# Electronic Theses and Dissertations at Pitt (a LaTeX 2$_\varepsilon$ class)

Federico Garcia [*]

2004/08/17

## Contents

[*] University of Pittsburgh, Electronic Theses and Dissertations (ETD) Working Group.

**Pitt ETD Webpage:** `http://www.pitt.edu/~graduate/etd/`

**pittetd's Webpage:**
`http://www.pitt.edu/~graduate/etd/latextemplate.html/`

**TEX Users Group:** `http://www.tug.org/` (with a link to CTAN)

# 1 Introduction

This is the guide to the pittetd LaTeX 2ε document class, designed for the preparation of electronic theses and dissertations (ETD) at the University of Pittsburgh. It is recommended that users read this entire documentation before starting using pittetd, so that they will have an idea of the different possibilities and options, some of which are particular to pittetd and therefore not usual in standard LaTeX classes.

Users will find below a description of pittetd usage, extended with an introduction to some of the most relevant features of the `hyperref` package. In addition, when this document has been produced by running LaTeX on the file `pittetd.dtx`, it also contains a commented transcript of the code, so that users can modify things if they need to (and know what they are doing). In the version downloadable from the Pitt ETD webpage, this latter part is omitted. A separate document, *Comments on using LaTeX for theses*, also prepared for Pitt ETD authors, describes some standard tools of LaTeX that may or may not be known to the reader but can certainly prove useful when writing the thesis or dissertation. Touched upon are topics such as inclusion of graphics and the handling of large, book-length documents [1].

Throughout this text reference is made to the *Format Guidelines for Electronic Thesis and Dissertation Preparation at the University of Pittsburgh*, downloadable from the Pitt ETD webpage. The abbreviation FG is used to refer to it; page numbers are indicated in parenthesis.

Section 2 describes the creation of interactive PDF files through LaTeX, introducing the two main tools for that effect, the programs PDFLaTeX and dvipdfm. Section 3 explains how to install pittetd and the main LaTeX packages needed for its proper working. Also, cursory information for the installation of PDFLaTeX and dvipdfm is given.

In section 4 some general considerations are given about the best ways to use pittetd (and to cope with its restrictions). Use of packages and the issues it might bring about (notably incompatibility)is treated in a special subsection.

Detailed information about the options, available commands, and use of pittetd can be found in section 5. Typesetting of the preliminary pages is described in subsection 5.4.

Section 6 describes in an introductory way the basic features of `hyperref`, the package that implements interactivity into LaTeX documents.

Finally, section 7 gives some suggestions for a final format review before submitting an ETD written with pittetd. It warns about those problems that are most likely to occur because they lie beyond pittetd's control.

## 2 PDF creation through LaTeX

In principle, pittetd is equipped to fulfill the basic interactivity requirements of the FG, namely the creation of bookmarks from the entries in the Table of Contents, the List of Figures, and the List of Tables, and the implementation of these entries themselves as links to the corresponding page. This is done by means of invoking the formidable hyperref package,[1] which offers the basic functions for interactive handling (section 6 below offers an introductory guide to other features from this package that users can take advantage of). Thus, hyperref has to be—and it usually is in standard distributions of LaTeX—installed in the system for pittetd to be able to fulfill these tasks (section 3 offers immediate help on the installation of hyperref and other tools, including pittetd itself).

pittetd has been written under the assumption that the user will create the final .pdf file through one of two tools, namely PDFLaTeX, or the program dvipdfm. The user indicates which of the two ways is to be used as an option to the pittetd class, namely pdftex or dvipdfm (on the way to load the class and specify options, see section 5.1). The following sections explain the particularities of each way. A third related option, nohyperref option, will be discussed in section 2.4. Note that it is possible to switch back and forth between the three ways just by modifying the relevant option; as far as pittetd is concerned, nothing else is necessary to effect the change.[2]

### 2.1 PDFLaTeX

The most direct way to obtain a .pdf output file is running PDFLaTeX[3] instead of LaTeX. Naturally, PDFLaTeX has to be installed in the system (again, it is usually included in standard distributions of LaTeX; see section 3). The user has to indicate pdftex as an option to pittetd, and

[1] Written by Sebastian Rahtz.

[2] However, when going to/from nohyperref, it is always good to delete any auxiliary files before running. Also, some of the hyperref package's commands discussed in section 6 are of course disabled when nohyperref is used.

[3] Created by Hàn Thé Thàn.

this latter will pass that option to other packages that need it, including `hyperref` but also `graphicx` and `color` (this latter used by `hyperref`).

Note that if this is the chosen method, a regular LaTeX (i.e., not PDFLaTeX) run will result in an error message ('`Why not use pdf(e)TeX binaries?`'). This could affect user's habits, batch files, etc.

There is one more significant drawback to the use of PDFLaTeX: the running time is sometimes clearly longer than regular LaTeX. This depends, to be sure, on one of the configuration options of PDFTeX, namely `\pdfcompresslevel`, and it could be modified. But in that case, the resulting file is incomparably larger.[4]

## 2.2 dvipdfm

As the name indicates, dvipdfm[5] is one of the programs available to convert `.dvi` files into `.pdf`. The procedure then consists in running LaTeX as usual while the document is in preparation, having loaded pittetd with `dvipdfm` option, thus obtaining (more quickly) the usual (and smaller) `.dvi` output. Only optionally, at strategic points in the development of the document (notably at the end), has the user to worry about PDF, and apply dvipdfm to the `.dvi` file. This is usually as simple as typing

```
dvipdfm doc.dvi
```

in the command line. The file `dvipdfm.pdf` is the user's manual for the program and explains the switches that can be used in the command line.

In some `.dvi` viewers the bookmarks (and even the links) are lost. But the relevant information is recovered by dvipdfm when creating the corresponding `.pdf` file.

Another significative advantage of dvipdfm is that it tries to solve inclusion of PostScript graphic files, so it is not always necessary to convert them (see also [1]). To do the job, however, dvipdfm uses GhostScript, and therefore this program must also be installed in the system.

## 2.3 Other ways to get a PDF file and bookmarks

There are other ways to obtain a final PDF output file, but they are all discouraged to use alongside with pittetd. For example, a common method

[4] In general, a `.pdf` is much larger—much less efficient in all respects—than the `.dvi`. This tendency is reinforced if PDFLaTeX is configured to run faster.

[5] Written by Mark A. Wicks.

is to use `dvips` to convert a file to the PostScript format, and then apply Acrobat Distiller on it. This method involves two conversions; links and especially bookmarks tend to have an erratic behavior.

Acrobat PDF Writer, a 'printer emulator' that 'prints' PDF files will, of course, ignore anything that cannot be printed, including bookmarks. And the other `.dvi`⤳`.pdf` converter in existence, `dvipdf`,[6] is not as widely available as dvipdfm.

On the other hand, `hyperref` is not the only way to create bookmarks with LaTeX. Older packages and systems exist, like VTeX, but since those are much less widely used, pittetd does not support them.

### 2.4 No hyperref

There is a third option concerning the creation of bookmarks and links in pittetd. Option `nohyperref` will prevent pittetd from taking care of almost all interactivity requirements, and the user is left the freedom (and the burden) to fulfill them by him- or herself.

This option might be more useful than it seems, because it allows users to use the `hyperref` package itself *their way*, not pittetd's. There are in the latter's code a series of minor, but substantial, modifications to `hyperref`, and some of the options with which the package is loaded are fixed. As a security measure, pittetd will not allow the user manually to load the package, unless `nohyperref` is specified. Thus, if a user wants to control `hyperref`'s behavior, this option will be necessary. Section 6.2 gives some directions on how to do this.

Also, if pittetd cannot run normally due to some complication in installation or configuration of `hyperref`, the `nohyperref` option provides a way to keep working on the contents of the document and worry about requirements later.

## 3 Installation

### 3.1 pittetd

The pittetd bundle is made of the following files:

[6] By Sergey Lesenko.

| | |
|---|---|
| pittetd.dtx | Source for the class and this documentation. |
| pittetd.ins | Batch file for installation. |
| pittetd.cls | The pittetd class itself. |
| pit10pt.clo | Definitions for 10pt-size option. |
| pit11pt.clo | Definitions for 11pt-size option. |
| pit12pt.clo | Definitions for 12pt-size option. |
| pitthesis.pit | Patch for pitthesis class |
| pittdiss.pit | Patch for pittdiss class |
| achicago.pit | Patch for achicago package |
| pittetd.dvi | This documentation |
| pittetd.pdf | |

All these files are individually downloadable from pittetd's webpage. It is only the two first files, however, that are necessary, for the rest can be extracted from them. To do this, the file `pittetd.ins` has to be processed with TeX (*not* LaTeX); the documentation results from running LaTeX (*not* TeX) on `pittetd.dtx`.[7]

It is the `.cls` and `.clo` files that conform the class itself, i.e., what LaTeX needs to have access to. Under a system that, like most TeX implementations today, use the standard TeX Directory Structure (TDS), LaTeX files are put in subdirectories of the `.../texmf/tex/latex` directory (for example, the standard classes are in `.../texmf/tex/latex/base`). So the best thing under such a system is to create a subdirectory for pittetd:

```
.../texmf/tex/latex/pittetd
```

and place there the `.cls` and `.clo` files. Likewise, the documentation (the file you are reading, `pittetd.dvi`) should be placed in

```
.../texmf/doc/latex/pittetd
```

and the source files (`pittetd.dtx` and `pittetd.ins`) in

```
.../texmf/source/latex/pittetd
```

The 'patches' should be placed in the same directory as the actual document's input files.

After placing the files in those directories, you might need to 'refresh' the database, i.e., to make TeX aware that a new class is loaded. This usually

[7] To get the index right, you have to run `makeindex` with `gind` style, saying, in the command line (and after a LaTeX run on `pittetd.dtx`), `makeindex -s gind.ist pittetd`. Then a final `latex pittetd.dtx` produces the document with a well-formatted index.

appears as a command (or button, or window, etc.) of the implementation.[8]

For non-TDS systems, the suggestion is 'put the files where TEX can find them.' For example, search your disk for the standard classes (e.g., `article.cls`), and put the pittetd files where they are. Alternatively, you can simply put the pittetd files in the directory that contains the input files of your document.

### 3.2 Installation of other required packages

In addition, you will need at least the `hyperref` and `color` packages, and PDFTEX if you use `pdftex` option. Most likely, you already have those packages installed. Even so, it is possible that you do not have the file `pdftex.def`, which is part of only relatively recent distributions. This file, available from the pittetd's webpage, should be copied to the same directory where the file `color.sty` is (`.../texmf/tex/latex/graphics` in a TDS system).

`hyperref` is a package used by pittetd (unless, of course, the `nohyperref` option is used), so it has to be in the system. In the very unlikely case it is not already installed, you will need to download it from either CTAN (through `http://www.tug.org`) or the pittetd's webpage, and install it by running TEX (*not* LATEX) on the file `hyperref.ins`. This will extract the files and instruct you on where to place them (which, in any case, is analogous to the placement of pittetd files).

Likewise, `hyperref` uses other packages from the standard distribution of LATEX (notably, `color`). Installation of those packages is analogous.

### 3.3 PDFTEX, dvipdfm

Installation of PDFTEX and dvipdfm is a more complex matter. Again, several implementations, including TEXLive, MiKTEX, teTEX, fpTEX, and CMacTEX, have both tools pre-installed. In case your system does not have either or both of them, you can download the relevant files, and obtain installation directions, at CTAN (through `www.tug.org`). The PDFTEX manual, file `pdftex-s.pdf`, is available from the pittetd's webpage, and contains information on the installation of the program.

[8] With MiKTEX, for example, you should run the program 'MiKTeX Options.'

# 4 To keep in mind

The pittetd class has been designed to fully comply with the format guidelines for Pitt ETDs. Due to this, there are some particularities that might create conflict with LaTeX users' habits. This section warns and advises about those particularities. Decisions have been made with two priorities: to discourage uses that go against the FG, and to facilitate conversion from standard LaTeX classes.

## 4.1 Headings and captions

pittetd will automatically capitalize the title of the document and those of the chapters. However, section titles have to be capitalized by the user.[9]

On the other hand, since both sectional headings and captions for tables and figures must have entries in the bookmarks panel, they are subject to two substantial limitations: they cannot be long, and must consist only of ASCII characters.[10]

When building the bookmarks, `hyperref` will convert some simple LaTeX commands, but in general will ignore most of them. It also will crop everything that goes beyond the maximum length of a bookmark (that varies among PDF viewers; Acrobat Reader makes it 64 characters). There are two tools to handle these limitations in LaTeX: the `hyperref` command `\texorpdfstring` (section 6.1), and the optional argument to `\caption` (section 5.6.1).

## 4.2 Preliminaries

The series of preliminaries in a Pitt ETD differs substantially from a paper-based thesis/dissertation. The committee page has changed, and dedication and acknowledgements pages have been eliminated (under the assumption and recommendation that these should be part of the preface). The order was modified accordingly (so that the preface goes immediately before the text of the thesis itself).

Thus, the preliminaries are in principle limited to the following:

[9] This is because modification of the `\section` command in order to capitalize not only the title itself, but also the bookmark, although possible, would highly increase the probability of incompatibilities with other packages.

[10] That is why the FG recommend using words (not formulas) in titles and keeping captions "to one line if possible" (p. 9).

Title page
Committee Membership page
Copyright page (optional)
Abstract
Table of contents
List of Tables
List of Figures
Preface (optional)

Since all these preliminaries have their own commands in pittetd (see section 5.4), there is in principle no need nor place for non-numbered chapters (`\chapter*` commands). In fact, the starred version behaves exactly as the regular one. If there is a need for additional preliminary pages, the (on purpose) cumbersome command `\preliminarychapter` is available (see section 5.4.7).

## 4.3 Use of packages

Almost certainly authors of Pitt ETD's will need to load a wide and unpredictable variety of packages. Although pittetd has been coded with the premise not to 'invite' incompatibilities, it is possible that some of these packages will create clashes, for there is simply no way to claim universal compatibility with the hundreds of packages already available and with those to come.

However, *partial* compatibility can be (and supposedly has been) achieved. A survey carried out in April–May 2003 gave us a list of packages that are of common use in the Pitt community, and those have been taken into account in the writing of pittetd.

For the handling of possible incompatibilities arising in the future, the following policy has been designed. The user who suspects he or she has found a clash should contact the Pitt ETD Working Group and explain the problem, ideally e-mailing a copy of the input file(s). Hopefully in a reasonable amount of time, a 'patch' will be created that solves the problem. The patch takes the form of a file with extension `.pit`, downloadable from pittetd's webpage.

After the file has been downloaded and put where LaTeX can find it (the easiest way is to put it in the same folder as the document itself), it should `\patch` be accessed. The command `\patch`, that takes the name of the package as its argument, reads any patch that exists for it. For example, there is already a patch for the `achicago` package; to ensure the proper behavior of this package, the user should type, *after* `\usepackage{achicago}`, the command `\patch{achicago}`.

`\usewithpatch`

Alternatively, the command `\usepackage` can itself be replaced by `\usewithpatch`. When a package is invoked by means of `\usewithpatch`, pittetd will search the system for the corresponding patch; if it exists, it loads it; if not, nothing happens. Options to the package, as usual, are indicated by the optional argument `[`⟨*options*⟩`]`. The drawback of this mechanism is that several packages *cannot* be loaded at once (i.e., by comma-separating them, as in `\usepackage{color,graphicx}`); each must receive its own `\usewithpatch`. But using `\usewithpatch` ensures that pittetd will always look for a patch when loading a package.

The following paragraphs mention some LaTeX packages and tell whether they are supported or not by pittetd. For information on compatibility with the bibliographical styles and packages, see section 5.8.

### 4.3.1 Unsupported packages

Many popular LaTeX packages provide formatting features that either go against the FG or are already incorporated into pittetd. Therefore, it is assumed that such packages will *not* be loaded. These include `setspace`, packages for the handling of floating objects (such as `float`, `floatflt`), for variations of layout (`fancyhdr`, `fncychap`, `multicol`), and sectioning (`titlesec`, `tocbibind`). Using any of those packages might result in error messages, anomalies, and unpredictable output. Before reporting or trying to solve these problems, keep in mind that departmental approval is needed to include the features.

### 4.3.2 Supported packages

Some packages provide features that are legitimate in a Pitt ETD. Font packages, such as those in the PSNFSS collection (`times`, `bookman`, `palatino`, `newcent`, etc.) are perfectly compatible with pittetd. In fact, if CM fonts are desired, it is recommendable that the `ae` package is loaded.[11]

The packages of the American Mathematical Society (`amsmath`, `amsthm`, etc.) are supported. Likewise, all the packages in the standard distribution

[11] The AE fonts emulate CM, but are PostScript, not bitmap, fonts, which gives them a better quality for screen display.

of LaTeX (`color`, `graphicx`, `xspace`, `verbatim`, etc.) are supposed to work. `caption2` works miraculously fine. In general, packages that *provide*, as opposed to *override*, features, should work fine.

## 4.4 Related classes

Currently there are two LaTeX classes that produce theses and dissertations for Pitt, namely pitthesis (by Wonkoo Kim, 1999) and pittdiss (by Will Slaughter, 2003). The former was designed for paper-based documents, following requirements somewhat different from those of an ETD; the latter, on the contrary, was created with ETD in mind.[12] Many features are shared by those classes and pittetd—notably the creation of preliminaries—but the detailed mechanisms (command names and things like that) are different. This release of pittetd includes two 'patch' files that allow using pittetd with conventions from the other two classes, so that the user does not have to change every command (some will require handling, though; pittetd will warn or complain).

The patches are called through either

```
\patch{pittdiss}
```

or

```
\patch{pitthesis}
```

(the latter only two t's.) Having read the corresponding patch, pittetd will try to interpret pittdiss- or pitthesis-commands. Hopefully, most times it will succeed; in any case, it will issue warnings (or error messages in the `final` option) for things that *have* to be changed. For example, if acknowledgements are created with pitthesis's `acknowledgements` environment, pittetd will warn that now there is no separate preliminary for that, and that this section should be part of the preface.

As an extra safety measure when going from pitthesis or pittdiss to pittetd, all auxiliary files should be deleted before the first pittetd run. Also, it is very much recommendable to change `\bibliography` to `\safebibliography` since the beginning.

## 4.5 Stage and interaction

In order to facilitate the process of converting files from standard classes into pittetd, annoying error messages due to the particularities of pittetd

[12] It does not produce links or bookmarks, however.

have been avoided as much as possible. The 'stage' of the document is used to decide whether or not the differences should make stop the LaTeX run. The idea is that when the `draft` option is used, most problems are reported as 'Class Warnings' that do not stop the process. But if `final` is used (and eventually it *should* be used), more prominent error messages appear instead.

However, many packages (including the seminal `hyperref`, `color`, and `graphicx`) themselves operate differently when `draft` is specified. So for example, `hyperref` does not create links or bookmarks, and `graphicx` does not import external graphic files. But the user might want to see these features, still not worrying about detailed pittetd concerns. That is why an intermediate stage `semifinal` is introduced. The packages will work as usual, but pittetd will issue mostly warnings, not error messages. This is the default option.

Both `semifinal` and `draft` issue a final warning at the end of the job, reminding the user to run the document with `final`. As usual, moreover, `draft` makes overfull boxes visible.

The 'stage' option also governs pittetd's complaints about the preliminaries when information for the different pages is missing, when the order is wrong, etc.: with `draft` and `semifinal`, there will be a warning, while with `final` there will be an error message.

### 4.6 Auxiliary files

In addition to the regular auxiliary files (`.aux`, `.toc`, `.lot`, `.lof`, etc.), a run of pittetd involving all its features will produce two files: one with extension `.out` (written by `hyperref` for the 'outlines,' or bookmarks), and one with extension `.etd`, used by pittetd to decide some details. Input or other files created by the user should avoid these extensions. Also, for some implementations of LaTeX that provide a quick erasing of auxiliary files, it is advisable to configure this tool to include `.out` and `.etd` files.

### 4.7 PDF Document Info

pittetd offers the possibility of filling in the fields of Acrobat Reader's 'Document Info' dialog box. 'Title' and 'Author' are filled in with the data of the title page. 'Subject' and 'Keywords' are handled by additional commands (see section 5.5).

# 5 Using pittetd

## 5.1 Loading the class

The pittetd class is loaded by typing

```
\documentclass[⟨options⟩]{pittetd}
```

at the very top of the input file. Table 1 shows all the ⟨*options*⟩ available. Most of the options for conventional classes (i.e., the standard classes article, book, report, and similar ones like amsart and amsbook) have been disabled in pittetd. The document will always be typeset letter paper (8.5 × 11 inches), portrait, and one column.[13]

| Characteristic | Available Options |
|---|---|
| Font size | `12pt` (default)<br>`11pt`<br>`10pt` |
| Stage<br>(see section 4.5) | `final`<br>`semifinal` (default)<br>`draft` |
| Bibliography layout | `openbib` ('open' bibliographies). |
| Equations | `leqno` (equation numbers on the left)<br>`fleqn` (flush-left displays) |
| Type | `phd` (dissertation, default)<br>`ms` (M.S.'s thesis)<br>`ma` (M.A.'s thesis) |
| Section numbering<br>(see section 5.3.2) | `sectionnumbers` (default)<br>`sectionletters` |
| PDF production<br>(see section 2) | `dvipdfm`<br>`pdftex`<br>`nohyperref` |

Table 1: Available options for pittetd

[13] Thus, options to modify these parameters, namely those for *a*) paper size (`letterpaper`, `legalpaper`, etc.); *b*) paper orientation (`portrait`, `landscape`); *c*) number of columns (`onecolumn`, `twocolumn`); and *d*) pagination (`oneside`, `twoside`; `openright`, `openany`; `titlepage`, `notitlepage`), are not implemented.

## 5.2 Font sizes and spacing

\Small
\SMALL

The usual LaTeX commands are defined according to the font size option selected. In addition, the `\Small` and `\SMALL` commands work as in the classes amsart and amsbook, i.e., are equivalent to `\footnotesize` and `\scriptsize` respectively. See figure 1.

The text of a Pitt ETD has to be at least "one half-spaced, with the exception of long quotations, footnotes, bibliographical references, and the Index (if included), which may be single-spaced" (p. 7). A spacing of little more than one-half for regular text has been built in into pittetd; the text in footnotes and quotations has been set to single spacing. The user can always adjust the spacing in the usual way, `\renewcommand`'ing the command `\baselinestretch`, so that

\baselinestretch

```
\renewcommand\baselinestretch{1.3}
```

increases the built-in spacing by a 30%—for *all* the text, footnotes included.

The spacing-scheme is achieved in pittetd by building it into the font sizes. Normal-size font (`\normalsize`) is one-half spaced, while all other sizes are single-spaced. The `quote` and `quotation` environments, as well as `\footnote`, all of which set a smaller font, produce thus single-spaced text. An additional 'size' has been implemented, namely `\singlespace`, which produces regular-size, but single-spaced, text.

\singlespace

\smallskip
\medskip
\bigskip

The spacing command `\smallskip` is set to an amount of a single space; `\medskip` is a line (a little more than one and a half space); and `\bigskip` a double space.

| \tiny | \SMALL or<br>\scriptsize | \Small or<br>\footnotesize | \small |
|---|---|---|---|

\normalsize

\large \Large \LARGE \huge \Huge

Figure 1: Font sizes

\acro

Adapted from the ltugboat class, pittetd implements the command `\acro`, that typesets its argument in a font smaller than the surrounding text. It is useful for all-uppercase acronyms like ETD (`\acro{ETD}`), UNICEF (`\acro{UNICEF}`), etc., which would be too large in regular size (compare ETD, CTAN, UNICEF); it is better than the direct `\small`, which is not good in contexts of font size other than normal.

## 5.3 Sectioning

### 5.3.1 Sectioning commands

`\chapter`
`\section`
`\subsection`
`\subsubsection`

The sectioning of a `pittetd` document is done through the usual commands `\chapter`, `\section`, `\subsection`, and `\subsubsection`. Note that `\part`, `\paragraph`, and `\subparagraph` are *not* implemented. The `\chapter` command takes care of capitalization of the title *both* in the text and in the bookmarks; however, since `\section` capitalizes in the text but not in the bookmarks, it is always advisable to capitalize manually.

Within preliminaries, the subdivisions `\section`, `\subsection`, and `\subsubsection` will produce neither a number nor a bookmark entry (`\chapter` is reserved for chapters in the body of the text; about additional preliminary 'chapters' see section 4.2). The starred variants `\chapter*`, `\section*`, etc., work exactly as the non-starred counterparts, although producing a warning.

All four sectioning commands have the usual optional argument, that contains the alternate version of the heading that appears in the table of contents. This, however, is implemented only for compatibility reasons, for the FG require that the table of contents lists the headings exactly as it appears in the text. The main reason why the optional argument could be used at all is that it permits to cope with the conversion of the text into ASCII text for the bookmarks, but that is best handled by the command `\texorpdfstring` (section 6.1).

On the other hand, there might be cases in which some letters must appear in lowercase even in headings (chemical elements is such a case). The command `\lowercase` works within the arguments to sectioning commands, and can be used for those cases.

### 5.3.2 Numbering

`\regularenum`

The divisions of a Pitt ETD can be numbered in two ways, depicted in Figure 2. The first one is the one used by default (`sectionnumbers`); the user can specify `pittetd`'s option `sectionletters` to use the second one. In this case, in addition, the labels for successive levels of the `enumerate` environment are also changed from their default appearance, to agree with the section numbering: the first level will be an uppercase roman numeral, the second an uppercase letter, and so on. The user has the command `\regularenum` to revert to the usual appearances (namely arabic, letter, roman, Letter).

| | |
|---|---|
| 1.0 FIRST CHAPTER | I. FIRST CHAPTER |
| 1.1 FIRST SECTION | A. FIRST SECTION |
| 1.11 First subsection | 1. First subsection |
| 1.11.1 First subsubsection | a. First subsubsection |
| `sectionnumbers` (default) | `sectionletters` |

Figure 2: The two possibilities for section numbering

## 5.4 The preliminaries

The first part of a Pitt ETD is made of the 'preliminaries.' They are created in pittetd with special commands that are the subject of the present section.

pittetd will keep track of the order in which the user typesets the preliminaries, and will warn or complain according to the 'stage' of the document (see section 4.5).

### 5.4.1 Title Page

`\maketitle` The title page is produced, as usual, by the command `\maketitle`, but involves several pieces of information in addition to `\title`, `\author` and `\date`, so it is only *deceivingly* similar to the same command in standard LaTeX classes. All efforts have been taken to prevent the differences to ruin the LaTeX run (making easier the conversion from other classes), but the user will eventually have to check it carefully.

`\title` The macro `\title` has an optional argument that sets the title of the document as it will appear in the 'Document Info' dialog box of Acrobat Reader. If no optional argument is given, the required argument will be used (however, bear in mind that only a limited portion of it will be visible). In order for this feature to work properly, `\title` must be issued in the preamble of the document. In the text, the title will be typeset uppercase. **Ex.:**

```
\title[An Anatomy of the World]{An Anatomy of %
                  the World on texts by John Donne, %
                for soprano and six instrumentalists}
```

`\author` The `\author` command works much the same as in standard LaTeX classes. Again, it should be issued in the preamble for the author's name to appear in the 'Document Info' dialog box. On the other hand, `\thanks` and `\and` are disabled. **Ex.:** `\author{Federico Garcia}`

The following macros set other information needed by pittetd to build the preliminaries. None of them is required unless `final` option is used. With `semifinal` and `draft`, a warning is issued informing of any missing commands.

These commands are analogous, but not identical, to additional commands in the classes pitthesis and pittdiss. Patches are available to facilitate conversion from those classes to pittetd.

`\year` The title page does not include the whole date, but only the year. By default, this is set to the current year; the user can optionally specify it with the command `\year`. **Ex.:** `\year{2002}`

`\degree` The information of the author's previous degrees is provided by the `\degree` command, and should contain the degree, institution, and year of the each degree. Several lines or degrees can be separated with `\\`.
**Ex.:** `\degree{B.S. in Music (Composition),\\Bogot\'a, 2001}`

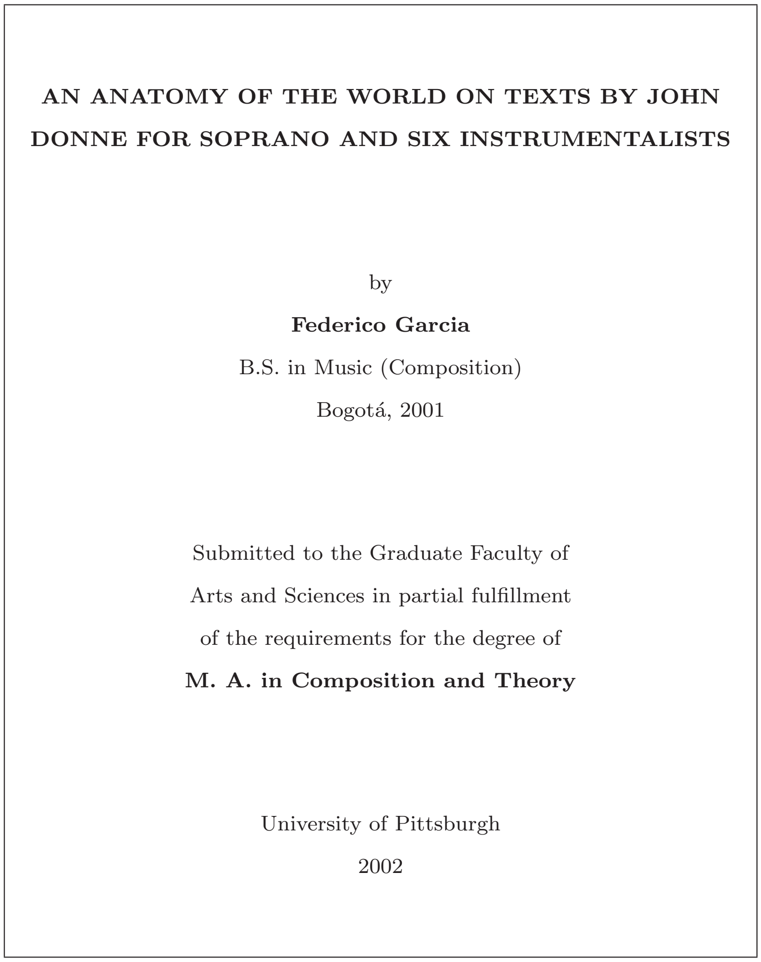
`\school` The title page includes the text 'submitted to the graduate faculty of', followed by the school name. The user sets this name with the command `\school`. By default, the article 'the' is appended to the school name, but the user can change it with the optional argument.
**Ex.:** `\school{Department of Mathematics}`
**Ex.:** `\school[]{FAS}`
**Ex.:** `\school[certain]{Other department}`

`\degreesought` According to the option used (`phd`, `ms`, or `ma`), pittetd sets the value of `\degreesought` to either 'Doctor of Philosophy,' 'Master of Sciences,' or 'Master of Arts'. If desired, `\renewcommand` can be used to modify it.
**Ex.:** `\renewcommand\degreesought{M. A. in Composition and Theory}`

Figure 3 is the title page produced by the examples above.

### 5.4.2 Committee membership page

`\makecommittee` The `\makecommittee` command builds up the committee membership page. The author is typeset as it was in the Title Page (i.e., as is provided by the `\author` command); the school name comes initially from the `\school` command, although capitalized (see section 5.4.1 for these two commands). In some cases, the name of the school in the committee membership page



**AN ANATOMY OF THE WORLD ON TEXTS BY JOHN DONNE FOR SOPRANO AND SIX INSTRUMENTALISTS**

by

**Federico Garcia**

B.S. in Music (Composition)

Bogotá, 2001

Submitted to the Graduate Faculty of
Arts and Sciences in partial fulfillment
of the requirements for the degree of
**M. A. in Composition and Theory**

University of Pittsburgh
2002

Figure 3: Example of title page

should be different from the one that appears on the Title Page (for example, when it starts with ‘Faculty’); the user can insert a new `\school` command right before `\makecommittee` (and after `\maketitle`) to control the second appearance.

In addition to that information, the committee membership page takes also the date and the committee members, which are provided with the next commands.

`\date` The `\date` command is intended for the date of the thesis/dissertation defense, which will appear after the text ‘It was defended on’. The default value is `\today`. It can be omitted with `\date{}` (in whose case there will be no ‘It was defended’), but a warning will be issued.

**Ex.:** `\date{May 15, 2003}`

`\committeemember` The list of committee members is typeset with information from one or more `\committeemember` commands (one for each member). The argument of `\committeemember` cannot contain more than one line.

**Ex.:** `\committeemember{N. Chimpsky, Ph.\ D., Professor}`

The *first* name will be treated as the thesis/dissertation advisor. When there are two advisors, the second one should be provided with the `\coadvisor` command.

`\coadvisor`

In master’s theses, inclusion of the rest of the committee is optional, but in Ph. D. dissertations it is required. Accordingly, if pittetd has been loaded with the `phd` option, it will require at least two `\committemember` commands, the requirement taking the form of a warning for `draft` and `semifinal` options, an error for `final`.

In any case, if just one member (the advisor) is listed, pittetd will omit the text ‘approved by’, only typesetting the advisor’s name at the bottom of the page.

### 5.4.3 Copyright page

`\copyrightpage` Optionally, a copyright page can be appended immediately after the committee membership page, through the command `\copyrightpage`.

### 5.4.4 Abstract

`abstract` An abstract of no more than 350 words is required for every Pitt ETD. It is created as usual with the `abstract` environment:

```
\begin{abstract}
  ⟨text of the abstract⟩
 \end{abstract}
```

The page will start with the title, the author, and the year of the document, followed by the text of the abstract.

Optionally, a list of keywords or descriptors can be appended at the end of the abstract. The keywords themselves have to be set in the preamble by the command `\keywords` (section 5.5). Then, an optional argument to the `abstract` environment sets the title of the list. For example, the command `\begin{abstract}[Keywords:]` produces, after the text of the abstract, the expression '**Keywords:**' followed by the contents of the previous `\keywords` command.

Some schools (including the School of Engineering) recommend that the word 'ABSTRACT' appears on the abstract page. pittetd provides for that requirement in the form of a starred version for the `abstract` environment:

```
\begin{abstract*}
  ⟨text of the abstract⟩
\end{abstract*}
```

Keywords can be appended to this kind of `abstract` in the same way.

### 5.4.5 Table of Contents, and Lists of Figures and Tables

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

The table of contents and the lists of figures and tables are created with the usual LaTeX commands. If `hyperref` is used, the entries in these lists are links pointing to the corresponding page, and are included as bookmarks.

### 5.4.6 Preface

`\preface`

The preface is optional. If one is desired, the user needs only to type `\preface` followed by the text itself. Acknowledgements, dedication, etc., should be included in this preliminary. The preface is the only preliminary that is included in the table of contents.

### 5.4.7 Additional preliminaries

`\preliminarychapter`

As has been said, preliminaries in a Pitt ETD are in principle limited to those described above. Just for the sake of completeness, however, a command for additional preliminaries is implemented (and its use is discouraged) in pittetd:

```
\preliminarychapter{⟨heading⟩}
```

The ⟨*heading*⟩ will be both typeset and bookmarked, but not included in the table of contents). Sections within the additional preliminary will be unnumbered.

## 5.5 PDF Document Info

The 'Document Info' dialog box of Acrobat Reader includes information for title, author, subject, and keywords. `pittetd` will fill in these fields (if `hyperref` is used) with, respectively: the optional argument to the command `\title`; the `\author`; the `\subject` command; and the `\keywords` command. All four commands must be issued in the preamble for the information to go to the Document Info (although there is no error message if any or all are missing).

`\subject`
`\keywords`

For example, the commands `\subject{Musical Composition}` and `\keywords{Music \& Text, John Donne, Vocal Music}` define the contents of the 'subject' and 'keywords' fields. The latter will, optionally, also be typeset at the end of the abstract (see section 5.4.4).

## 5.6 Main body

The way the main body of the document is typeset by LaTeX is very little modified by `pittetd`. As has been said, footnotes and quotations appear in a smaller font, and single-spaced. Within the `table` and `figure` environments, moreover, `\singlespace` is declared, so their contents appears single-spaced. To resort to one-half spacing, the declaration `\normalsize` is enough.

### 5.6.1 Numbering and captions for tables and figures

`\chapterfloats`

By default, figures and tables are numbered consecutively (1, 2, etc.), independently from the chapter. This can be changed with the `\chapterfloats` command, that has to appear before `\begin{document}`. In that case, figures and tables will be numbered within chapters (1.4, 2.5, etc., or I.4, II.5, etc.); `pittetd` reserves enough space for the figure or table number in the list of figures or tables (that might be something long like 'VIII.14') , but this requires several runs.

`\caption`

As has been mentioned, captions are subject to the limitations of bookmarking: they must be short and contain only ASCII text. In case this poses problems, the optional argument to the `\caption` command is the best tool to deal with them:

```
\caption[⟨alternate caption⟩]{⟨caption⟩}
```

When present, it is ⟨*alternate caption*⟩, instead ⟨*caption*⟩, what is actually typeset in the list of tables or figures, and into the corresponding bookmark. So, if a long caption is necessary, it can be handled as in the following example (note the avoidance of `\cite` in the optional argument):

```
\caption[A modern ‘wave model’ of the Indo-European
languages according to Raimo~Antilla~(1972).]{A
modern ‘wave model’ of the Indo-European languages
according to \cite{r-a}. The numbers indicate 24
isogglosses (similarities) shared among different
Indo-European languages. Isogloss 1 indicates the
centum:satem split...
}
```

Refer also to section 6.1 for more details on `hyperref` conversion of TEX into ASCII text.

### 5.6.2 Cross references

When using `hyperref`, cross references created with the `\ref` and `\pageref` commands are interactive links. The package offers, as an alternative, the command `\nameref`, that is used exactly as `\ref`, but typesets the *name* of the chapter or section, instead of its number.[14] This kind of reference seems to be more consistent with interactivity (for, when a click is enough, the main motivation for an ordered numbering is called into question).

In any case, with pittetd, the `\nameref` command is slightly modified when it refers to an appendix: it does not produce the appendix's title, but its label (‘APPENDIX’, or ‘APPENDIX A’, etc.).

## 5.7 Appendices

`\appendix` The `\appendix` command tells pittetd that the following chapters (i.e., the following `\chapter` commands) are appendices. If there is only one appendix, its heading will be ‘APPENDIX’; if there are more, they will be numbered with capital letters, ‘APPENDIX A’, etc. pittetd needs a second run to know which way to follow.

[14] `hyperref` achieves this by means of invoking the `nameref` package.

## 5.8 Bibliography

### 5.8.1 BibTeX styles

This section applies only to documents whose bibliography is generated through BibTeX. Manually-created bibliographies (i.e., produced with the `thebibliography` environment) need no special warning to work properly with pittetd, which handles spacing after the FG (single space within entries; entries separated by one-half space).

As far as pittetd is concerned, there are three kinds of BibTeX styles (`.bst` files). The first kind includes the styles that limit themselves to ordering and formatting the different pieces of information within the bibliography entries (without modifying the appearance of the list as a whole). The vast majority of BibTeX styles, including the standard ones (`plain`, `unsrt`, `alpha`, `abbrv`), fall in this category. These styles pose no problem to pittetd, and nothing special is needed to fulfill the requirements of the FG.

The second group comprises those styles that, in addition to the individual entries, format the list as well. In general, styles that do not use bracketed labels ('[1]' or '[Cas44]') are part of this group, for they need to redefine the `thebibliography` environment to conform to the absence of such labels. They usually come with an associated package (`.sty` file) that takes care of this task. To ensure proper behavior when using these packages, pittetd offers the command `\safebibliography`. Its use is identical to that of `\bibliography`, and it tries to make a compromise between the style's conventions and the FG.[15]

`\safebibliography`

The last kind of bibliography styles is that of systems that modify aspects of formatting other than the final list of bibliographical references. All these systems have not only `.bst` files, but also substantial packages (`.sty`). `harvard`, `natbib` and `achicago` are common instances. When pittetd has a close encounter with packages of the third kind, there can be erratic behavior. It is recommended that `\safebibliography` is used instead of `\bibliography`, but this will probably not be enough. Since there is no general solution, the problems have to be treated individually, with patches, as explained in 4.3 above.

The three mentioned systems have already been tackled: `natbib` is an extremely well-written program, so that conciliating it with pittetd is easy

[15] It of course is not guaranteed that it will always succeed in doing so. If it does not, it is possible that the bibliography style is actually of the third kind.

and does not merit a separate patch file. No special treatment (other than using `\safebibliography`) will normally be needed.

On the other hand, `harvard` is a more complicated case, for the package creates interactive links. The `hyperref` package has support for `harvard`, but there is no way to foresee potential problems. It is strongly recommended if possible not to use this package, replacing it with `natbib`.

`achicago` poses other kinds of problems. It is an ambitious package that modifies things other than bibliography-related functions. For example, using this package, the effect of `\emph` will not be *italic*, but *slanted* shape; the `quote` and `quotation` environments are also modified, so that pittetd cannot set single spacing within them. Again, it is recommended to avoid this package, but in any case there is a patch available at pittetd's webpage, the file `achicago.pit`. It should be loaded saying `\patch{achicago}`.

Thus, through the means just explained, a broad range of bibliographical usages is supported by pittetd. Bracketed-labels referencing, being what LaTeX is designed for, can generally be used without restriction; for author-year referencing, `natbib` and `achicago` are supported; and for footnote referencing, the package `opcit` (available from CTAN) works fine if `\safebibliography` is used.

### 5.8.2 Citation packages

There are some packages that handle the way bibliographical references are handled within the text, rather than the way the entries of the final list are typeset. It is unfortunate that the package `cite`, that sorts the numbers of a multiple `\cite`, creates deep and quite un-traceable conflicts with `hyperref`. The package can be loaded, but it will have no effect. As a result, the `overcite` package will *not* sort the numbers either, although it will typeset them as superscripts (which, in addition, will be interactive links). `achemso` also causes problems, and it is recommended not to use it at all.[16] `chapterbib`, going against FG, is not supposed to be loaded.

[16] In `hyperref`'s documentation, Sebastian Rahtz admits not having been able to make hyper-bibliography robust, "since many styles redefine these things... Any or all of *achemso*, *chapterbib*, and *drftcite* may break." For the case of `cite`, I tried to make a compromise, sacrificing the interactivity of the bibliographical references to keep the effects of the package. But I got completely lost in the attempt... As Rahtz says, "life is too short," and I am not going to understand all the workings of `\@cite`, `\@citex`, `\@citen`, ... Sigh.

On the other hand, support for the multibib package, that allows multiple lists of references in the same document, is in progress. For the time being, the recommendation is to plan on writing one general bibliography if possible. In any case, several reference lists can be manually created (i.e., without using BibTeX).

### 5.9 The index

The code of pittetd defines the environment theindex to suit the FG, but otherwise exactly as standard classes define it. This means that the production of the index, be it manually or through *MakeIndex*, remains the same. hyperref offers an option to create a 'hyper-index,' whose page numbers are interactive links. However, the option is not very robust, and therefore pittetd uses hyperref but turns hyper-indexing off.

At the moment there is no support for multiple indexes to be generated automatically by *MakeIndex*, although several indexes can be manually created.

## 6 Using the hyperref package

This section is a very brief and incomplete guide to some extra features of the hyperref package that have not been explained before. Unfortunately, if something is missing to hyperref, it is documentation. Useful information is to be found in [2] and [3], but those documents are not intended for the average user. The present section is a translated adaptation of the relevant section in [4], to my knowledge the most complete (but still not comprehensive) user's guide on the package.

Section 6.2 provides a starting point to use hyperref in a way different of pittetd's default.

### 6.1 New user's commands

Certain character strings (notably the text of the bookmarks) are converted by hyperref into ASCII text, ignoring most LaTeX commands. In general, macros that expand into a piece of text (such as the `\LaTeX` command itself, the italic correction `\/`, or things like ` and ') are appropriately handled. But math mode, for example, is completely ignored. The process leads virtually never to an error message; warnings, however, are issued for every ignored token.

`\texorpdfstring` In any case, the user has a way to 'help' `hyperref` in the conversion, namely the command

```
\texorpdfstring{⟨TEX text⟩}{⟨PDF text⟩}
```

that can be used in sectioning commands or captions for figures and tables. For example, a caption with the text 'An $H_2O$ molecule,' that would produce a bad bookmark entry, can be fixed by typing

```
\caption{\texorpdfstring{An H$_2$O molecule}{A water molecule}}
```

After this, the caption for the figure will feature '$H_2O$' (both in the figure and the list of figures), but its bookmark will substitute '`water`'.

To create links other than those produced by the LaTeX commands `\ref`, `\pageref`, and `\cite`, `hyperref` makes available other commands. Only some of them will be mentioned here. See [2] for the rest.

`\nameref` The `\nameref` command works like `ref`, but creates a link with the chapter or section *name*. It is only applicable to sectioning commands.

`\url` The command `\url{`⟨*URL address*⟩`}` prints the ⟨*URL address*⟩ as a link that launches the local Internet surfer and leads to the corresponding page.

`\hypertarget` Analogous to `\label`, the command `\hypertarget{`⟨*key*⟩`}{`⟨*text*⟩`}` makes the ⟨*text*⟩ to be the target of a cross reference.

`\hyperlink` Analogous to `\ref`, the command `\hyperlink{`⟨*key*⟩`}{`⟨*expression*⟩`}` sets up an internal link whose target has been previously defined with `\hypertarget`.

`\Acrobatmenu` Through the command `\Acrobatmenu{`⟨*menu function*⟩`}{`⟨*text*⟩`}`, a ⟨*text*⟩ is typeset as a link that activates the ⟨*menu function*⟩ of Acrobat Reader (or Exchange). For a list of the available functions, see section 4 of [2].

## 6.2 Overriding pittetd's preferences

As has been said, `pittetd` loads `hyperref` with a fixed set of options. In order to access the package keeping control of it, it is needed to specify the `nohyperref` option for `pittetd` and then load `hyperref`:

```
\documentclass[nohyperref]{pittetd}
```

```
\usepackage[⟨personal options⟩]{hyperref}
```

This procedure is of course recommended only to users experienced with hyperref. A comprehensive list of hyperref's options is given in [5]. Here is the list of options that pittetd uses by default (when allowed to):

```
letterpaper, colorlinks,
hyperindex=false
bookmarks, bookmarksnumbered, bookmarksopen,
citecolor=blue, urlcolor=blue
```

An option not used by pittetd that might be relevant is backref, that makes the bibliographical entries produce links to the sections in which the corresponding \cite appear (there is also the alternative pagebackref, with links leading to the *page* of the \cite's).

In any case, it is always good to indicate the driver for hyperref, for example pdftex or dvipdfm, as an option to this package. In fact, when such an option is given to pittetd, all that is done by the latter is to pass it on to packages that need it, including hyperref, graphicx and color.

By loading hyperref manually, some automatic features of pittetd are lost: the bookmarks for the bibliography, the index, and the appendices; and the filling in of the 'Document Info' dialog box of Acrobat Reader. Figures and tables, however, will still create bookmarks. To get those bookmarks created was the thorniest issue in the writing of pittetd, and we have decided to keep this working even if the user has chosen to override pittetd's preferences about hyperref (see the code for \listoffigures and \listoftables).

\pdfbookmark

To create bookmarks additional to those that come from sections in the table of contents (or from the lists of figures and tables), hyperref provides the \pdfbookmark command:

```
\pdfbookmark[⟨level⟩]{⟨bookmark text⟩}{⟨key⟩}
```

where ⟨*level*⟩ is 0 for chapters, 1 for sections, and so on. The ⟨*key*⟩ is a unique name chosen by the user. The bookmark will be appended to the panel in the current position, and will point to the page of the text, in which \pdfbookmark appears. For more complicated instances (bookmarks that lead to a different location in the document, or that lead to different documents), see sections 5.2.4 and 7.2 of [3].

# 7 Before submitting

The pittetd LaTeX class is programmed to follow closely and consistently the FG. In general, the author of a thesis or dissertation needs not to be concerned about most of the formatting requirements (for example, checking the bookmarks and links one by one is unnecessary). However, this creates the danger of implying that nothing can go wrong. There are in fact some things beyond pittetd's control, and those things must be checked by the authors themselves (and will probably be checked closely by format reviewers). This section highlights the most common and likely problems.

**Captions of tables and figures.** Captions for tables should appear at the top of the table, while those for figures go at the bottom. pittetd does not force nor check this requirement.

**Captions as bookmarks.** Very long captions for tables and figures tend to be truncated when converted to bookmarks. Also, LaTeX constructions (like formulas, cite commands, etc.) are lost. Sections 5.6.1 and 6.1 show two ways of dealing with these limitations.

**Capitalization of sections.** The section titles are capitalized by pittetd in the text, but *not* in the bookmarks. The best thing is to provide `\section` with an already-all-capitals argument.

**The `final` option.** Before submitting it is always very important to run the document with `final` option (i.e., adding '`final`' to the list of options to `\documentclass`). This will catch and make evident any problems in the preliminary pages. See section 4.5.

**Bad line breaks.** Sometimes LaTeX cannot break a paragraph into lines satisfactorily. The result is one (or more) 'overfull' lines, that stick to the right of the margin. LaTeX always gives a warning about each and every overfull, and these can be seen in the `.log` file. This file, a plain-text file, can (and should) be read for overfull and other kinds of warnings. Overfull warnings start with the text '`Overfull \hbox in paragraph`'.

**Bad page breaks.** Similarly, LaTeX issues an 'underfull' warning for bad page breaking—when it is able to recognize it. But sometimes LaTeX will break a page just after a heading, which is wrong. The best way to check page breaking is to make the pages fit the screen and go scanning quickly page by page (PgDn).

**Warnings.** There are also warnings about other things, such as incomplete cross references, undefined `\cite`'s, etc., which are important to fix. The warnings are all collected in the `.log` file, and usually reveal at least one problem that had not been noticed before. It is not good to neglect reading this file; getting it to report no problems should be the crowning, final step in the thesis/dissertation production.

## References

[1] Federico Garcia, *Comments on using LaTeX for theses*, July 2003, file `comments.dvi` or `comments.pdf`, available at pittetd's webpage.

[2] Sebastian Rahtz, *Hypertext marks in LaTeX: the hyperref package*, June 1998, file `manual.pdf`, part of the `hyperref` package distribution. Available at pittetd's webpage.

[3] Heiko Oberdiek, *PDF information and navigation elements with hyperref, pdfTeX, and thumbpdf*, paper at EuroTeX'99. File `paper.pdf`, part of the `hyperref` package distribution. Available at pittetd's webpage.

[4] Rodrigo De Castro, *El Universo LaTeX*, 2nd. edition, Bogotá, Universidad Nacional de Colombia, 2003.

[5] Sebastian Rahtz, *hyperref package options*, October 1999, file `options.pdf`, part of the `hyperref` package distribution. Available at pittetd's webpage.

## Index

Numbers written in italics refer to the page where the corresponding entry is described. A list of mentioned packages is found under 'packages mentioned;' of environments under 'environments;' of pittetd options under 'options to pittetd.'