \verb|\label| part  gives you a label so you can reference your
figure. The command \verb|\ref{fig:graph1}| will be replaced in your
```

```
text by the actual number of the figure;
e.g. \verb|Figure~\ref{fig:graph1}| will come out as ``Figure 3.1''.
For this number to be correct, the \verb|\label| part {\em must} come
{\em after} the \verb|\caption| part.  Also don't forget that if you
have a figure with a caption you are required to refer to it somewhere
in your text.

Here is the figure again;
this time we will make \LaTeX\ try {\em really hard} to put the figure
right {\em here}.

\begin{figure}[!h]
\begin{center}
\includegraphics[width = 100mm]{graph1.pdf}
\caption[Another example of a floating figure.]{This figure has been
forced into position by using the [!h] option.}
\label{fig:graph2}
\end{center}
\end{figure}

Figure~\ref{fig:graph2} has been placed right where it is included in
the document, using the [!h] instead of [htb].  If [!h] is not strong
enough, it can be made even stronger by using the [H] option.

You can also scale figures based on their original sizes;
Figure~\ref{fig:graph3} has been scaled by a factor of 50\% using the
option [scale = 0.5].

\begin{figure}[!h]
\begin{center}
\includegraphics[scale = 0.5]{graph1.pdf}
\caption[A scaled example of a floating figure.]{This figure has been
been scaled from its original size.}
\label{fig:graph3}
\end{center}
\end{figure}

This is one of the most obvious reasons to use PDF
graphics when creating diagrams---PDF vector diagrams will scale
perfectly and print out crisply and
clearly, with no ``jaggies''.  Just about every graphics program in
common use will output PDF graphics for you.

If you use GNU programs like {\tt xfig}, {\tt dia} or {\tt gimp} to do
your diagrams or screen-shots, you will have no problem exporting files as
PDF.

Figure~\ref{fig:page} is an example of the point above.  It is
the contents page of this very document; first, the document was
compiled by typing {\tt make}.  Next, the contents page was viewed
using {\tt gv} and saved as a separate PDF file.  It was
rotated about \(90^\circ\) by using the [angle=90] option in the
\verb|\includegraphics| command.

\begin{figure}[tb]
```

```
\begin{center}
\includegraphics[scale = 0.6, angle = 90]{page.pdf}
\caption[A PDF page included as a figure.]{This is the contents
page of this document, included as a floating figure and scaled to
60\% of its original size}
\label{fig:page}
\end{center}
\end{figure}

The \verb|otagothesis| Makefile has a feature especially for xfig
users.  If you uncomment the line that starts with the word
\verb|FIGFILES|, put your xfig figures in the same directory as
your thesis, and put their filenames in place of
\verb|ex1.fig ex2.fig| etc., the Makefile will call \verb|fig2dev| on
them to convert them all to pdf format.  As a result, you don't have
to export your figures from xfig as pdf; just save them as regular
\verb|.fig| format in your thesis directory.  NOTE: this requires
installation of the \verb|xfig/fig2dev| software on MacOsX.

A final word about figures, from \citet[p141 ff]{goosens94}:
\begin{quotation}
Floats are often problematical in the present version of \LaTeX,
since the system was developed at a time when the amount of graphical
material in a document was considerably less than it is now \ldots\
If \ldots\ a lot of floating material is present \ldots\ then it is often
the case that all material from a certain point onwards floats to the
end of the document.
\end{quotation}
In fact, this can cause a real problem---not only is it ugly, but if
too many floats remain unprocessed before the end of the document is
reached, \LaTeX\ will die with a ``too many unprocessed floats'' error.
This can be fixed by periodically issuing the command
\verb|\clearpage|, which forces all as-yet unprocessed floats to be
printed.
```

A.1.6 implementation.tex

```
\chapter{Implementation}
\label{chap:code}

In this chapter, a Computer Science student would be expected to
explain the design-decisions made regarding the program they have
written.  Therefore, some way of including code snippets is required,
as well as some way of printing out all of the code as an appendix.

\section{Code Snippets}
The best package for printing out code snippets is {\tt moreverb}, by
Angus Duggan.  This lets you print out code listings, with
numbered lines.  For code snippets, I suggest you use the {\tt
listing} environment, like so:

\linespread{1} \small
\begin{quote}
\begin{verbatim}
\begin{listing}[1]{1}
```

```
  ... some code here ...
\end{listing}
\end{verbatim}
\end{quote}

\linespread{1.3} \normalsize
The example above will produce a code listing with numbers every line,
starting from line 1, like this:

\linespread{1} \small
\begin{quote}
\begin{listing}[1]{1}
#include <stdio.h>
int main(void) {
  printf("hello world!\n");
}
\end{listing}
\end{quote}
\linespread{1.3} \normalsize

I also prefer to issue the command \verb|\linespread{1}\small| before
presenting code snippets, to condense the example somewhat.  If you do
this, start a new paragraph just before the command, otherwise it
will be applied to the paragraph above.  Also, remember to go back to
normal spacing after the code snippet by issuing the command\\
\verb|\linespread{1.3}\normalsize|.

\section{Code Dumps}
If you look at the file {\tt appendices.tex}, you will see how the
code for this document has been included as an appendix.  The
command \verb|\listinginput| has been used---it works just like
\verb|\begin{listing}| except that it reads in a file.

For the purposes of reading your code into a document, I strongly
suggest that you make a directory under your {\tt thesis} directory
called {\tt src\_links}.  Then, make symbolic links to all your code
in this directory.  Next, use those links as the arguments to
\verb|\listinginput|. Now if you change some code in a file, those
changes will be reflected in your code dump.
```

A.1.7 results.tex

```
\chapter{Results}
\label{chap:tables}

Chapters containing experimental results often contain a number of
tables.  These are ``floated'' in \LaTeX, just like figures.  The
short caption is automatically added to the ``List of Tables'' page,
and---like a figure---the table can contain almost anything
(graphics, text, tabular material, code snippets or whatever).

The most important thing to establish is the difference between
the {\tt table} and {\tt tabular} environments.  A table is just a
floating body, but the tabular environment is used to create
{\em actual} tables.  The following examples show what you can do with
```

```
both environments.

\section{The {\tt table} Environment}

Table~\ref{tab:ex1} is an example of a table with graphical content.
Note one convention:  unlike figures, tables are usually captioned at
the {\bf top}, not the bottom.

As with figures, the best place to put the \verb|\label| is directly
after the \verb|\caption|, otherwise you will end up with the section
label instead.  You can find the code for Table ~\ref{tab:ex1} on page~42,
starting at line~106.

\section{The {\tt tabular} Environment}

Most likely what you will want to put inside a {\tt table} environment
is a table of data.  This is done using the {\tt tabular} environment,
as in the following example.  This code:

\linespread{1}\small
\begin{quote}
\begin{verbatim}
\begin{tabular}{l|c|c}
Day & Cups of Coffee & Lines of Code \\
\hline
Mon & 3 & 200 \\
Tue & 5 & 500 \\
Wed & 5 & 300 \\
Thu & 4 & 200 \\
Fri & 3 & 100 \\
\end{tabular}
\end{verbatim}
\end{quote}
\linespread{1.3}\normalsize

will produce this table:

\begin{center}
\begin{tabular}{l|c|c}
Day & Cups of Coffee & Lines of Code \\
\hline
Mon & 3 & 200 \\
Tue & 5 & 500 \\
Wed & 5 & 300 \\
Thu & 4 & 200 \\
Fri & 3 & 100 \\
\end{tabular}
\end{center}

The vertical bars in the \verb+{l|c|c}+ argument tell \LaTeX\ to put
lines between each column. The argument \verb+{|l|c|c|}+ would have
resulted in lines around the outside as well. The letters indicate the
text formatting in the table (l for left, c for center, etc.).  The
command \verb|\hline| is used to add horizontal lines (including at
the top and bottom of the table) and \verb|\\| is used to end each
```

```
row.  You do not need to use \verb|\\| at the end of \verb|\hline|.
Finally, the {\tt \&} character is used to separate entries within
rows.

Table~\ref{tab:ex2} is a slightly more complex example of tabular
material, included as a table in the document.  Here is the code which
produced it:

\linespread{1}\small
\begin{quote}
\begin{listing}{1}
\begin{table}
\hrulefill
\caption[A tabular table]{An example of a table whose contents are
formatted using the {\tt tabular} environment.}
\label{tab:ex2}
\hrulefill
\begin{center}
\begin{tabular}{|l|l|r|} \hline\hline
{\em type} & \multicolumn{2}{c|}{\em style} \\ \hline
smart & red & short \\
rather silly & puce & tall \\ \hline\hline
\end{tabular}
\end{center}
\par
\bigskip
\hrulefill
\end{table}
\end{listing}
\end{quote}
\linespread{1.3}\normalsize

The \verb|\multicolumn| command on line 8 allows the spread of data
over several columns, and overrides the normal vertical bar placement.
Lines 2, 6 and 14--16 produce the horizontal lines which separate
the table from the rest of the text.

\begin{table}
\hrulefill
\caption[A table with graphical input]{An example of a table which
uses graphical input as its content.  This is once again the
contents page of this document, saved as PDF.}
\label{tab:ex1}
\hrulefill
\begin{center}
\includegraphics[scale = 0.5, angle = 270]{page.pdf}
\end{center}
\par
\bigskip
\hrulefill
\end{table}


\begin{table}
\hrulefill
```

```
\caption[A tabular table]{An example of a table whose contents are
formatted using the {\tt tabular} environment.}
\label{tab:ex2}
\hrulefill
\begin{center}
\begin{tabular}{|l|l|r|}
\hline\hline
{\em type} & \multicolumn{2}{c|}{\em style} \\ \hline
smart & red & short \\
rather silly & puce & tall \\ \hline\hline
\end{tabular}
\par
\bigskip
\hrulefill
\end{center}
\end{table}
```

A.1.8 conclusion.tex

```
\chapter{Conclusion}
\label{chap:final}

Here are some final comments about putting together a thesis.  They
are mostly opinion (and certainly opinionated) so take on board what
you will.  This is free advice after all (and may be worth as much)
but it is meant to make life easier for you.

\section{10 Things You Need to Believe \ldots}
\subsection{Use Sections}
Chapters are BIG things.  Write an opening paragraph to your chapter,
then mark in each section you are going to cover.  Fill in the actual
content AFTER you have worked out just what the sections are going to
be.

\subsection{Do your Literature Review First}
Lots of people do their literature review after they have written
their program/done their experiment.  Don't fall into this trap---you
think the write-up will only take six weeks, but it won't, because
the lit.\ review will take four.  Create a BibTeX file while you do
the review, so you can cite as you go.  You would be amazed at how
many people don't do this.

\subsection{Do Not Write a Diary}
Nobody is interested in how you went about writing your program.  The
academic community is interested in how you have integrated the
existing theory with your own ideas, and the department is interested
in why you made particular design decisions.  Remember that we are a
Humanities department as well as a Science department, so we want to
see some sort of convincing argument for your ideas.

\subsection{Avoid Visual Formatting}
Try not to use commands like \verb|\vspace, \pagebreak| or
\verb|\enlargethispage|.  The whole point of \LaTeX\ is that it
provides a markup language that works perfectly 90\% of the time.
This means that when you have {\bf finished}, the last thing you
```

```
should do is print up a draft, then go through and mark any formatting
you don't like (there will probably be about one thing every ten
pages).  That is the time to insert things like \verb|\pagebreak|
commands.

\subsection{Learn \LaTeX\ Early}
The learning curve for \LaTeX\ is steep but short.  The idea is that
some poor sod like me does all the dirty work, and all you have to do
is {\em fill in the content}.  However you may spend about a week
learning all the fiddly bits (like tables and figures), and you don't
want to be taking time away from the actual thesis writing.

\subsection{Read Some Documentation}
In the Systems Lab, we are going to make every attempt to have
printed documentation lying around.  If you need quick help, check out
the file {\tt essential.dvi} on any Linux \LaTeX\ installation.  It
is probably the best short guide to \LaTeX\ around, and even contains
lots of tricky maths examples.

\subsection{Save Paper}
On a mac, use the layout options to print two-up and double-sided. On a
linux machine, use {\tt  psutils} to print your drafts.  This way you
use a quarter of the paper.  Here's what to do if you don't have
a duplex printer:
\begin{enumerate}
\item Use the twosided option and make the PDF file.
\item Use \verb|pdftops| to convert \verb|thesis.pdf| to \verb|thesis.ps|.
\item Use {\tt psbook} to arrange the pages for booklet printing;
\begin{quote}
e.g. {\tt psbook thesis.ps thesis.bk.ps}.
\end{quote}
\item Use {\tt psnup} to get the book to a booklet;
\begin{quote}
e.g. {\tt psnup -2 thesis.bk.ps thesis.2up.bk.ps}.
\end{quote}
\item Use {\tt psselect} to print up the even pages first (in reverse),
then the odd pages;
\begin{quote}
e.g.\\
type {\tt psselect -e -r thesis.2up.bk.ps | lpr}\\
Put the result into the sheet feeder, blank side up\\
type {\tt psselect -o thesis.2up.bk.ps | lpr}\\
\end{quote}
\end{enumerate}

That's the theory; in practice it goes something like this:
\begin{quote}
{\tt psbook thesis.ps | psnup -2 | psselect -e -r | lpr}\\
Now open the laserjet side-door and put the paper which came out after
the first command on the tray.  Don't change its orientation or
anything: just pick it up, move it to the tray and drop it.  Then:\\
{\tt psbook thesis.ps | psnup -2 | psselect -o | lpr}\\
\end{quote}

\subsection{Use \LaTeX, not Word}
```

```
Microsoft Word is not your friend.  Word is not {\em anybody's} friend.
It is possible to write large documents in Word, but you have to be
{\bf so} strict on yourself (using heading styles, TOC entries, etc.)
that you may as well have used \LaTeX\ in the end anyway.  Once a
problem is nailed in \LaTeX, it stays nailed.  The same is not true of
Word.  Pdflatex produces standard PDF which can be
printed anywhere, and is fast becoming a standard for on-line article
publication.  Ask anyone how many problems they have had printing Word
documents to PostScript printers.

\subsection{The First Copy is Not the Final Copy}
Well, unless you're Isaac Asimov, or Mozart. Print up
draft copies of chapters and give them to your supervisor to read,
then implement any changes when they get returned.  Then, a week
later, reread the chapter and change anything that makes you cringe.

\subsection{Spell Check your Work}
If you are using UNIX (as I have assumed throughout this document)
then the {\tt ispell} program is a pretty good spell checker.  Also
get someone to check your punctuation and grammar, and be on the
lookout for spellchecker errors (like ``fro'' instead of ``for'',
which will not be picked up at all).  Even the {\em worst} writer of
novels who gets published has a good grasp of grammar, because it is
unpleasant to read work which doesn't make sense.  There is no point
in putting the examiner in a bad mood by turning the reading of your
thesis into an unpleasant chore.

\section{General Comments}
Traditionally, academic works are written using the passive
voice---``this was done'' rather than ``we did this''.  Also, the use
of the personal pronoun ``I'' is avoided in favour of ``we''; but here
you must be careful.  There are two ways of using ``we'': either {\em
in}clusively or {\em ex}clusively.  Inclusively is pretty much alright
as it makes the reader feel like part of the story; e.g. ``Reducing
Equation 2.3 to Equation 2.4, we can see \ldots''.  Exclusively sounds
pompous---``we now present a new algorithm which will solve this
problem \ldots'' and should be avoided unless writing a paper with
more than one author.

There are some other things worth remembering:
\begin{itemize}
\item When referring to another chapter or section, Use a capital
letter; e.g. ``see Chapter~3'' not ``see chapter~3''.  Use a tilde
character $(\sim)$ to put a non-breaking space before the number.  That
way you will never begin a new line with the number.
\item The layout suggested in this document (introduction, lit.\
review, new ideas, implementation, results, conclusion) is pretty
generic.  If you stick to it, you will complete a thorough but boring
thesis.  You will almost certainly need to digress occasionally, and
perhaps integrate the literature review more with your own work.
While academic writing isn't usually noted for its racey prose, it
doesn't have to be boring.
\end{itemize}

Finally, at the end of your thesis, don't be afraid to blow your own
```

```
trumpet.  If you have done something new and original, restate just
what that is.  The last part of the concluding chapter is what most
people read straight after the abstract, so it has to be just as pithy
and imagination-capturing.
```

A.1.9 appendices.tex

```
%%
%% Now we have to get the source code in as a set of Appendices.
%% Source code will be Appendix A, with each file numbered X.y
%%
\appendix

%%
%% -> \chapter will cause the next bit to be labelled Appendix A
%% -> \section will give us A.1, \subsection A.1.1 etc.
%%
%% I suggest a section for each program and a subsection for each file
%% in the program.  Alternatively, a chapter for each program, a
%% section for each library and a subsection for each file.
%%
\chapter{Source Code for thesis.dvi}

\linespread{1}
\footnotesize

\section{thesis.tex}
\listinginput[10]{1}{thesis.tex}

\subsection{abstract.tex}
\listinginput[10]{1}{abstract.tex}

\subsection{acknowledgements.tex}
\listinginput[10]{1}{acknowledgements.tex}

\subsection{intro.tex}
\listinginput[10]{1}{intro.tex}

\subsection{literature.tex}
\listinginput[10]{1}{literature.tex}

\subsection{new\_ideas.tex}
\listinginput[10]{1}{new_ideas.tex}

\subsection{implementation.tex}
\listinginput[10]{1}{implementation.tex}

\subsection{results.tex}
\listinginput[10]{1}{results.tex}

\subsection{conclusion.tex}
\listinginput[10]{1}{conclusion.tex}

\subsection{appendices.tex}
\listinginput[10]{1}{appendices.tex}
```

```
50  %%
    %% \listinginput[x]{y}{filename} gives a listing of <filename>,
    %% starting at line y with a line-number at every xth line.
    %%
```