More Math Into LaTeX

4th Edition

George Grätzer

More Math Into LaTeX

4th Edition

Foreword by
Rainer Schöpf
LaTeX3 team

Springer

George Grätzer
Department of Mathematics
University of Manitoba
Winnipeg, MB R3T 2N2
Canada
gratzer@ms.umanitoba.ca

Cover design by Mary Burgess.
Typeset by the author in LaTeX.

Library of Congress Control Number: 2007923503

ISBN-13: 978-387-32289-6 e-ISBN-13: 978-0-387-68852-7

Printed on acid-free paper.

©2007 Springer Science+Business Media, LLC
All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media LLC, 233 Spring Street, New York, NY 10013, USA) and the author, except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.
The use in this publication of trade names, trademarks, service marks and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

9 8 7 6 5 4 3 2 1

springer.com (HP)

To the **Volunteers**
without whose dedication over 15 years,
this book could not have been done

and to my four grandchildren
Danny (11),

Anna (8),

Emma (2),

and **Kate** (0)

Short Contents

Contents

Foreword

It was the autumn of 1989—a few weeks before the Berlin wall came down, President George H. W. Bush was president, and the American Mathematical Society decided to outsource TEX programming to Frank Mittelbach and me.

Why did the AMS outsource TEX programming to us? This was, after all, a decade before the words "outsourcing" and "off-shore" entered the lexicon. There were many American TEX experts. Why turn elsewhere?

For a number of years, the AMS tried to port the mathematical typesetting features of $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX to LATEX, but they made little progress with the AMSFonts. Frank and I had just published the New Font Selection Scheme for LATEX, which went a long way to satisfy what they wanted to accomplish. So it was logical that the AMS turned to us to add AMSFonts to LATEX. Being young and enthusiastic, we convinced the AMS that the $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX commands should be changed to conform to the LATEX standards. Michael Downes was assigned as our AMS contact; his insight was a tremendous help.

We already had LATEX-NFSS, which could be run in two modes: compatible with the old LATEX or enabled with the new font features. We added the reworked $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX code to LATEX-NFSS, thus giving birth to $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX, released by the AMS at the August 1990 meeting of the International Mathematical Union in Kyoto.

$\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX was another variant of LATEX. Many installations had several LATEX variants to satisfy the needs of their users: with old and new font changing commands, with and without $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX, a single and a multi-language version. We decided to develop a Standard LATEX that would reconcile all the variants. Out of a group of interested people grew what was later called the *LATEX3 team*—and the LATEX3 project got underway. The team's first major accomplishment was the release of LATEX 2_ε in June 1994. This standard LATEX incorporates all the improvements we wanted back in 1989. It is now very stable and it is uniformly used.

Under the direction of Michael Downes, our $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LATEX code was turned into AMS packages that run under LATEX just like other packages. Of course, the LATEX3

team recognizes that these are special; we call them "required packages" because they are part and parcel of a mathematician's standard toolbox.

Since then a lot has been achieved to make an author's task easier. A tremendous number of additional packages are available today. The *LaTeX Companion,* 2nd edition, describes many of my favorite packages.

George Grätzer got involved with these developments in 1990, when he got his copy of $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX in Kyoto. The documentation he received explained that $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX is a LaTeX variant—read Lamport's LaTeX book to get the proper background. $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX is not $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX either—read Spivak's $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX book to get the proper background. The rest of the document explained in what way $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX differs from LaTeX and $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX. Talk about a steep learning curve ...

Luckily, George's frustration working through this nightmare was eased by a lengthy e-mail correspondence with Frank and lots of telephone calls to Michael. Three years of labor turned into his first book on LaTeX, providing a "simple introduction to $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX".

This fourth edition is more mature, but preserves what made his first book such a success. Just as in the first book, Part I is a short introduction for the beginner, dramatically reducing the steep learning curve of a few weeks to a few hours. The rest of the book is a detailed presentation of what you may need to know. George "teaches by example". You find in this book many illustrations of even the simplest concepts. For articles, he presents the LaTeX source file and the typeset result side-by-side. For formulas, he discusses the building blocks with examples, presents a *Formula Gallery,* and a *Visual Guide* to multiline formulas.

Going forth and creating "masterpieces of the typesetting art"—as Donald Knuth put it at the end of the *TeXbook*—requires a fair bit of initiation. This is the book for the LaTeX beginner as well as for the advanced user. You just start at a different point.

The topics covered include everything you need for mathematical publishing.

- Starting from scratch, by installing and running LaTeX on your own computer
- Instructions on creating articles, from the simple to the complex
- Converting an article to a presentation
- Customize LaTeX to your own needs
- The secrets of writing a book
- Where to turn to get more information or to download updates

The many examples are complemented by a number of easily recognizable features:

Rules which you must follow

Tips on how to achieve some specific results

Experiments to show what happens when you make mistakes—sometimes, it can be difficult to understand what went wrong when all you see is an obscure LaTeX error message

This book teaches you how to convert your mathematical masterpieces into typographical ones, giving you a lot of useful advice on the way. How to avoid the traps for the unwary and how to make your editor happy. And hopefully, you'll experience the fascination of doing it right. Using good typography to better express your ideas.

If you want to learn LaTeX, buy this book and start with the *Short Course*. If you can have only one book on LaTeX next to your computer, this is the one to have. And if you want to learn about the world of LaTeX packages, also buy a second book, the *LaTeX Companion,* 2nd edition.

Rainer Schöpf
LaTeX3 team

Preface to the Fourth Edition

This is my fourth full-sized book on LaTeX.

The first book, *Math into TeX: A Simple Introduction to AMS-LaTeX* [19], written in 1991 and 1992, introduced the brand new AMS-LaTeX, a LaTeX variant not compatible with the LaTeX of the time, LaTeX 2.09. It brought together the features of LaTeX and the math typesetting abilities of AMS-TeX, the AMS typesetting language.

The second book, *Math into LaTeX: An Introduction to LaTeX and AMS-LaTeX* [27], written in 1995, describes the new LaTeX introduced by the LaTeX3 team and the AMS typesetting features implemented as extensions of LaTeX, called packages.

The third book, *Math into LaTeX*, 3rd edition [30], published in 2000, reports on the same system. By 2000, both the "new" LaTeX and the AMS packages were quite mature. The feverish debugging of the new LaTeX every six months bore fruit. LaTeX became very stable. It has changed little since 2000. Version 2.0 of the AMS packages was released and it also became very stable. The third book reports on a rock solid typesetting system.

What also changed between 1995 and 2000 is the widespread use of the Internet. Several chapters of the third book deal with the impact of the Internet on mathematical publications.

Now, seven years later, we can still report that LaTeX—no longer new—and the AMS packages have changed very little. However, the impact of the Internet became even more important. Computers also changed. They are now much more powerful. When I started typesetting math with LaTeX, it took two and a half minutes to typeset a page. This book takes 1.8 seconds to typeset on my computer, a Mac desktop from 2006. As a result, we do not have to be very selective in what we load into memory; we can load everything we may possibly need.

Circumincession

So this is the first big change compared to the previous books. In this book, we roll TEX, LATEX, and the AMS packages into one, and we call it simply LATEX. This results in a great simplification in the exposition and makes the learning curve a little less steep.

I am sure with some advanced users this will prove to be a controversial decision. They want to know where a command is defined. For the beginner and the non-expert user this does not make any difference. What matters is that the command they need be available when they need it.

From the beginner's point of view, this approach is very beneficial. Take as an example the `\text` command. In all three of my books, we first introduce the LATEX command `\mbox` for typing text in math formulas. After half a page of discussion comes the sentence: "It is better to enter text in formulas with the `\text` command provided by the `amsmath` package." Then another half page discusses the command `\text`. In this book, we ignore `\mbox` and go right-away to `\text`. You do not have to do anything to access the command, the `amsmath` package is always loaded for you.

And what to do if you want to find out where a command is defined. Now for both the PC and the Mac, you can easily search for contents of files. Do you want to know where a command is defined? Search for it and it is easy to find the file in which it is introduced.

Presentations

The second big change is the widespread acceptance of the Adobe PDF format. As a result, the majority of the lectures today at math meetings are given as *presentations*, PDF files projected to screens using computers. Blackboards and whiteboards have largely disappeared and computer projections are overtaking projectors. So this book takes up presentations as a major topic, introducing it in Part I and discussing it in detail in Chapter 14.

Installations

In the third book, I report a recurring question that comes up from my readers again and again:

Can you help me get started from scratch, covering everything from installing a working LATEX system to the rudiments of text editing?

And here is the third big change that has happened in the last few years. While earlier there were dozens of different LATEX implementations and hundreds of text editors, today most PC users use `MiKTeX` with the text editor/front end `WinEdt` and most Mac users use TEX Live with the text editor/front end `TeXShop`. So if you want help to

install LaTeX, it is easy for me to help you. Appendix A provides instructions on how to install these systems.

Acknowledgments

This book is based, of course, on the three previous books. I would like to thank the many people who read and reread those earlier manuscripts.

The editors Richard Ribstein, Thomas R. Scavo, Claire M. Connelly.

The professionals Michael Downes (the project leader for the AMS), Frank Mittelbach and David Carlisle (of the LaTeX3 team) read and criticized some or all of the three books.

Oren Patashnik (the author of BibTeX) carefully corrected the BibTeX chapter for two editions.

Sebastian Rahtz (the author of the `hyperref` package and coauthor of *The LaTeX Web Companion* [18]) read the chapter on the Web in the third book.

Last but not least, Barbara Beeton of the AMS read all three books with incredible insight.

The volunteers for the second book alone, there were 29—listed there. The volunteer readers made tremendous contributions and offered hundreds of pages of corrections. No expert can substitute for the diverse points of view I got from them.

My colleagues especially Michael Doob, Harry Lakser, and Craig Platt, who have been very generous with their time.

The publishers Edwin Beschler, who believed in the project from the very beginning and guided it through a decade and Ann Kostant who continued Edwin's work.

For this book, I have had the most talented and thorough group of readers ever: Andrew Adler of the University of British Columbia, Canada, Joseph Maria Font of the University of Barcelona, Spain, and Alan Litchfield, of the Auckland University of Technology, New Zealand. Chapter 14 was read by David Derbes, Adam Goldstein, Mark Eli Kalderon, Michael Kubovy, Matthieu Masquelet, and Charilaos Skiadas—and Chapter 15 by Ross Moore. Interestingly, only half of them are mathematicians, the rest are philosophers, linguists, and so on. Appendix A.1 was read by Brian Davey and Appendix A.2 by Richard Koch (the author of `TeXShop`).

The fourth edition was edited by Barbara Beeton, Edwin Beschler, and Clay Martin with Ann Kostant as the Springer editor. The roles of Edwin and Ann have changed, but not the importance of their contributions. The index was compiled with painstaking precision by Laura Kirkland. Barbara Beeton also provided a number of intriguing illustrations of quaint commands. My indebtedness to her cannot be overstated.

George Grätzer

Introduction

Is this book for you?

This book is for the mathematician, physicist, engineer, scientist, linguist, or technical typist who has to learn how to typeset articles containing mathematical formulas or diacritical marks. It teaches you how to use LaTeX, a typesetting markup language based on Donald E. Knuth's typesetting language TeX, designed and implemented by Leslie Lamport, and greatly improved by the AMS.

Part I provides a quick introduction to LaTeX, from typing examples of text and math to typing your first article (such as the sample article on pages 42–43) and creating your first presentation (such as the sample presentation on pages 57–58) in a very short time. The rest of the book provides a detailed exposition of LaTeX.

LaTeX has a huge collection of rules and commands. While the basics in Part I should serve you well in all your writings, most articles and presentations also require you to look up special topics. Learn Part I well and become passingly familiar enough with the rest of the book, so when the need arises you know where to turn with your problems.

You can find specific topics in one or more of the following sources: the Short Contents, the detailed Contents, and the Index.

What is document markup?

When you work with a word processor, you see your document on the computer monitor more or less as it looks when printed, with its various fonts, font sizes, font shapes (e.g., roman, italic) and weights (e.g., normal, boldface), interline spacing, indentation, and so on.

Working with a *markup language* is different. You type the *source file* of your article in a *text editor,* in which all characters appear in the same font. To indicate changes in the typeset text, you must add *text markup commands* to the source file.

For instance, to emphasize the phrase `detailed description` in a LaTeX source file, type

```
\emph{detailed description}
```

The `\emph` command is a markup command. The marked-up text yields the typeset output

detailed description

In order to typeset math, you need *math markup commands*. As a simple example, you may need the formula $\int \sqrt{\alpha^2 + x^2}\, dx$ in an article you are writing. To mark up this formula in LaTeX, type

```
$\int \sqrt{\alpha^{2} + x^{2}}\,dx$
```

You do not have to worry about determining the size of the integral symbol or how to construct the square root symbol that covers $\alpha^2 + x^2$. LaTeX does it all for you.

On pages 290–293, I juxtapose the source file for a sample article with the typeset version. The markup in the source file may appear somewhat challenging at first, but I think you agree that the typeset article is a pleasing rendering of the original input.

The three layers

The markup language we shall discuss comes in three layers: TeX, LaTeX, and the AMS packages, described in detail in Appendix D. Most LaTeX installations—including the two covered in Appendix A—automatically place all three on your computer. You do not have to know what comes from which layer, so we consider the three together and call it LaTeX.

The three platforms

Most of you run LaTeX on one of the following three computer types:

- A PC, a computer running Microsoft Windows
- A Mac[1], a Macintosh computer running OS X
- A computer running a UNIX variant such as Solaris or Linux

The LaTeX source file and the typeset version both look the same independent of what computer you have. However, the way you type your source file, the way you typeset it, and the way you look at the typeset version depends on the computer and on the LaTeX implementation you use. In Appendix A, we show you how to install LaTeX for a PC and a Mac. Many UNIX systems come with LaTeX installed.

[1] In the old days, I used to run TEXTURES under OS 9. Unfortunately, TEXTURES does not run on new Intel Macs.

What's in the book?

Part I is the *Short Course;* it helps you to get started quickly with LaTeX, to type your first articles, to prepare your first presentations, and it prepares you to tackle LaTeX in more depth in the subsequent parts. We assume here that LaTeX is installed on your computer. If it is not, jump to Appendix A.

Chapter 1 introduces the *terminology* we need to talk about your LaTeX implementations. **Chapter 2** introduces how LaTeX uses the *keyboard* and how to *type text*. You do not need to learn much to understand the basics. Text markup is quite easy. You learn math markup—which is not so straightforward—in **Chapter 3**. Several sections in this chapter ease you into *mathematical typesetting.* There is a section on the basic building blocks of math formulas. Another one discusses equations. Finally, we present the two simplest multiline formulas, which, however, cover most of your everyday needs.

In **Chapter 4**, you start writing your *first article* and prepare your *first presentation*. A LaTeX article is introduced with the sample article `intrart.tex`. We analyze in detail its structure and its source file, and we look at the typeset version. Based on this, we prepare an article template, and you are ready for your first article. A quick conversion of the article `intrart.tex` to a presentation introduces this important topic.

Part II introduces the two most basic skills for writing with LaTeX in depth, *typing text* and *typing math.*

Chapters 5 and **6** introduce *text* and *displayed text.* Chapter 5 is especially important because, when you type a LaTeX document, most of your time is spent typing text. The topics covered include special characters and accents, hyphenation, fonts, and spacing. Chapter 6 covers displayed text, including *lists* and *tables,* and for the mathematician, *proclamations* (theorem-like structures) and *proofs*.

Typing math is the heart of any mathematical typesetting system. **Chapter 7** discusses inline formulas in detail, including basic constructs, delimiters, operators, math accents, and horizontally stretchable lines. The chapter concludes with the *Formula Gallery*.

Math symbols are covered in three sections in **Chapter 8**. How to space them, how to build new ones. We also look at the closely related subjects of math alphabets and fonts. Then we discuss tagging and grouping equations.

LaTeX knows a lot about typesetting an inline formula, but not much about how to display a multiline formula. **Chapter 9** presents the numerous tools LaTeX offers to help you do that. We start with a *Visual Guide* to help you get oriented.

Part III discusses the parts of a LaTeX document. In **Chapter 10**, you learn about the *structure* of a LaTeX document. The most important topics are *sectioning* and *cross-referencing*. In **Chapter 11**, we discuss the `amsart` *document class* for articles. In particular, I present the title page information. Chapter 11 also features `sampart.tex`, a sample article for `amsart`, first in typeset form, then in mixed form, juxtaposing the source file and the typeset article. You can learn a lot about LaTeX just by reading the source file one paragraph at a time and seeing how that paragraph is typeset. We con-

clude this chapter with a brief description of the AMS distribution, the packages and document classes, of which `amsart` is a part.

In **Chapter 12** the most commonly used *legacy document classes* are presented, `article`, `report`, and `letter` (the `book` class is discussed in Chapter 18), along with a description of the standard LaTeX distribution. Although `article` is not as sophisticated as `amsart`, it is commonly used for articles not meant for publication.

In **Part IV**, we start with **Chapter 13**, discussing PDF *files, hyperlinks,* and the `hyperref` package. This prepares you for *presentations,* which are PDF files with hyperlinks. In **Chapter 14** we utilize the `beamer` *package* for making LaTeX presentations.

Part V (**Chapter 15**) introduces techniques to *customize* LaTeX: user-defined commands, user-defined environments, and command files. We present a sample command file, `newlattice.sty`, and a version of the sample article utilizing this command file. You learn how parameters that affect LaTeX's behavior are stored in counters and length commands, how to change them, and how to design your own custom lists. A final section discusses the pitfalls of customization.

In **Part VI** (**Chapters 16** and **17**), we discuss the special needs of longer documents. Two applications, contained in the standard LaTeX distribution, BibTeX and *MakeIndex*, make compiling *large bibliographies* and *indexes* much easier.

LaTeX provides the `book` and the `amsbook` document classes to serve as foundations for well-designed books. We discuss these in **Chapter 18**. Better quality books have to use document classes designed by professionals. We provide some sample pages from a book using Springer's `svmono.cls` document class.

Detailed instructions are given in **Appendix A** on how to install LaTeX on a PC and a Mac. On a PC we install `WinEdt` and `MiKTeX`. On a Mac, we install `MacTeX`, which consists of TeX Live and `TeXShop`. For both installations, we describe the editing cycle and three productivity tools in sufficient detail so that you be able to handle the tasks on the sample files of the *Short Course*.

You will probably find yourself referring to **Appendices B** and **C** time and again. They contain the *math and text symbol tables.*

Appendix D relates some historical background material on LaTeX. It gives you some insight into how LaTeX developed and how it works. **Appendix E** discusses the many ways we can find LaTeX material on the *Internet.*

Appendix F is a brief introduction to the use of *PostScript fonts* in a LaTeX document. **Appendix G** briefly describes the use of LaTeX for languages other than American English.

Finally, **Appendix H** discusses what we left out and points you towards some areas for further reading.

Mission statement

This book is a guide for typesetting mathematical documents within the constraints imposed by LaTeX, an elaborate system with hundreds of rules. LaTeX allows you to perform almost any mathematical typesetting task through the appropriate application of its rules. You can customize LaTeX by introducing user-defined commands and environments and by changing LaTeX parameters. You can also extend LaTeX by invoking packages that accomplish special tasks.

It is *not my goal*

- to survey the hundreds of LaTeX packages you can utilize to enhance LaTeX
- to teach how to write TeX code and to create your own packages
- to discuss how to design beautiful documents by writing document classes

The definitive book on the first topic is Frank Mittelbach and Michel Goosens's *The LaTeX Companion,* 2nd edition [46] (with Johannes Braams, David Carlisle, and Chris Rowley). The second and third topics still await authoritative treatment.

Conventions

To make this book easy to read, I use some simple conventions:

- Explanatory text is set in this typeface: Times.
- `Computer Modern typewriter is used to show what you should type, as well as messages from LaTeX. All the characters in this typeface have the same width, making it easy to recognize.`
- I also use Computer Modern typewriter to indicate
  - Commands (`\parbox`)
  - Environments (`\align`)
  - Documents (`intrart.tex`)
  - Document classes (`amsart`)
  - Document class options (`draft`)
  - Folders or directories (`work`)
  - The names of *packages,* which are extensions of LaTeX (`verbatim`)
- When I show you how something looks when typeset, I use Computer Modern, TeX's standard typeface:

I think you find this typeface sufficiently different from the other typefaces I have used. The strokes are much lighter so that you should not have much difficulty recognizing typeset LaTeX material. When the typeset material is a separate paragraph or paragraphs, corner brackets in the margin set it off from the rest of the text—unless it is a displayed formula.

- For explanations in the text, such as

  Compare iff with iff, typed as `iff` and `if{f}`, respectively.

  the same typefaces are used. Because they are not set off spatially, it may be a little more difficult to see that iff is set in Computer Modern roman (in Times, it looks like this: iff), whereas `iff` is set in the Computer Modern typewriter typeface.

- I usually introduce commands with examples, such as

  `\\[22pt]`

  However, it is sometimes necessary to define the syntax of a command more formally. For instance,

  `\\[`*`length`*`]`

  where *length*, typeset in Computer Modern typewriter italic font, represents the value you have to supply.

Good luck and have fun.

E-mail:
`gratzer@ms.umanitoba.ca`
Home page:
`http://www.maths.umanitoba.ca/homepages/gratzer.html`

PART I

Short Course

CHAPTER

1

Your LaTeX

Are you sitting in front of your computer, your LaTeX implementation up and running? In this chapter we get you ready to tackle this *Short Course*. When you are done with Part I, you will be ready to start writing your articles in LaTeX.

If you do not have a LaTeX implementation up and running, go to Appendix A. There you find precise and detailed instructions how to set up LaTeX on a PC or a Mac. There is enough in the appendix for you to be able to handle the tasks in this *Short Course*. You will be pleasantly surprised at how little time it takes to set LaTeX up. If you use some variant of UNIX, turn to a UNIX guru who can help you set up LaTeX on your computer and guide you through the basics. If all else fails, read the documentation for your UNIX system.

1.1 Your computer

We assume very little, only that you are familiar with your keyboard and with the operating system on your computer. You should know standard PC and Mac menus, pull down menus, buttons, tabs, the menu items, such as `Edit>Paste`, the menu item `Paste` on the menu `Edit`. You should understand folders (we use this terminology regardless of the platform, with apologies to our UNIX readers), and you need to know how to save a file and copy a file from one folder to another.

On a PC, `work\test` refers to the subfolder `test` of the folder `work`. On a Mac, `work/test` designates this subfolder. To avoid having to write every subfolder twice, we use `work/test`, with apologies to our PC readers.

1.2 Sample files

We work with a few sample documents in this *Short Course*. You can type the sample documents as presented in the text, or you can download them from the Internet (see Section E.1). The `samples` folder also contains a copy of `SymbolTables.pdf`, a PDF version of Appendices B and C, the symbol tables.

I suggest you create a folder on your computer named `samples`, to store the downloaded sample files, and another folder called `work`, where you will keep your working files. Copy the documents from the `samples` to the `work` folder as needed. *In this book, the* `samples` *and* `work` *folders refer to the folders you have created.*

If you `Save As...` a sample file under a different name, remember the naming rule.

Rule ■ Naming of source files

The name of a LaTeX source file should be *one word* (no spaces, no special characters), and end with `.tex`.

So `first art.tex` is bad, but `art1.tex` and `FirstArt.tex` are good.

1.3 Editing cycle

Watch a friend type a mathematical article in LaTeX and you learn some basic steps.

1. *A text editor is used to create a LaTeX source file*. A source file might look like the top window in Figure 1.1:

```
\documentclass{amsart}
\begin{document}
The hypotenuse: $\sqrt{a^{2} + b^{2}}$. I can type math!
\end{document}
```

   Note that the source file is different from a typical word processor file. All characters are displayed in the same font and size.

2. *Your friend "typesets" the source file* (*tells the application to produce a typeset version*) *and views the result on the monitor* (the two corners indicate material typeset by LaTeX):

The hypotenuse: $\sqrt{a^2 + b^2}$. I can type math!

as in the middle window in Figure 1.1.

3. *The editing cycle continues.* Your friend goes back and forth between the source file and the typeset version, making changes and observing the results of these changes.

4. *The file is printed.* Once the typeset version is satisfactory, it is printed, creating a paper version of the typeset article. Alternatively, your friend creates a PDF file of the typeset version (see Chapter 13.1.2).

If LaTeX finds a mistake when typesetting the source file, it opens a new window, the *log window,* illustrated as the bottom window in Figure 1.1, and displays an error message. The same message is saved into a file, called the *log file*. Look at the figures in Appendix A, depicting a variety of editing windows, windows for the typeset article, and log windows for the two LaTeX implementations discussed there.

Various LaTeX implementations have different names for the source file, the text editor, the typeset file, the typeset window, the log window, and the log file. Become familiar with these names for the LaTeX implementation you use, so you can follow along with our discussions. In Appendix A, we bring you up to speed for the LaTeX implementations discussed therein.

1.4 Three productivity tools

Most LaTeX implementations have these important productivity tools:

Synchronization To move quickly between the source file and the typeset file, most LaTeX implementations offer *synchronization*, the ability to jump from the typeset

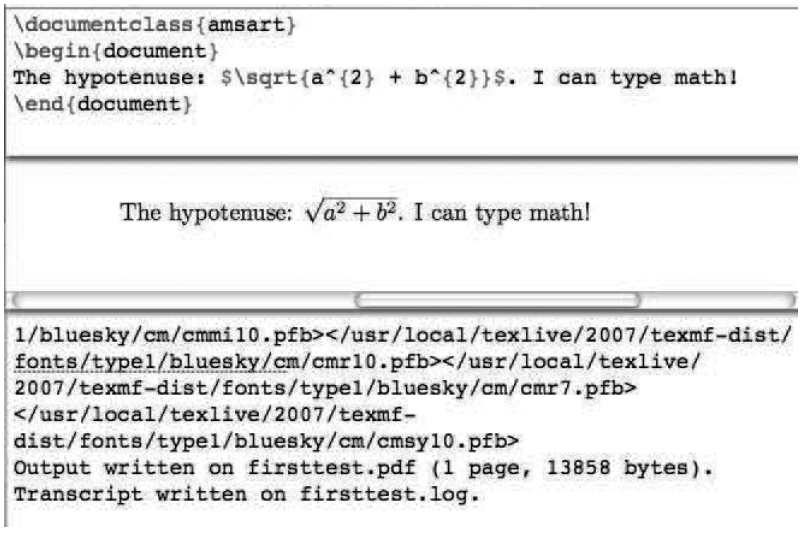


Figure 1.1: Windows for the source and typeset files and the log window.

file to the corresponding place in the source file and from the source file to the corresponding place in the typeset file.

Block comment Block comments are very useful:

1. When looking for a LaTeX error, you may want LaTeX to ignore a block of text in the source file (see page 51).

2. Often you may want to make comments about your project but not have them printed or you may want to keep text on hand while you try a different option. To accomplish this, insert a comment character, %, at the start of each line where the text appears. These lines are ignored when the LaTeX file is processed.

   Select a number of lines in a source document, then by choosing a menu option all the lines (the whole block) are commented out (a % sign is placed at the beginning of each line). This is *block comment*. The reverse is *block uncomment*.

Jump to a line This is specified by the line number in the source file. To find an error, LaTeX suggests that you jump to a line.

Find out how your LaTeX implements these features. In Appendix A, we discuss how these features are implemented for the LaTeX we install.

Pay careful attention how your LaTeX implementation works. This enables you to rapidly perform the editing cycle and utilize the productivity tools when necessary.

CHAPTER

2

Typing text

In this chapter, I introduce you to typesetting text by working through examples. More details are provided throughout the book, in particular, in Chapters 5 and 6.

A source file is made up of *text, math* (*formulas*), and *instructions* (*commands*) *to* LaTeX. For instance, consider the following variant of the first sentence of this paragraph:

```
A source file is made up of text, math (e.g.,
$\sqrt{5}$), and \emph{instructions to} \LaTeX.
```

This typesets as

A source file is made up of text, math (e.g., $\sqrt{5}$), and *instructions to* LaTeX.

In this sentence, the first part

```
A source file is made up of text, math (e.g.,
```

is text. Then

```
$\sqrt{5}$
```

is math

```
), and
```

is text again. Finally,

```
\emph{instructions to} \LaTeX.
```

are instructions. The instruction `\emph` is a *command with an argument,* while the instruction `\LaTeX` is a *command without an argument.*

Commands, as a rule, start with a backslash (`\`) and tell LaTeX to do something special. In this case, the command `\emph` emphasizes its *argument* (the text between the braces). Another kind of instruction to LaTeX is called an *environment.* For instance, the commands

```
\begin{flushright}
```

and

```
\end{flushright}
```

enclose a `flushright` environment; the *content*, that is, the text that is typed between these two commands, is right justified (lined up against the right margin) when typeset. (The `flushleft` environment creates left justified text; the `center` environment creates text that is centered horizontally on the page.)

In practice, text, math, and instructions (commands) are mixed. For example,

```
My first integral: $\int \zeta^{2}(x) \, dx$.
```

is a mixture of all three; it typesets as

My first integral: $\int \zeta^2(x)\,dx$.

Creating a document in LaTeX requires that we type the text and math in the source file. So we start with the keyboard, proceed to type a short note, and learn some simple rules for typing text in LaTeX.

2.1 The keyboard

The following keys are used to type text in a source file:

```
a-z   A-Z   0-9
+ = * / ( ) [ ]
```

You may also use the following punctuation marks:

```
, ; . ? ! : ‘ ’ -
```

and the space bar, the Tab key, and the Return (or Enter) key.

Since TEX source files are "pure text" (ASCII files), they are very portable. There is one possible problem limiting this portability, the line endings used in the source file. When you press the Return key, your text editor writes an invisible code into your source file that indicates where the line ends. Since this code may be different on different platforms (PC, Mac, and UNIX), you may have problems reading a source file created on a different platform. Luckily, many text editors include the ability to switch end-of-line codes and some, including the editors in `WinEdt` and `TeXShop`, do so automatically.

Finally, there are thirteen special keys that are mostly used in LATEX commands:

```
# $ % & ~ _ ^ \ { } @ " |
```

If you need to have these characters typeset in your document, there are commands to produce them. For instance, $ is typed as `\$`, the underscore, _, is typed as `\_`, and % is typed as `\%`. Only @ requires no special command, type `@` to print @. There are also commands to produce composite characters, such as accented characters, for example ä, which is typed as `\"{a}`. See Section 5.4.4 for a complete discussion of symbols not available directly from the keyboard and Appendix C for the text symbol tables. Appendices B and C are reproduced in the `samples` folder as a PDF file, `SymbolTables.pdf`.

LATEX prohibits the use of other keys on your keyboard—unless you are using a version of LATEX that is set up to work with non-English languages (see Appendix G). When trying to typeset a source file that contains a prohibited character, LATEX displays an error message similar to the following:

```
! Text line contains an invalid character.
l.222 completely irreducible^^?
                                ^^?
```

In this message, `l.222` means line 222 of your source file. You must edit that line to remove the character that LATEX cannot understand. The `log` file (see Section D.3.4) also contains this message. For more about LATEX error messages, see Sections 3.2 and 4.3.1.

2.2 Your first note

We start our discussion on how to type a note in LATEX with a simple example. Suppose you want to use LATEX to produce the following:

It is of some concern to me that the terminology used in multi-section math courses is not uniform.

In several sections of the course on matrix theory, the term "hamiltonian-reduced" is used. I, personally, would rather call these "hyper-simple". I invite others to comment on this problem.

Of special concern to me is the terminology in the course by Prof. Rudi Hochschwabauer. Since his field is new, there is no accepted terminology. It is imperative that we arrive at a satisfactory solution.

To produce this typeset document, create a new file in your `work` folder with the name `note1.tex`. Type the following, including the spacing and linebreaks shown, but not the line numbers:

```
 1  % Sample file: note1.tex
 2  \documentclass{sample}
 3
 4  \begin{document}
 5  It is of some concern to me   that
 6  the terminology used in  multi-section
 7   math courses is not uniform.
 8
 9  In several sections of the course on
10  matrix theory, the  term
11   ``hamiltonian-reduced'' is used.
12   I, personally, would rather call these
13  ``hyper-simple''. I invite others
14   to comment on this  problem.
15
16  Of special concern to me is the terminology
17  in the course by Prof.~Rudi Hochschwabauer.
18  Since his field is new, there is no accepted
19  terminology.   It is imperative
20  that we arrive at a satisfactory solution.
21  \end{document}
```

Alternatively, copy the `note1.tex` file from the `samples` folder (see page 4). Make sure that `sample.cls` is in your `work` folder.

The first line of `note1.tex` starts with `%`. Such lines are called *comments* and are ignored by LaTeX. Commenting is very useful. For example, if you want to add some notes to your source file and you do not want those notes to appear in the typeset version of your article, you can begin those lines with a `%`. You can also comment out part of a line:

```
simply put, we believe % actually, it's not so simple
```

Everything on the line after the % character is ignored by LaTeX.

Line 2 specifies the *document class* (in our case, `sample`)[1] that controls how the document is formatted.

The text of the note is typed within the `document` environment, that is, between the lines

```
\begin{document}
```

and

```
\end{document}
```

Now typeset `note1.tex`. If you use `WinEdt`, click on the `TeXify` icon. If you use `TeXShop`, click the `Typeset` button. You should get the typeset document as shown on page 10. As you can see from this example, LaTeX is different from a word processor. It disregards the way you input and position the text, and follows only the formatting instructions given by the markup commands. LaTeX notices when you put a blank space in the text, but it ignores *how many blank spaces* have been inserted. LaTeX does not distinguish between a blank space (hitting the space bar), a tab (hitting the Tab key), and a *single* carriage return (hitting Return once). However, hitting Return twice gives a blank line; *one or more* blank lines mark the end of a paragraph.

LaTeX, by default, fully justifies text by placing a flexible amount of space between words—the *interword space*—and a somewhat larger space between sentences—the *intersentence space*. If you have to force an interword space, you can use the `\␣` command (in LaTeX books, we use the symbol ␣ to mean a blank space). See Section 5.2.2 for a full discussion.

The `~` (tilde) command also forces an interword space, but with a difference; it keeps the words on the same line. This command is called a *tie* or *nonbreakable space* (see Section 5.4.3).

Note that on lines 11 and 13, the left double quotes are typed as ``` `` ``` (two left single quotes) and the right double quotes are typed as `''` (two right single quotes or apostrophes). The left single quote key is not always easy to find. On an American keyboard,[2] it is usually hidden in the upper-left or upper-right corner of the keyboard, and shares a key with the tilde (`~`).

[1] I know you have never heard of the `sample` document class. It is a special class created for these exercises. You can find it in the `samples` folder (see page 4). If you have not yet copied it over to the `work` folder, do so now.

[2] The location of special keys on the keyboard depends on the country where the computer was sold. It also depends on whether the computer is a PC or a Mac. In addition, notebooks tend to have fewer keys than desktop computers. Fun assignment: Find the tilde (`~`) on a Spanish and on a Hungarian keyboard.

2.3 Lines too wide

LaTeX reads the text in the source file one line at a time and when the end of a paragraph is reached, LaTeX typesets the entire paragraph. Occasionally, LaTeX gets into trouble when trying to split the paragraph into typeset lines. To illustrate this situation, modify `note1.tex`. In the second sentence, replace `term` by `strange term` and in the fourth sentence, delete `Rudi␣`, including the blank space following `Rudi`. Now save this modified file in your `work` folder using the name `note1b.tex`. You can also find `note1b.tex` in the `samples` folder (see page 4).

Typesetting `note1b.tex`, you obtain the following:

It is of some concern to me that the terminology used in multi-section math courses is not uniform.

In several sections of the course on matrix theory, the strange term "hamiltonian-reduced" is used. I, personally, would rather call these "hyper-simple". I invite others to comment on this problem.

Of special concern to me is the terminology in the course by Prof. Hochschwabauer. Since his field is new, there is no accepted terminology. It is imperative that we arrive at a satisfactory solution.

The first line of paragraph two is about $1/4$ inch too wide. The first line of paragraph three is even wider. In the `log` window, LaTeX displays the following messages:

```
 Overfull \hbox (15.38948pt
too wide) in paragraph at lines 9--15 []\OT1/cmr/m/n/10 In sev-eral
sec-tions of the course on ma-trix the-ory, the strange term
‘‘hamiltonian-

Overfull \hbox (23.27834pt too wide) in paragraph
at lines 16--21
[]\OT1/cmr/m/n/10 Of spe-cial con-cern to me is the
ter-mi-nol-ogy in the course by Prof. Hochschwabauer.
```

You will find the same messages in the `log` file (see Sections 1.3 and D.2.1).

The first message,

```
Overfull \hbox (15.38948pt too wide) in paragraph
at lines 9--15
```

refers to the second paragraph (lines 9–15 in the source file—its location in the typeset document is not specified). The typeset version of this paragraph has a line that is 15.38948 points too wide. LaTeX uses *points* (pt) to measure distances; there are about 72 points in 1 inch (or about 28 points in 1 cm).

The next two lines,

```
[]\OT1/cmr/m/n/10 In sev-eral sec-tions of the course
on ma-trix
the-ory, the strange term ‘‘hamiltonian-
```

identify the source of the problem: LaTeX did not properly hyphenate the word

```
hamiltonian-reduced
```

because it (automatically) hyphenates a hyphenated word *only at the hyphen.*

The second reference,

```
Overfull \hbox (23.27834pt too wide) in paragraph
at lines 16--21
```

is to the third paragraph (lines 16–21 of the source file). There is a problem with the word `Hochschwabauer`; LaTeX's standard hyphenation routine cannot handle it (a German hyphenation routine would have no difficulty hyphenating this name—see Appendix G). If you encounter such a problem, you can either try to reword the sentence or insert one or more *optional* (*or discretionary*) *hyphen commands* (`\-`), which tell LaTeX where it may hyphenate the word. In this case, you can rewrite `Hochschwabauer` as `Hoch\-schwa\-bauer` and the second hyphenation problem disappears. You can also utilize the `\hyphenation` command (see Section 5.4.9).

Sometimes a small horizontal overflow can be difficult to spot. The `draft` document class option may help (see Sections 11.5, 12.1.2, and 18.1 for more about document class options). LaTeX places a black box (or *slug*) in the margin to mark an overfull line. You can invoke this option by changing the `\documentclass` line to

```
\documentclass[draft]{sample}
```

A version of `note1b.tex` with this option can be found in the `samples` folder under the name `noteslug.tex`. Typeset it to see the "slugs".

2.4 More text features

Next, we produce the following note:

September 12, 2006

From the desk of George Grätzer

October 7–21 *please* use my temporary e-mail address:

`George_Gratzer@yahoo.com`

Type in the source file, without the line numbers. Save it as `note2.tex` in your `work` folder (`note2.tex` can be found in the `samples` folder—see page 4):

```
 1  % Sample file: note2.tex
 2  \documentclass{sample}
 3
 4  \begin{document}
 5  \begin{flushright}
 6     \today
 7  \end{flushright}
 8  \textbf{From the desk of George Gr\"{a}tzer}\\[22pt]
 9  October~7--21 \emph{please} use my
10  temporary e-mail address:
11  \begin{center}
12     \texttt{George\_Gratzer@yahoo.com}
13  \end{center}
14  \end{document}
```

This note introduces several additional text features of LaTeX:

- The `\today` command (in line 6) to display the date on which the document is typeset (so you will see a date different from the date shown above in your own typeset document).

- The environments to *right justify* (lines 5–7) and *center* (lines 11-13) text.

- The commands to change the text style, including the `\emph` command (line 8) to *emphasize* text, the `\textbf` command (line 9) for **bold** text, and the `\texttt` command (line 12) to produce `typewriter style` text.

  These are *commands with arguments.* In each case, the argument of the command follows the name of the command and is typed between braces, that is, between `{` and `}`.

- The form of the LaTeX commands: Almost all LaTeX *commands* start with a backslash (`\`) followed by the *command name*. For instance, `\textbf` is a command and `textbf` is the command name. The command name is terminated by the first *non-alphabetic character*, that is, by any character other than a–z or A–Z. So `textbf1` is not a command name, in fact, `\textbf1` typesets as **1**. (Let us look at this a bit more closely. `\textbf` is a valid command. If a command needs an argument and is not followed by braces, then it takes the next character as its argument. So `\textbf1` is the command `\textbf` with the argument `1`, which typesets as bold `1`: **1**.) Note that command names are *case sensitive.* Typing `\Textbf` or `\TEXTBF` generates an error message.

- The multiple role of hyphens: Double hyphens are used for number ranges. For example, `7--21` (in line 9) typesets as 7–21. The punctuation mark – is called an *en*

dash. Use triple hyphens for the *em dash* punctuation mark—such as the one in this sentence.

- The *new line* command, \\ (or `\newline`): To create additional space between lines (as in the last note, under the line **From the desk...**), you can use the \\ command and specify an appropriate amount of vertical space: `\\[22pt]`. Note that this command uses *square brackets* rather than braces because the argument is *optional*. The distance may be given in points (`pt`), centimeters (`cm`), or inches (`in`). (There is an analogous *new page* command, `\newpage`, not used in this short note.)

- Special rules for special characters (see Section 2.1), for *accented characters* and for some *European characters*. For instance, the accented character ä is typed as `\"{a}`. Accents are explained in Section 5.4.7 (see also the tables in Section C.2).

When you need to know more about typing text than we have discussed here, see Chapters 5 and 6. See also Appendix C, where all text symbols are organized into tables. Recall that we also have the `SymbolTables.pdf` in the `samples` folder.

CHAPTER

3

Typing math

While marking up text in LaTeX is easy, marking up math is less intuitive because math formulas are two-dimensional constructs and we have to mark them up with a one-dimensional string of characters. However, even the most complicated two-dimensional formula is made up of fairly simple building blocks. So by concentrating on the building blocks—selectively, just learn the ones you need—you can get started with math quickly.

3.1 A note with math

In addition to the regular text keys and the 13 special keys discussed in Section 2.1, two more keys are used to type math:

< >

The formula $2 < |x| > y$ (typed as `$2 < |x| >y$`) uses both. Note that such math formulas, called *inline*, are enclosed by `$` symbols. We discuss shortly another kind of math formula called *displayed*.

We begin typesetting math with the following note:

In first-year calculus, we define intervals such as (u, v) and (u, ∞). Such an interval is a *neighborhood* of a if a is in the interval. Students should realize that ∞ is only a symbol, not a number. This is important since we soon introduce concepts such as $\lim_{x\to\infty} f(x)$.

When we introduce the derivative

$$\lim_{x\to a} \frac{f(x) - f(a)}{x - a},$$

we assume that the function is defined and continuous in a neighborhood of a.

To create the source file for this mixed text and math note, create a new document with your text editor. Name it `math.tex`, place it in the `work` folder, and type in the following source file—without the line numbers—or simply copy `math.tex` from the `samples` folder (see page 4):

```
 1  % Sample file: math.tex
 2  \documentclass{sample}
 3
 4  \begin{document}
 5  In first-year calculus, we  define intervals  such
 6  as  $(u, v)$ and $(u, \infty)$. Such an interval
 7  is a \emph{neighborhood} of  $a$
 8  if  $a$ is in the interval. Students should
 9  realize that  $\infty$ is only a
10  symbol, not a number. This is important since
11  we soon introduce concepts
12   such as $\lim_{x \to \infty} f(x)$.
13
14  When we introduce the derivative
15  \[
16     \lim_{x \to a} \frac{f(x) - f(a)}{x - a},
17  \]
18  we assume that the function is defined and
19  continuous in a neighborhood of  $a$.
20  \end{document}
```

This note introduces several basic concepts of math in LaTeX:

- There are two kinds of math formulas and environments in `math.tex`:
  - *Inline* math environments open and close with `$` (as seen throughout this book) or open with `\(` and close with `\)`.
  - *Displayed* math environments open with `\[` and close with `\]`.

- Within math environments, LaTeX uses its own spacing rules and completely ignores the white space you type, with two exceptions:
  - Spaces that terminate commands. So in `$\infty a$` the space is not ignored, `$\inftya$` produces an error.
  - Spaces in the arguments of commands that temporarily revert to regular text. `\text` is such a command (see Sections 3.3 and 7.4.6).

  The white space that you add when typing math is important only for the readability of the source file. We summarize with a simple rule.

Rule ■ Spacing in text and math
Many spaces equal one space in text, whereas your spacing is ignored in math, unless the space terminates a command.

- A math symbol is invoked by a command. For example, the command for ∞ is `\infty` and the command for $\to$ is `\to`. The math symbols are organized into tables in Appendix B (see also `SymbolTables.pdf` in the `samples` folder).
- Some commands, such as `\sqrt`, need *arguments* enclosed by `{` and `}`. To typeset $\sqrt{5}$, type `$\sqrt{5}$`, where `\sqrt` is the command and `5` is the argument. Some commands need more than one argument. To get

$$\frac{3+x}{5}$$

type

```
\[
   \frac{3+x}{5}
\]
```

where `\frac` is the command, `3+x` and `5` are the arguments—we indent for readability.

3.2 Errors in math

Even in such a simple note there are opportunities for errors. To help familiarize yourself with some of the most commonly seen LaTeX math errors and their causes, we deliberately introduce mistakes into `math.tex`. The version of `math.tex` with mistakes is `mathb.tex`. By inserting and deleting `%` signs, you make the mistakes visible to LaTeX one at a time—recall that lines starting with `%` are comments and are therefore ignored by LaTeX.

Type the following source file, and save it under the name `mathb.tex` in the `work` folder or copy the file `mathb.tex` from the `samples` folder (see page 4). Do not type the line numbers—they are shown here to help you with the experiments.

```
 1  % Sample file: mathb.tex
 2  \documentclass{sample}
 3
 4  \begin{document}
 5  In first-year calculus, we   define intervals  such
 6  % as $(u, v)$ and $(u, \infty)$. Such an interval
 7  as $(u, v)$ and  (u, \infty)$. Such an interval
 8  is a \emph{neighborhood} of $a$
 9  if $a$ is in the interval. Students should
10  realize that  $\infty$ is only a
11  symbol, not a number. This is important since
12  we soon introduce concepts
13   such as  $\lim_{x \to \infty} f(x)$.
14  %such as $\lim_{x \to \infty  f(x)$.
15
16  When we introduce the derivative
17  \[
18      \lim_{x \to a} \frac{f(x) - f(a)}{x - a}
19  %   \lim_{x \to a} \frac{f(x) - f(a)  x - a}
20  \]
21  we assume that the function is defined and
22  continuous in a neighborhood of  $a$.
23  \end{document}
```

Experiment 1 In line 7, the `$` before the `(u` is missing. Typeset the `mathb.tex` source file. LaTeX generates the following error message:

```
! Missing $ inserted.
<inserted text>
                $
l.7 as $(u, v)$ and  (u, \infty
                                )$. Such an interval
```

Since the `$` was omitted, LaTeX reads `(u, \infty)` as text; but the `\infty` command instructs LaTeX to typeset a math symbol, which can only be done in a math formula. So LaTeX offers to put a `$` in front of `\infty` while typesetting the source file—it does not put the `$` in the source file itself. LaTeX attempts a cure, but in this example it comes too late, because the math formula *should* start just before `(u`.

Whenever you see the `?` prompt, you may press Return to ignore the error and continue typesetting the document. Section D.4 lists a number of other options and prompts.

Experiment 2 Uncomment line 6 by deleting the `%` at the beginning of line 6 and comment out line 7 by inserting a `%` at the beginning of line 7. This eliminates the previous error. Uncomment line 14 and comment out line 13. This introduces a new error, the closing brace of the subscript is missing. Now typeset the note. You get the error message

```
! Missing } inserted.
<inserted text>
                }
l.14 such as $\lim_{x \to \infty f(x)$
```

LaTeX reports that a closing brace (`}`) is missing, but it is not sure where the brace should be. LaTeX noticed that a subscript (see page 23) started with `{`, but LaTeX reached the end of the math formula before finding a closing brace `}`. To remedy this, you must look in the formula for an opening brace `{` that is not balanced, and insert the missing closing brace `}`. Make the necessary change and typeset again to view the difference.

Experiment 3 Uncomment line 13 and comment out line 14, removing the previous error. Delete the `%` at the beginning of line 19 and insert a `%` at the beginning of line 18, introducing our final error, omitting the closing brace of the first argument and the opening brace of the second argument of `\frac`. Save and typeset the file. You get the error message

```
! Too many }'s.
\frac  #1#2->{\begingroup #1\endgroup \@@over #2}

l.20 \]
```

LaTeX got confused. The second line of the message explains that `\frac` has two arguments and it is not working out, but the error is incorrectly identified.

If the typo on line 19 is `\frac{f(x)-f(a) x-a}g`, then LaTeX produces

$$\frac{f(x) - f(a)x - a}{g}$$

and no error message is generated.

Experiment 4 Make sure all the errors are commented out. Typeset `mathb.tex`, testing that there are no errors. Now delete the two $ signs in line 22, that is, replace `$a$` by `a`. Typeset the file. It typesets with no errors. Here is the last line of the typeset file you get:

we assume that the function is defined and continuous in a neighborhood of a.

instead of

we assume that the function is defined and continuous in a neighborhood of a.

This is probably the error most often made by beginners. There is no error message by LaTeX and the typeset version looks good. You need sharp eyes to catch such an error.

See Section 4.3.1 for more information about finding and fixing problems in your LaTeX source files.

3.3 Building blocks of a formula

A formula is built from a large collection of components. We group them as follows:

- Arithmetic
  - Subscripts and superscripts
- Binomial coefficients
- Congruences
- Delimiters
- Ellipses
- Integrals
- Math accents
- Matrices
- Operators
  - Large operators
- Roots
- Text

In this section, I describe each of these groups, and provide examples illustrating their use.

Arithmetic The arithmetic operations $a + b$, $a - b$, $-a$, a/b, and ab are typed in the natural way (the spaces are typed only for readability, others may type fewer spaces):

```
$a + b$, $a - b$, $-a$, $a / b$, $a b$
```

If you wish to use $\cdot$ or $\times$ for multiplication, as in $a \cdot b$ or $a \times b$, use `\cdot` or `\times`, respectively. The expressions $a \cdot b$ and $a \times b$ are typed as follows:

```
$a \cdot b$  $a \times b$
```

Displayed fractions, such as

$$\frac{1 + 2x}{x + y + xy}$$

are typed with `\frac`:

```
\[
   \frac{1 + 2x}{x + y + xy}
\]
```

The `\frac` command is seldom used inline because it can disrupt the interline spacing of the paragraph.

Subscripts and superscripts Subscripts are typed with _ (underscore) and superscripts with ^ (caret). Subscripts and superscripts should be enclosed in braces, that is, typed between { and }. To get a_1, type `$a_{1}$`. Omitting the braces in this example causes no harm, but to get a_{10}, you *must* type `$a_{10}$`. Indeed, `$a_10$` is typeset as a_10. Further examples, a_{i_1}, a^2, a^{i_1}, a_n^2, are typed as

```
$a_{i_{1}}$, $a^{2}$, $a^{i_{1}}$, $a_{n}^{2}$
```

There is one symbol, the prime (’), that is automatically superscripted in math. To get $f'(x)$, just type `$f'(x)$`.

Binomial coefficients Binomial coefficients are typeset with the `\binom` command. For example, $\binom{a}{b+c}$ is typed inline as

```
$\binom{a}{b + c}$
```

whereas a displayed version,

$$\binom{\frac{n^2-1}{2}}{n+1}$$

is typed as

```
\[
   \binom{ \frac{n^{2} - 1}{2} }{n + 1}
\]
```

Congruences The two most important forms are

$a \equiv v \pmod{\theta}$	typed as	`$a \equiv v \pmod{\theta}$`
$a \equiv v \pod{\theta}$	typed as	`$a \equiv v \pod{\theta}$`

Delimiters Parentheses and square brackets are examples of delimiters. They are used to delimit some subformulas, as in `$[(a*b)+(c*d)]^{2}$`, which typesets as $[(a*b)+(c*d)]^2$. LaTeX can be instructed to expand them vertically to enclose a formula such as

$$\left(\frac{1 + x}{2 + y^2} \right)^2$$

which is typed as

```
\[
   \left( \frac{1 + x}{2 + y^{2}} \right)^{2}
\]
```

The `\left(` and `\right)` commands tell LaTeX to size the parentheses correctly, relative to the size of the symbols inside the parentheses. Two further examples,

$$\left| \frac{a + b}{2} \right|, \quad \left\| A^2 \right\|$$

would be typed as

```
\[
  \left| \frac{a + b}{2} \right|,
      \quad \left\| A^{2} \right\|
\]
```

where `\quad` is a spacing command (see Sections 8.1 and B.9).

Additional delimiters are listed in Sections 7.5 and B.6.

Ellipses The *ellipsis* (. . .) in text is provided by the `\dots` command:

A. . . Z is typed as `A \dots Z`

In formulas, the ellipsis is printed either as *low* (or *on-the-line*) *dots*:

$F(x_1, \ldots, x_n)$ is typed as `$F(x_{1}, \dots, x_{n})$`

or as centered dots:

$x_1 + \cdots + x_n$ is typed as `$x_{1} + \dots + x_{n}$`

The command `\dots` typesets the correct ellipsis with the correct spacing in most cases. If it does not, see Section 7.4.3 on how to specify the appropriate ellipsis from the four types available.

Integrals The command for an integral is `\int`. The lower limit is specified as a subscript and the upper limit is specified as a superscript. For example, the formula $\int_0^{\pi} \sin x \, dx = 2$ is typed as

```
$\int_{0}^{\pi} \sin x \, dx = 2$
```

where `\,` is a spacing command (see Sections 8.1 and B.9).

Math accents The four most frequently used math accents are:

$\bar{a}$ typed as `$\bar{a}$`

$\hat{a}$ typed as `$\hat{a}$`

$\tilde{a}$ typed as `$\tilde{a}$`

$\vec{a}$ typed as `$\vec{a}$`

See Sections 7.7 and B.8 for complete lists.

Matrices You type the matrix

$$\begin{matrix} a+b+c & uv & x-y & 27 \\ a+b & u+v & z & 134 \end{matrix}$$

with the `\matrix` command

```
\[
   \begin{matrix}
     a + b + c & uv    & x - y & 27\\
     a + b     & u + v & z     & 134
   \end{matrix}
\]
```

The `matrix` environment separates adjacent matrix elements within a row with ampersands (`&`). Rows are *separated* by new line commands (`\\`). Do not end the last row with a new line command!

The `matrix` environment has to appear within a math environment, as in the example. As a rule, it is in a displayed math environment, since inline it appears too large. It can be used in the `align` environment discussed in Section 3.4.2.

The `matrix` environment does not provide delimiters. Several variants do, including `pmatrix` and `vmatrix`. For example,

$$\mathbf{A} = \begin{pmatrix} a+b+c & uv \\ a+b & u+v \end{pmatrix} \begin{vmatrix} 30 & 7 \\ 3 & 17 \end{vmatrix}$$

is typed as follows:

```
\[
   \mathbf{A} =
   \begin{pmatrix}
     a + b + c & uv\\
     a + b & u + v
   \end{pmatrix}
   \begin{vmatrix}
     30 & 7\\
     3 & 17
   \end{vmatrix}
\]
```

As you can see, `pmatrix` typesets as a `matrix` between a pair of `\left(` and `\right)` commands, while `vmatrix` typesets as a `matrix` between a pair of `\left|` and `\right|` commands. See Section 9.7.1 for a listing of all the matrix variants.

Operators To typeset the sine function, $\sin x$, type `$\sin x$`.

Note that `$sin x$` would be typeset as $sinx$ because LaTeX interprets this expression as the product of four variables.

LaTeX calls `\sin` an *operator.* Sections 7.6.1 and B.7 list a number of operators. See Section 7.6.2 for user-defined operators. Some are just like `\sin`. Others produce a more complex display, for example,

$$\lim_{x \to 0} f(x) = 0$$

is typed as

```
\[
   \lim_{x \to 0} f(x) = 0
\]
```

Large operators The command for *sum* is `\sum` and for *product* is `\prod`. The following examples,

$$\sum_{i=1}^{n} x_i^2 \qquad \prod_{i=1}^{n} x_i^2$$

are typed as

```
\[
   \sum_{i=1}^{n} x_{i}^{2} \qquad
    \prod_{i=1}^{n} x_{i}^{2}
\]
```

where \qquad is a spacing command (see Sections 8.1 and B.9) used to separate the two formulas, yielding twice the space produced by \quad.

Sums and products are examples of *large operators.* They are typeset bigger when displayed than inline. They are listed in Sections 7.6.4 and B.7.1.

Roots \sqrt produces a square root. For instance, $\sqrt{a+2b}$ is typed as

```
$\sqrt{a + 2b}$
```

The n-th root, $\sqrt[n]{5}$, requires the use of an *optional argument,* which is specified using brackets (see Section 5.3.1): `$\sqrt[n]{5}$`.

Text You can include text in a formula with a \text command. For instance,

$$a = b, \qquad \text{by assumption}$$

is typed as

```
\[
   a = b, \text{\qquad by assumption}
\]
```

Note the spacing command \qquad in the argument of \text. You could also type

```
\[
   a = b, \qquad \text{by assumption}
\]
```

because \qquad works in math as well as in text (see Sections 8.1 and B.9).

3.4 Displayed formulas

3.4.1 Equations

The equation environment creates a displayed math formula and automatically generates an equation number. The equation

$$\int_0^{\pi} \sin x \, dx = 2 \tag{1}$$

is typed as

```
\begin{equation}\label{E:firstInt}
  \int_{0}^{\pi} \sin x \, dx = 2
\end{equation}
```

The equation number, which is automatically generated, depends on how many other numbered equations occur before the given equation.

To reference this formula without having to remember a number—which may change when you edit your document—give the equation a symbolic label by using the `\label` command and refer to the equation in your document by using the symbolic label, the argument of the `\label` command. In this example, I have called the first equation `firstInt` (first integral), and used the convention that the label of an equation starts with `E:`, so that the complete `\label` command is

```
\label{E:firstInt}
```

The number of this formula is referenced with the `\ref` command. Its page is referenced using the `\pageref` command. For example, to get

see (1) on page 27

type

```
see~(\ref{E:firstInt}) on page~\pageref{E:firstInt}
```

The `\eqref` command provides the reference number in parentheses. So the last example could be typed

```
see~\eqref{E:firstInt} on page~\pageref{E:firstInt}
```

The `\eqref` command is smart. Even if the equation number is referenced in emphasized or italicized text, the reference typesets upright (in roman type).

Note the use of the nonbreakable space (`~`) to ensure that when typeset the equation number is on the same line as the word `see`. (See the footnote on page 11.) You should always use a nonbreakable space to link a `\ref` command to the name of its part, for instance, equation, page, section, chapter. Use two nonbreakable spaces in

```
Sections~\ref{S:main} and~\ref{S:subsidiary}.
```

The main advantage of this cross-referencing system is that when you add, delete, or rearrange equations, LaTeX automatically renumbers the equations and adjusts the references that appear in your typeset document. You can split a long article into two or move a section to the end, and LaTeX takes care of the renumbering. This significantly reduces the amount of time you need to spend working on your document. It also reduces the potential for errors in the finished project.

Rule ■ Typeset twice

For renumbering to work, you have to typeset the source file twice.

The first run creates a list of references that need to be linked. The second creates the cross references and inserts the relevant text throughout the document (see Sections 18.2 and D.3.4). LaTeX issues a warning if you forget. Such warnings do not interrupt the typesetting, you only see them in the `log` window—if the window is visible—and in the `log` file. It is a good idea to check for warnings periodically.

An equation is numbered whether or not there is a `\label` command attached to it. Of course, if there is no `\label` command, the number generated for the equation by LaTeX cannot be referenced with the command `\ref` or `\eqref`.

The system described here is called *symbolic referencing*. The symbol for the number is the argument of the `\label` command, and that symbol can be referenced with `\ref`, `\eqref`, or `\pageref` commands. LaTeX uses the same mechanism for all of the generated numbering systems: sections, subsections, subsubsections, equations, theorems, lemmas, and bibliographic references—except that for bibliographic references, LaTeX uses the `\bibitem` command to define a bibliographic item and the `\cite` command to cite a bibliographic item (see Section 4.2.4 and Chapter 16).

What happens if you misspell a reference, e.g., typing `\ref{E:firstint}` instead of `\ref{E:firstInt}`? LaTeX typesets **??**. There are two warnings in the `log` file:

```
LaTeX Warning: Reference ‘E:firstint’ on page 39
        undefined on input line 475.
```

for the typeset page and the other one close to the end:

```
LaTeX Warning: There were undefined references.
```

If a `\cite` is misspelled, you get [**?**] and similar warnings.

Equations can also be *tagged* by attaching a name to the formula with the `\tag` command. The tag replaces the equation number.

For example,

(Int) $$\int_0^{\pi} \sin x \, dx = 2$$

is typed as

```
\begin{equation}
  \int_{0}^{\pi} \sin x \, dx = 2 \tag{Int}
\end{equation}
```

Tags (of the type discussed here) are *absolute.* This equation is *always* referred to as (Int). Equation numbers, on the other hand, are *relative,* they may change when equations are added, deleted, or rearranged.

3.4.2 Aligned formulas

LaTeX has many ways to typeset multiline formulas. We discuss three constructs in this section: *simple alignment, annotated alignment,* and *cases*. See Chapter 9 for many others.

For simple and annotated alignment we use the `align` environment. Each line in the `align` environment is a separate equation, which LaTeX automatically numbers.

Simple alignment

Simple alignment is used to align two or more formulas. To obtain the formulas

$$r^2 = s^2 + t^2, \tag{2}$$

$$2u + 1 = v + w^{\alpha}, \tag{3}$$

$$x = \frac{y + z}{\sqrt{s + 2u}}; \tag{4}$$

type the following, using `\\` as the *line separator* and `&` as the *alignment point:*

```
\begin{align}
  r^{2}  &= s^{2} + t^{2},              \label{E:Pyth}\\
  2u + 1 &= v + w^{\alpha},             \label{E:alpha}\\
       x &= \frac{y + z}{\sqrt{s + 2u}};\label{E:frac}
\end{align}
```

Note that you should not have a `\\` to terminate the last line.

Figure 3.1 displays the source and the typeset versions of formulas (2)–(4), emphasizing the alignment points of the source and the typeset formula. Of course, in the source, the alignment points do not have to line up.

These formulas are numbered (2), (3), and (4) because they are preceded by one numbered equation earlier in this section.

The `align` environment can also be used to break a long formula into two or more parts. Since numbering both lines in such a case would be undesirable, you can prevent the numbering of the second line by using the `\notag` command in the second part of the formula.

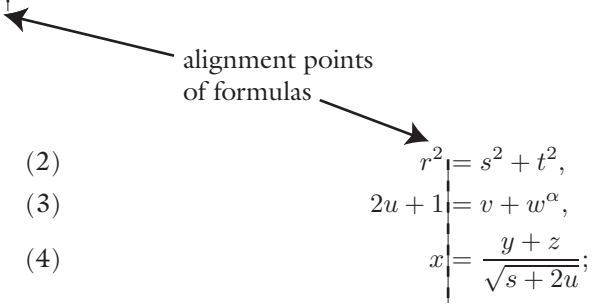
For example,

$$\begin{aligned} h(x) &= \int \left(\frac{f(x) + g(x)}{1 + f^2(x)} + \frac{1 + f(x)g(x)}{\sqrt{1 - \sin x}} \right) dx \\ &= \int \frac{1 + f(x)}{1 + g(x)}\, dx - 2\tan^{-1}(x - 2) \end{aligned} \tag{5}$$

is typed as follows:

```
\begin{align}
  h(x) &= \int \left( \frac{f(x) + g(x)}{1+ f^{2}(x)}
```

```
r^{2}  &= s^{2} + t^{2},                \label{E:Pyth}\\
2u + 1 &= v + w^{\alpha},               \label{E:alpha}\\
     x &= \frac{y + z}{\sqrt{s + 2u}};  \label{E:frac}
```



$$r^2 = s^2 + t^2, \tag{2}$$
$$2u + 1 = v + w^{\alpha}, \tag{3}$$
$$x = \frac{y + z}{\sqrt{s + 2u}}; \tag{4}$$

Figure 3.1: Simple alignment: source and typeset.

```
          + \frac{1+ f(x)g(x)}{\sqrt{1 - \sin x}}
          \right) \, dx\label{E:longInt}\\
       &= \int \frac{1 + f(x)}{1 + g(x) } \, dx
         - 2 \tan^{-1}(x-2)\notag
\end{align}
```

The rules for simple alignment are easy to remember.

Rule ■ Simple alignments

- Use the `align` environment.
- Separate the lines with `\\`.
- In each line, indicate the alignment point with `&`, one `&` per line. If the alignment point is adjacent to an $=$, $+$, and so on, place it *before* to ensure proper spacing.
- Place a `\notag` command in each line that you do not wish numbered.
- If no line should be numbered, use the `align*` environment.
- Place a `\label` command in each numbered line you may want to reference with `\ref`, `\eqref`, or `\pageref`.

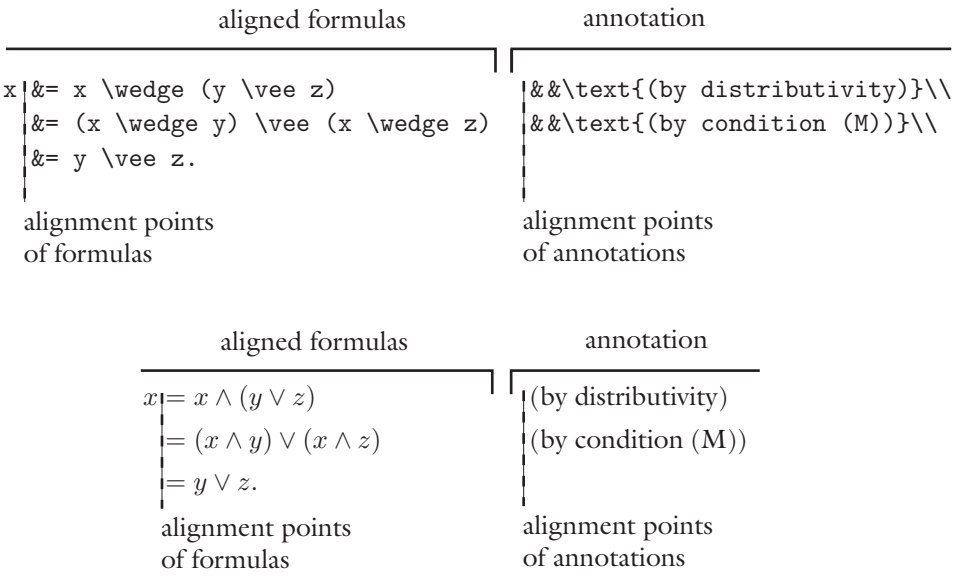


Figure 3.2: Annotated alignment: source and typeset.

Annotated alignment

Annotated alignment allows you to align formulas and their annotations, that is, explanatory text, separately (see Figure 3.2):

(6)
$$\begin{aligned} x &= x \wedge (y \vee z) && \text{(by distributivity)}\\ &= (x \wedge y) \vee (x \wedge z) && \text{(by condition (M))}\\ &= y \vee z. \end{aligned}$$

This example is typed as

```
\begin{align}
   x &= x \wedge (y \vee z)
    &&\text{(by distributivity)}\label{E:DoAlign}\\
   &= (x \wedge y) \vee (x \wedge z)
    &&\text{(by condition (M))} \notag\\
   &= y \vee z. \notag
\end{align}
```

The rules for annotated alignment are similar to the rules of simple alignment. In each line, in addition to the alignment point marked by `&`, there is also a mark for the start of the annotation: `&&`.

3.4.3 Cases

The `cases` construct is a specialized matrix. It has to appear within a math environment such as the `equation` environment (see Section 3.4.1) or the `align` environment (see Section 3.4.2). Here is a typical example:

$$f(x) = \begin{cases} -x^2, & \text{if } x < 0; \\ \alpha + x, & \text{if } 0 \leq x \leq 1; \\ x^2, & \text{otherwise.} \end{cases}$$

is typed as follows:

```
\[
   f(x)=
   \begin{cases}
     -x^{2},          &\text{if $x < 0$;}\\
     \alpha + x,      &\text{if $0 \leq x \leq 1$;}\\
     x^{2},           &\text{otherwise.}
   \end{cases}
\]
```

Notice how you can put inline math, opened and closed with `$`, inside the argument of a `\text` command.

The rules for using the `cases` environment are the same as for matrices. Separate the lines with `\\` and indicate the annotation with `&`.

CHAPTER

4

Your first article and presentation

4.1 The anatomy of an article

To begin, we use the sample article `intrart.tex` (in the `samples` folder) to examine the anatomy of an article. Copy it over to the `work` folder or type it, and save it in the `work` folder as we discuss the parts of an article.

Every LaTeX article has two parts, the preamble and the body. The *preamble* of an article is everything from the first line of the source file down to the line

```
\begin{document}
```

For a schematic view of an article, see Figure 4.1.

The preamble contains instructions affecting the entire document. The *only* required command in the preamble is the `\documentclass` command. There are other commands (such as the `\usepackage` commands) that must be placed in the preamble if they are used, but these commands do not have to be present in every document.

Here is the preamble of the introductory sample article:

```
% Introductory sample article: intrart.tex

\documentclass{amsart}
\usepackage{amssymb,latexsym}
\usepackage{graphicx}
\newtheorem{theorem}{Theorem}
\newtheorem{lemma}{Lemma}
\newtheorem{definition}{Definition}
\newtheorem{notation}{Notation}
```

The preamble specifies the *document class* and then the LaTeX enhancements, or *packages,* used in the article. The preamble can also specify additional commands that are used throughout the document, such as proclamation definitions, user-defined commands, and so on.

`intrart.tex` specifies the `amsart` document class. This class defines the format used by the AMS journals—and many others—for articles. LaTeX then loads two pack-

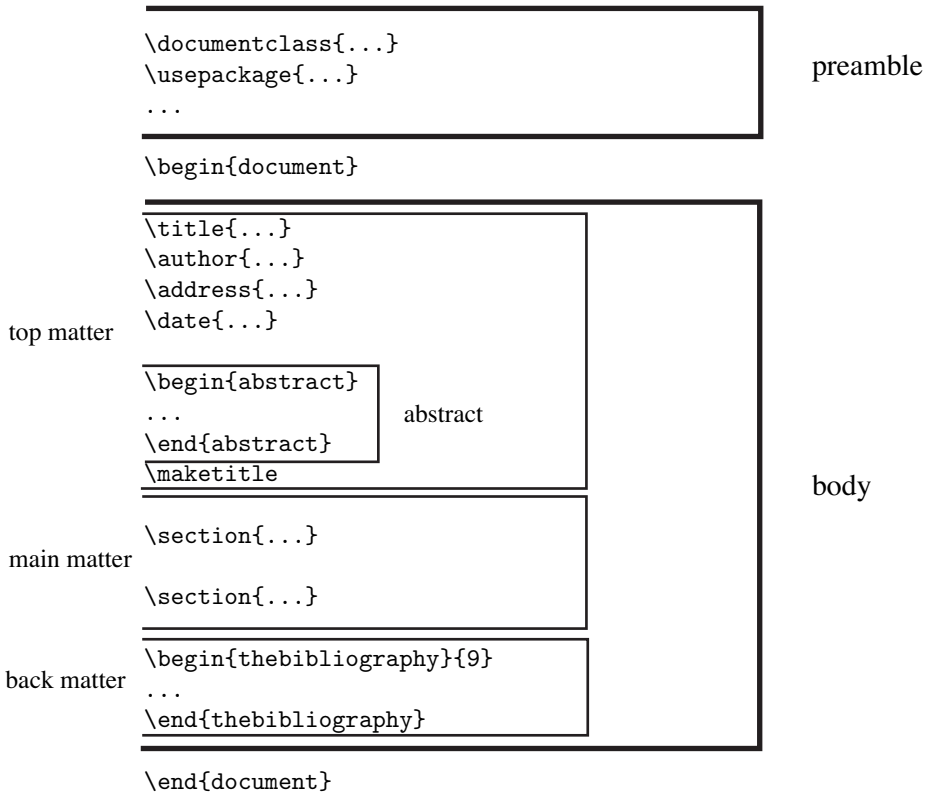


Figure 4.1: A schematic view of an article.

ages, `latexsym` and `amssymb`, that provide the names of some mathematical symbols. Finally, LaTeX loads the `graphicx` package, which we need because of the illustration.

The preamble concludes with the *proclamations*. A proclamation is a theorem, lemma, definition, corollary, note, or other similar construct. The `intrart.tex` article defines four proclamations. The first of these,

```
\newtheorem{theorem}{Theorem}
```

defines the `theorem` environment, which then can be used in the body of the article, as explained in Section 4.2.3. The other three are similar. LaTeX automatically numbers and formats proclamations.

The article proper, called the *body,* is the content of the `document` environment—it is between the lines

```
\begin{document}
```

and

```
\end{document}
```

as illustrated in Figure 4.1. The body of an article is split into several parts, starting with the *top matter* containing title page information and the abstract. The top matter follows the line

```
\begin{document}
```

and concludes with the line

```
\maketitle
```

Here is the top matter of the introductory sample article:

```
\title{A construction of complete-simple\\
       distributive lattices}
\author{George~A. Menuhin}
\address{Computer Science Department\\
         University of Winnebago\\
         Winnebago, MN 53714}
\date{March 15, 2006}

\begin{abstract}
In this note, we prove that there exist
\emph{complete-simple distributive lattices,}
that is, complete distributive lattices
with only two complete congruences.
\end{abstract}

\maketitle
```

And here is the rest of the body of the introductory sample article with some commentary, exclusive of the bibliography:

```
\section{Introduction}\label{S:intro}
In this note, we prove the following result:

\begin{theorem}
There exists an infinite complete distributive
lattice~$K$ with only the two trivial complete
congruence relations.
\end{theorem}

\section{The $\Pi^{*}$ construction}\label{S:P*}
The following construction is crucial in the proof
of our Theorem (see Figure~\ref{Fi:products}):

\begin{definition}\label{D:P*}
Let $D_{i}$, for $i \in I$, be complete distributive
lattices satisfying condition~\textup{(J)}. Their
$\Pi^{*}$ product is defined as follows:
\[
   \Pi^{*} ( D_{i} \mid i \in I ) =
   \Pi ( D_{i}^{-} \mid i \in I ) + 1;
\]
that is, $\Pi^{*} ( D_{i} \mid i \in I )$ is
$\Pi ( D_{i}^{-} \mid i \in I )$ with a new
unit element.
\end{definition}

\begin{notation}
If $i \in I$ and $d \in D_{i}^{-}$, then
\[
  \langle \dots, 0, \dots, d, \dots, 0, \dots \rangle
\]
is the element of $\Pi^{*} ( D_{i} \mid i \in I )$ whose
$i$-th component is $d$ and all the other components
are $0$.
\end{notation}

See also Ernest~T. Moynahan~\cite{eM57a}.

Next we verify the following result:
```

```
\begin{theorem}\label{T:P*}
Let $D_{i}$, $i \in I$, be complete distributive
lattices satisfying condition~\textup{(J)}.
Let $\Theta$ be a complete congruence relation on
$\Pi^{*} ( D_{i} \mid i \in I )$.
If there exist $i \in I$ and $d \in D_{i}$ with
$d < 1_{i}$ such that, for all $d \leq c < 1_{i}$,
\begin{equation}\label{E:cong1}
   \langle \dots, d, \dots, 0, \dots \rangle \equiv
   \langle \dots, c, \dots, 0, \dots \rangle
   \pod{\Theta},
\end{equation}
then $\Theta = \iota$.
\end{theorem}
```

We include an illustration, `products.eps` or `products.pdf` (in your `samples` folder). We copy them over to the `work` folder and load the `graphicx` package. We name the illustration in the `figure` environment. The illustration must be in the graphic image file formats EPS or PDF (see Chapter 13.1.2). We left the argument of the `\caption` command empty—it normally contains the name or a description of the figure. The illustration is centered with the `\centering` command (see Section 6.3).

```
\begin{figure}[hbt]
\centering\includegraphics{products}
\caption{}\label{Fi:products}
\end{figure}
```

The `figure` environment *floats,* that is, LaTeX decides where to place the typeset figure. We can influence LaTeX's choice (see Section 10.4.3).

Then we place a proof in a `proof` environment.

```
\begin{proof}
Since
\begin{equation}\label{E:cong2}
\langle \dots, d, \dots, 0, \dots \rangle \equiv
\langle \dots, c, \dots, 0, \dots \rangle
\pod{\Theta},
\end{equation}
and $\Theta$ is a complete congruence relation,
it follows from condition~(J) that
\begin{equation}\label{E:cong}
 \langle \dots, d, \dots, 0, \dots \rangle \equiv
 \bigvee ( \langle \dots, c, \dots, 0, \dots \rangle
 \mid d \leq c < 1 ) \pod{\Theta}.
```

```
\end{equation}

Let $j \in I$, $j \neq i$, and let $a \in D_{j}^{-}$.
Meeting both sides of the congruence \eqref{E:cong2}
with $\langle \dots, a, \dots, 0, \dots \rangle$,
we obtain that
\begin{equation}\label{E:comp}
   0 = \langle \dots, a, \dots, 0, \dots \rangle
     \pod{\Theta},
\end{equation}
Using the completeness of $\Theta$ and \eqref{E:comp},
we get:
\[
   0 \equiv \bigvee ( \langle \dots, a, \dots, 0,
     \dots \rangle \mid a \in D_{j}^{-} ) = 1
     \pod{\Theta},
\]
hence $\Theta = \iota$.
\end{proof}
```

At the end of the body, the *bibliographic entries* are typed between the lines

```
\begin{thebibliography}{9}
```

and

```
\end{thebibliography}
```

There are fewer than 10 references in this article, so we tell LaTeX to make room for single-digit numbering by providing the argument `9` to the `thebibliography` environment. We use `99` if the number of references is between 10 and 99. The typeset bibliography is titled References.

The bibliography of `intrart.tex` is structured as follows:

```
\begin{thebibliography}{9}

\bibitem{sF90}
Soo-Key Foo,
\emph{Lattice Constructions},
Ph.D. thesis,
University of Winnebago, Winnebago, MN, December, 1990.

\bibitem{gM68}
George~A. Menuhin,
```

```
\emph{Universal algebra}.
D.~Van Nostrand, Princeton, 1968.

\bibitem{eM57}
Ernest~T. Moynahan,
\emph{On a problem of M. Stone},
Acta Math. Acad. Sci. Hungar. \textbf{8} (1957),
455--460.

\bibitem{eM57a}
Ernest~T. Moynahan,
\emph{Ideals and congruence relations in
lattices}. II,
Magyar Tud. Akad. Mat. Fiz. Oszt. K\"{o}zl.
\textbf{9} (1957), 417--434.

\end{thebibliography}
```

The body and the article end when the `document` environment is closed with

```
\end{document}
```

4.1.1 The typeset sample article

On the next two pages, you find the typeset `intrart.tex`, the introductory sample article.

A CONSTRUCTION OF COMPLETE-SIMPLE DISTRIBUTIVE LATTICES

GEORGE A. MENUHIN

Abstract. In this note, we prove that there exist *complete-simple distributive lattices,* that is, complete distributive lattices with only two complete congruences.

1. Introduction

In this note, we prove the following result:

Theorem 1. *There exists an infinite complete distributive lattice K with only the two trivial complete congruence relations.*

2. The Π^* construction

The following construction is crucial in the proof of our Theorem (see Figure 1):

Definition 1. *Let D_i, for $i \in I$, be complete distributive lattices satisfying condition* (J)*. Their Π^* product is defined as follows:*

$$\Pi^*(D_i \mid i \in I) = \Pi(D_i^- \mid i \in I) + 1;$$

that is, $\Pi^*(D_i \mid i \in I)$ is $\Pi(D_i^- \mid i \in I)$ with a new unit element.

Notation 1. *If $i \in I$ and $d \in D_i^-$, then*

$$\langle \dots, 0, \dots, d, \dots, 0, \dots \rangle$$

is the element of $\Pi^*(D_i \mid i \in I)$ whose i-th component is d and all the other components are 0*.*

See also Ernest T. Moynahan [4].

Next we verify the following result:

Theorem 2. *Let D_i, $i \in I$, be complete distributive lattices satisfying condition* (J)*. Let Θ be a complete congruence relation on $\Pi^*(D_i \mid i \in I)$. If there exist $i \in I$ and $d \in D_i$ with $d < 1_i$ such that, for all $d \leq c < 1_i$,*

$$\langle \dots, d, \dots, 0, \dots \rangle \equiv \langle \dots, c, \dots, 0, \dots \rangle \quad (\Theta), \tag{1}$$

then $\Theta = \iota$.

Proof. Since

$$\langle \dots, d, \dots, 0, \dots \rangle \equiv \langle \dots, c, \dots, 0, \dots \rangle \quad (\Theta), \tag{2}$$

and Θ is a complete congruence relation, it follows from condition (J) that

$$\langle \dots, d, \dots, 0, \dots \rangle \equiv \bigvee (\langle \dots, c, \dots, 0, \dots \rangle \mid d \leq c < 1) \quad (\Theta). \tag{3}$$

Date: March 15, 2006.

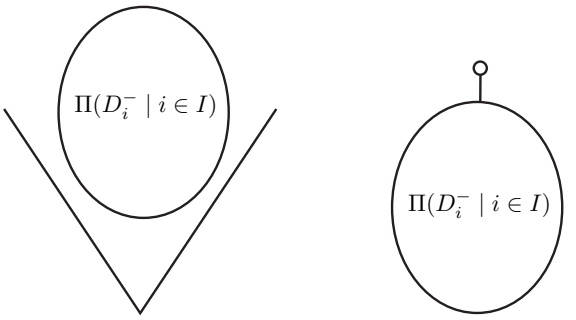


FIGURE 1

Let $j \in I$, $j \neq i$, and let $a \in D_j^-$. Meeting both sides of the congruence (2) with $\langle \dots, a, \dots, 0, \dots \rangle$, we obtain that

$$0 = \langle \dots, a, \dots, 0, \dots \rangle \quad (\Theta), \tag{4}$$

Using the completeness of Θ and (4), we get:

$$0 \equiv \bigvee (\langle \dots, a, \dots, 0, \dots \rangle \mid a \in D_j^-) = 1 \quad (\Theta),$$

hence $\Theta = \iota$. □

REFERENCES

[1] Soo-Key Foo, *Lattice Constructions*, Ph.D. thesis, University of Winnebago, Winnebago, MN, December, 1990.
[2] George A. Menuhin, *Universal algebra*. D. Van Nostrand, Princeton, 1968.
[3] Ernest T. Moynahan, *On a problem of M. Stone*, Acta Math. Acad. Sci. Hungar. **8** (1957), 455–460.
[4] Ernest T. Moynahan, *Ideals and congruence relations in lattices.* II, Magyar Tud. Akad. Mat. Fiz. Oszt. Közl. **9** (1957), 417–434.

COMPUTER SCIENCE DEPARTMENT, UNIVERSITY OF WINNEBAGO, WINNEBAGO, MN 53714

4.2 An article template

In this section, you create an article template. To start a new article, open the template and start writing!

Make a copy of `intrart.tex` and give the copy an appropriate name. I named mine `gg.tex` (it is in the `samples` folder). Remember the naming rule (page 4): The name should have no spaces, no special characters, and end with `.tex`. So the name `my template.tex` is not good, but `MyTemplate.tex` is fine.

4.2.1 Editing the top matter

Edit the top matter to contain the relevant information, e.g., title and address, for your template. Here are some simple rules to follow.

Rule ■ Top matter

1. `\thanks` places an unmarked footnote at the bottom of the first page, for instance to acknowledge research support. If it is not needed, comment it out.
2. Separate the lines of your address with `\\`. Do not put a `\\` at the end of the last line.
3. `\date{\today}` typesets today's date. If you do not want *any* date to appear, comment out the `\date` command. For a specific date, such as March 15, 2006, type `\date{March 15, 2006}`.
4. The `\title` command is the only required command. The others are optional.

Actually, if you do not give the `\maketitle` command, even the `\title` command is optional. On the other hand, if you do give the `\maketitle` command and omit the `\title` command, then you get the error message

```
! Undefined control sequence.
<argument> \shorttitle

l.27 \maketitle
```

Now delete all the content of the article, leaving you with the skeleton. Here is the edited `gg.tex` (it is in your `samples` folder):

```
% GG's article template: gg.tex

\documentclass{amsart}
\usepackage{amssymb,latexsym}
\usepackage{graphicx}
```

```
\newtheorem{theorem}{Theorem}
\newtheorem{lemma}{Lemma}
\newtheorem{definition}{Definition}
\newtheorem{notation}{Notation}

\begin{document}
\title{Title!}
\author{George Gr\"{a}tzer}
\address{Department of Mathematics\\
         University of Manitoba\\
         Winnipeg, MB R3T 2N2\\
         Canada}
\date{\today}

\begin{abstract}
To come!
\end{abstract}

\maketitle

\section{Introduction}\label{S:intro}

\begin{thebibliography}{9}

\end{thebibliography}

\end{document}
```

I also made a version for a joint article with another author: `gg2.tex` (see the `samples` folder). It adds the lines

```
\author{Second author}
\address{line1\\
         line2\\
         line3\\
         line4}
```

before the `\date` command.

When I start writing an article, I open `gg.tex` or `gg2.tex`, save it under a new name, and edit the top matter. Here are two more rules about the top matter to keep in mind:

Rule ■ Top matter (continued)

5. If necessary, break the title into separate lines with `\\`. Do not put a `\\` at the end of the last line.

6. Multiple authors get separate `\author` and `\address` commands.

4.2.2 Sectioning

An article, as a rule, is divided into sections. To start the section Introduction, type

```
\section{Introduction}\label{S:intro}
```

after the `\maketitle` command. `Introduction` typesets as the title of the section. I use the convention that `S:` starts the label for a section, so the label is `S:intro` (or something similar). The section's number is automatically assigned by LaTeX. You can refer to this section number with `\ref{S:intro}`:

```
In Section~\ref{S:intro}, we introduce
```

The command `\section*` produces an unnumbered section.

Sections have subsections, and subsections have subsubsections. The corresponding commands are

```
\subsection  \subsubsection
```

Their unnumbered variants are

```
\subsection*  \subsubsection*
```

4.2.3 Invoking proclamations

In the preamble of the article `intrart.tex`, we defined the theorem, lemma, definition, and notation proclamations. These proclamations define environments.

For example, you type a theorem within a `theorem` environment. The body of the theorem, that is, the part of the source file that produces the theorem, is typed between the lines

```
\begin{theorem}\label{T:xxx}
```

and

```
\end{theorem}
```

where `T:`*xxx* is the label for the theorem. You should replace *xxx* with a label that is somewhat descriptive of the contents of your theorem. LaTeX automatically assigns a number to the theorem, and the theorem can be referenced by using a command of the form `\ref{T:`*xxx*`}`.

4.2.4 Inserting references

The works to be listed are placed in the bibliography. Below are typical entries for the most frequently used types of references, an article in a journal, a book, a Ph.D. thesis, and a technical report. For more examples, see the bibliographic template file, `bibl.tpl`, in the `samples` folder.

```
\begin{thebibliography}{9}
   \bibitem{sF90}
      Soo-Key Foo,
      \emph{Lattice Constructions},
      Ph.D. thesis,
      University of Winnebago, Winnebago, MN,
      December, 1990.
   \bibitem{gM68}
      George~A. Menuhin,
      \emph{Universal algebra}.
      D.~Van Nostrand, Princeton, 1968.
   \bibitem{eM57}
      Ernest~T. Moynahan,
      \emph{On a problem of M. Stone},
      Acta Math. Acad. Sci. Hungar. \textbf{8} (1957),
      455--460.
   \bibitem{eM57a}
      Ernest~T. Moynahan,
      \emph{Ideals and congruence relations in
      lattices.} II,
      Magyar Tud. Akad. Mat. Fiz. Oszt. K\"{o}zl.
      \textbf{9} (1957), 417--434.
\end{thebibliography}
```

Each item listed in the bibliography can be referenced in the body of the article. You reference with the `\cite` command. The argument is the argument of the `\bibitem` command. So to reference Menuhin's article, type

```
\cite{gM68}
```

which typesets as [2] since Menuhin's article is the second in the list. So

> this result was first published in [2]

is typed as

```
this result was first published in~\cite{gM68}
```

How you write each label is up to you, subject only to the rule in Section 10.4.2, provided the labels are unique. I use the convention that the label for a `\bibitem` consists of the initials of the author and the year of publication. For example, a publication by Andrew B. Reich in 1987 would have the label `aR87` (a second publication by that author from that year would be `aR87a`). For joint publications, the label consists of the initials of the authors and the year of publication. For example, a publication by John Bradford and Andrew B. Reich in 1987 would have the label `BR87`.

You have to arrange the references in your document's `thebibliography` environment in the order you wish to see them. LaTeX only takes care of the numbering and the citations in the text.

4.3 On using LaTeX

Now that you are ready to type your first article, we give you some pointers on using LaTeX.

4.3.1 LaTeX error messages

You probably make a number of mistakes in your first article. These mistakes fall into the following categories:

1. Typographical errors, which LaTeX blindly typesets
2. Errors in mathematical formulas or in the formatting of the text
3. Errors in your instructions to LaTeX, that is, in commands and environments

Typographical errors can be corrected by viewing and spell checking the typeset article, finding the errors, and then editing the source file.

Mistakes in the second and third categories probably trigger errors during the typesetting process, such as the math errors in Section 3.2. Some of these mistakes may have to be corrected before your article can be completely typeset.

We now look at some examples of the third class of errors by deliberately introducing a number of mistakes into the source file of the article `intrart.tex` (in your `samples` folder, source file on pages 35–41, and shown typeset on pages 42–43), and examining the error messages that occur.

When LaTeX displays a `?` prompt, you can either try to continue typesetting the document by pressing Return, or type `x` to stop typesetting immediately. See Section D.4 for other options.

Experiment 1 In `intrart.tex`, go to line 20 by using your editor's `Go to Line` command and remove the closing brace so that it reads

```
\begin{abstract
```

When you typeset `intrart.tex`, LaTeX reports a problem:

```
Runaway argument?
{abstract In this note, we prove that there exist
\emph {complete-sim\ETC.
! Paragraph ended before \begin was complete.
<to be read again>
                   \par
l.26
```

Line 26 of the file is the line after `\end{abstract}`. The error message informs you that the name of the environment that ends before line 26 is not completed before the end of the paragraph. Press Return to tell LaTeX to typeset the remainder of the article, leaving out the abstract.

`Runaway argument?` is an error message that comes up often. It usually means that the argument of a command is either longer than expected or it contains material that the argument cannot accept. Most often a closing brace solves the problem, as in the experiment.

Experiment 2 Now correct line 20, then go to line 25 and change it from

```
\end{abstract}
```

to

```
\end{abstrac}
```

and typeset the article again. LaTeX informs you of another error:

```
! LaTeX Error: \begin{abstract} on input line 20
ended by \end{abstrac}.


l.25 \end{abstrac}
```

This is perfect. LaTeX correctly analyzes the problem and tells you where to make the change.

You may continue typesetting the article by pressing Return. LaTeX then gives you the message:

```
! Missing } inserted.
<inserted text>
                }
l.25 \end{abstrac}
```

The missing `}` inserted is the "special brace" `\end{abstract}` (more about this in Section 5.3.2). Press Return. LaTeX recovers from this error and the article is typeset correctly.

Experiment 3 Instead of correcting the error in line 25, comment it out with

```
% \end{abstrac}
```

and also comment out the four lines of the `figure` environment using block comment. Introduce an additional error in line 96. This line reads

```
and $\Theta$ is a complete congruence relation,
```

Change `\Theta` to `\Teta`:

```
and $\Teta$ is a complete congruence relation,
```

Now, when you typeset the article, LaTeX reports

```
! Undefined control sequence.
<recently read>\Teta

l.96 and $\Teta
               $ is a complete congruence relation,
```

Pressing Return results in the message

```
! LaTeX Error: \begin{abstract} on input line 20
ended by \end{document}.

Type  H <return>  for immediate help.
 ...

l.150 \end{document}
```

These two mistakes are easy to identify: `\Teta` is a misspelling of `\Theta`, and since `\end{abstract}` is missing, LaTeX is trying to match

```
\begin{abstract}
```

with

```
\end{document}
```

Now undo the changes you made to lines 25 and 96. Uncomment the figure.

Experiment 4 In line 42, delete the closing brace of the `\label` command:

```
\begin{definition}\label{D:P*
```

This results in a message for line 54, the blank line following the paragraph, that

```
! Paragraph ended before \label was complete.
```

This is easy to understand. You cannot begin a new paragraph within the argument of a `\label` command.

Undo the change to line 42.

Experiment 5 Add a blank line following line 58:

```
\langle \dots, 0, \dots, d, \dots, 0, \dots \rangle
```

This change results in the message

```
! Missing $ inserted.
<inserted text>
                $
l.59
```

There can be no blank lines within a displayed math environment. LaTeX catches the mistake, but the message itself is misleading. Pressing Return does not help; you cannot recover from the error. Delete the blank line.

Experiment 6 Add a $ somewhere in line 58 (such errors often occur when cutting and pasting formulas):

```
 \langle $\dots, 0, \dots, d, \dots, 0, \dots \rangle
```

You get the message:

```
! Display math should end with $$.
<to be read again>
                   \protect
l.58   \langle $\dots
                  , 0, \dots, d, \dots, 0, \dots \rangle
```

Maybe this could be more to the point?

Error messages from LaTeX are not always helpful, but there is always some information that can be gleaned from them. Try to identify the structure, that is, the command or environment, that causes the error—read the section of this book that describes that command or environment. This should help you correct the error. Keep in mind that the error could be quite far from the line LaTeX indicates, but it is always on or before that line in the source file.

If you have difficulty isolating a problem, block comment all but the paragraph you suspect might have problems. If necessary, split a large paragraph into smaller pieces.

Tip Typeset often.

To some extent, you can avoid having to isolate problems by following this tip. For instance, if I were to typeset *First Steps into LaTeX* [29], with the closing brace of the first `\caption` command on line 480 of the source file missing, I would get the error message

```
! Text line contains an invalid character.
l.1227 ...pletely irreducible^^?
```

where the reference is to line 1227, about 700 lines removed from the actual error. However, if the only thing I did before typesetting was to insert that figure with its incorrect caption command, at least I would know where to look for errors. If you make a dozen corrections and then typeset, you may not know where to start.

4.3.2 Logical and visual design

The typeset version of `intrart.tex` (pp. 42–43) looks impressive. To produce such articles, you need to understand that there are two aspects to article design: *visual* and *logical*.

As an example, let us look at a theorem from `intrart.tex` (see the typeset form of the theorem on page 43). You tell LaTeX that you want to state a theorem by using a `theorem` environment:

```
\begin{theorem}\label{T:P*}
Let $D_{i}$, $i \in I$, be complete distributive
lattices satisfying condition~\textup{(J)}.
Let $\Theta$ be a complete congruence relation on
$\Pi^{*} ( D_{i} \mid i \in I )$.
If there exist $i \in I$ and $d \in D_{i}$ with
$d < 1_{i}$ such that, for all $d \leq c < 1_{i}$,
\begin{equation}\label{E:cong1}
   \langle \dots, d, \dots, 0, \dots \rangle \equiv
   \langle \dots, c, \dots, 0, \dots \rangle
   \pod{\Theta},
\end{equation}
then $\Theta = \iota$.
\end{theorem}
```

The logical part of the design is choosing to define a theorem by placing material inside a `theorem` environment. For the visual design, LaTeX makes hundreds of decisions. Could you have specified all of the spacing, font size changes, centering, numbering, and so on? Maybe, but would you *want* to? And would you want to repeat that process for every theorem in your document?

Even if you did, you would have spent a great deal of time and energy on the *visual design* of the theorem rather than on the *logical design* of your article. The idea

behind LaTeX is that you should concentrate on what you have to say and let LaTeX take care of the visual design.

This approach allows you to easily alter the visual design by changing the document class (or its options, see Sections 11.5, 12.1.2, and 18.1). Section 11.1 provides some examples. If you code the visual design into the article—hard coding it, as a programmer would say—such changes are much harder to accomplish, for you and for the journal publishing the article.

4.4 Converting an article to a presentation

To produce a document in LaTeX for use as a presentation, you have to output it as a PDF file. You make your presentation using a PDF viewer such as Adobe Reader or print the pages of the PDF file on transparencies and use a projector.

So a *presentation* is a PDF file. To display the presentation, connect your computer to a projector. Open the PDF file in Adobe Reader, put it in full screen mode. Then project the presentation a page at a time by pressing the space bar or the forward and back arrow keys.

In LaTeX, you use a presentation package—really, a document class—to prepare such a PDF file. We deal with presentations in detail in Chapter 14, but as a quick introduction, we convert `intrart.tex` into a presentation.

For the conversion, we use the presentation package FoilTeX, while in Chapter 14 we discuss the `beamer` package. To use the FoilTeX package, we have to learn only one new command, `\foilhead`.

Open `intrart.tex`, save it as `intropres.tex` in the `work` folder. We introduce some changes to the document to prepare it for the conversion. Once you are satisfied with the changes made, the `tex` file created for FoilTeX is typeset so as to produce the PDF file. For `WinEdt`, click on the `PDF TeXify` icon. For `TeXShop`, just click on `Typeset`. For other TeX installations, check your user manual on how to create a PDF output.

4.4.1 Preliminary changes

Make the following changes in the preamble, top matter, and abstract.

1. Change the first line to
   `% Introductory presentation:intropres.tex`
2. Change the documentclass to `foils`.
3. Add the line
   `\usepackage{amsmath}`
   after the documentclass line. We have to do this because FoilTeX does not automatically load the AMS math package.

4. Delete the definitions of `theorem`, `lemma`, and `definition`. FoilTEX redefines these.

5. Copy the address into the `\author` command:

   ```
   \author{George~A. Menuhin\\
                Computer Science Department\\
                University of Winnebago\\
                Winnebago, MN 53714}
   ```

   and delete the `\address` command. This may seem strange, but it is necessary because FoilTEX is based on a legacy document class that does not have an `\address` command (see Chapter 12).

6. Move the abstract after the `\maketitle` command, as was customary in legacy document classes.

7. Add the `[scale=2]` option to the `\includegraphics` command, so the command becomes

   ```
   \centering\includegraphics[scale=2]{products}
   ```

   FoilTEX uses fonts in 20 point size, twice the usual size. So it is appropriate that we scale up the illustration to 200%.

So the new version is

```
% Introductory presentation:intropres.tex

\documentclass{foils}
\usepackage{amsmath}
\usepackage{amssymb,latexsym}
\usepackage{graphicx}

\begin{document}
\title{A construction of complete-simple\\
       distributive lattices}
\author{George~A. Menuhin\\
                Computer Science Department\\
                University of Winnebago\\
                Winnebago, MN 53714}
\date{March 15, 2006}

\maketitle
\begin{abstract}
   In this presentation, we prove that there exist
```

```
   \emph{complete-simple distributive lattices,}
   that is, complete distributive lattices
   with only two complete congruences.
\end{abstract}
```

Declarations in the body

In your LaTeX editor, perform four search and replace operations in the body of the article. Change all of the following:

1. `{theorem}` to `{Theorem}`
2. `{lemma}` to `{Lemma}`
3. `{definition}` to `{Definition}`
4. `{proof}` to `{Proof}`

FoilTeX defines and uses the capitalized versions.

Sectioning

Comment out all the `\section` commands. FoilTeX uses the `\foilhead` command to break the material into pages and also as a substitute for sectioning.

4.4.2 Making the pages

We cut the presentation into pages (transparencies or foils) by inserting as many page breaking commands of FoilTeX, `\foilhead{}`, as seems appropriate. The argument of the command becomes the "title" for the page. If the argument is empty, the page has no title.

Add the `\foilhead{The result}` command after the abstract. This ends the title page and adds the title The result to the next page.

See the `intropres.tex` document for all the other `\foilhead{}` commands we have added.

4.4.3 Fine tuning

We have eliminated the equation numbering, because it would make the equations too wide. Also, in a presentation, references to another page are not recommended. We made some additional changes to accommodate that we have fewer characters per line. Compare the documents `intrart.tex` and `intropres.tex` to see all the changes. Note how in the PDF file the fonts are automatically changed to sans serif, because sans serif text is easier to read when projected. The font size is 20pt, twice the size of the font in the article.

Of course, `intropres.tex` is not the most elegant presentation. But I hope it helps you to make your first presentation. On pages 57 and 58, we show the first two pages of this presentation.

There are, of course, a number of useful commands in FoilTEX in addition to the one we used, `\foilhead`. We did not even do justice to this one command. It has an optional argument to enlarge or shrink the space between the header and the body of the foil. So

```
\foilhead[-.5in]{A diagram}
```

shrinks that space by half an inch. This is especially useful with large diagrams.

For numerous other features of FoilTEX, see the user manual [33].

A construction of complete-simple distributive lattices

George A. Menuhin
Computer Science Department
University of Winnebago
Winnebago, MN 53714

March 15, 2006

Abstract

In this presentation, we prove that there exist *complete-simple distributive lattices,* that is, complete distributive lattices with only two complete congruences.

– Typeset by FoilTEX –

The result

In this presentation, we prove the following result:

Theorem 1. *There exists an infinite complete distributive lattice K with only the two trivial complete congruence relations.*

APPENDIX

A

Installation

In case you do not already have a LaTeX installation, in Sections A.1 and A.2, we describe how to install LaTeX on your computer, a PC or a Mac. The installation is much easier if you obtain TeX Live 2007 (or later) from the TeX Users Group, TUG (see Section E.2). It contains both the TeX implementations we discuss.

No installation is given for UNIX computers. The attraction of UNIX to its users is the incredibly large number of options, from the UNIX dialect, to the shell, the editor, and so on. A typical UNIX user downloads the code and compiles the system. This is obviously beyond the scope of this book. Nevertheless, TeX Live 2007 (or later) from the TeX Users Group supplies the compiled (binaries) of LaTeX for a number of UNIX variants.

First read Chapter 1, so that in this Appendix you recognize the terminology we introduce there. I will assume that you become sufficiently familiar with your LaTeX distribution to be able to perform the editing cycle with the sample documents.

A.1 LaTeX on a PC

On a PC, most mathematicians use MiKTeX and the editor WinEdt. So it seems appropriate that we start there.

A.1.1 Installing MiKTeX

If you made a donation to MiKTeX or if you have the TeX Live 2007 (or later) from the TeX Users Group, then you have a CD or DVD with the MiKTeX installer. Installation then is in one step and very fast. In case you do not have this CD or DVD, we show how to install from the Internet. To begin, go to the MiKTeX home page:

http://miktex.org

and under the Download/Install click on the version you want to install.[1] You are directed to the MiKTeX download page, where you click on Download MikTeX Net Installer. This takes you to a list of the download sites, called "mirrors". Choose one geographically close to you and click Download next to your pick. You are asked whether to Run or Save the installer application. Choose Save to save and now you have the setup application on your computer.

Run setup and the MiKTeX Setup Wizard should start automatically. Then click Next and choose the task, Download only. Click Next again to choose the size of the download and choose Complete MikTeX. Again you have to choose a download site, and click Next a few more times, then Start, and the download starts. When it is complete, almost 35,000 files later, click Close. Now you have the files you need in the next step.

The next task is installation. Run setup again, and up comes the Wizard. Click Next, and the task Install MikTeX is selected for you. Click Next, make sure you select Complete MikTeX. Click Next a few more times, select the default paper size, click Start, and the installation starts. When it is finished, click Close.

A.1.2 Installing WinEdt

You can download WinEdt from its Web site and use it for 30 days before you pay the license fee. We now install WinEdt from the CD you are sent after you pay the license fee. Go to License and Registration at

http://www.winedt.com

Put the WinEdt installer CD in the DVD drive. The WinEdt Setup Wizard starts automatically. After accepting the licence, click Next a few times until WinEdt is installed and then click Finish.

After installation, the
WinEdt Configuration Wizard

[1] We follow the instructions for MiKTeX 2.5. Hopefully, this will also assist you with later versions.

starts automatically. Click on the `File Associations` tab and click on `Modify file type associations...` under `Current User`, which is down the right side of the window, and then click `OK`. This gives all TeX files a lion icon and automatically associates them with `WinEdt` so that double clicking a TeX file automatically opens it in `WinEdt`. Clicking `OK` to close the Wizard.

`WinEdt`'s claim to fame is its incredible customizability. Once you become familiar with the basic operations, you can make `WinEdt` behave the way you like.

A.1.3 The editing cycle

In Section 1.2 you created the `work` folder for your work files. Start `WinEdt` by double clicking the `WinEdt` icon and open the file `note1.tex` in `work`, see the top half of Figure A.1. Observe:

- In the right-hand corner, under the X button, the close application button, there is a small black x. All windows of `WinEdt` have such an x, this is the *close window* button.
- There are two rows of icons. The seventh from the right in the first row, a darkened lion's head, is the `TeXify` button, use it to typeset your LaTeX file. If you are not sure what an icon represents, let the cursor hover over it, and a brief description appears.
- A blue arrow in the left margin points at the line where the cursor is.

Click on the `TeXify` icon. `MiKTeX` typesets `note1.tex` and produces another file, `note1.dvi`. The new file is displayed by the application `Yap`, which was automatically installed for you with `MiKTeX`. Rearrange the `WinEdt` and `Yap` windows. You should get an arrangement similar to the bottom half of Figure A.1.

A.1.4 Making a mistake

`WinEdt` tells you if there is a mistake in your source file. To see what happens, click on the `WinEdt` window, and add a `\` in line 11 of `note1.tex`, so that `personally` reads `\personally`. This makes `\personally` a command, which is a mistake (see Chapter 2). Click on the `TeXify` icon. We get the `TeXify ...` window, as in the top part of Figure A.2. At the `?` prompt, type `x` for "exit" and press Return.[2] You now see three windows, as shown in the bottom half of Figure A.2. The `Yap` window is mostly covered up. There is a new window, `note1.log`, the `log` window, in which the information from the `TeXify ...` window is recorded. In the `WinEdt` window, the blue arrow line pointer on the left indicates the offending line, and the mistake, `\personally`, is highlighted in red. Correct the mistake by deleting `\`, click on `TeXify`, and you are back in business.

[2] The user, recorded on Figure A.2, typed `s` for "scroll mode". LaTeX then completes the typesetting without stopping for errors (see Section D.4).

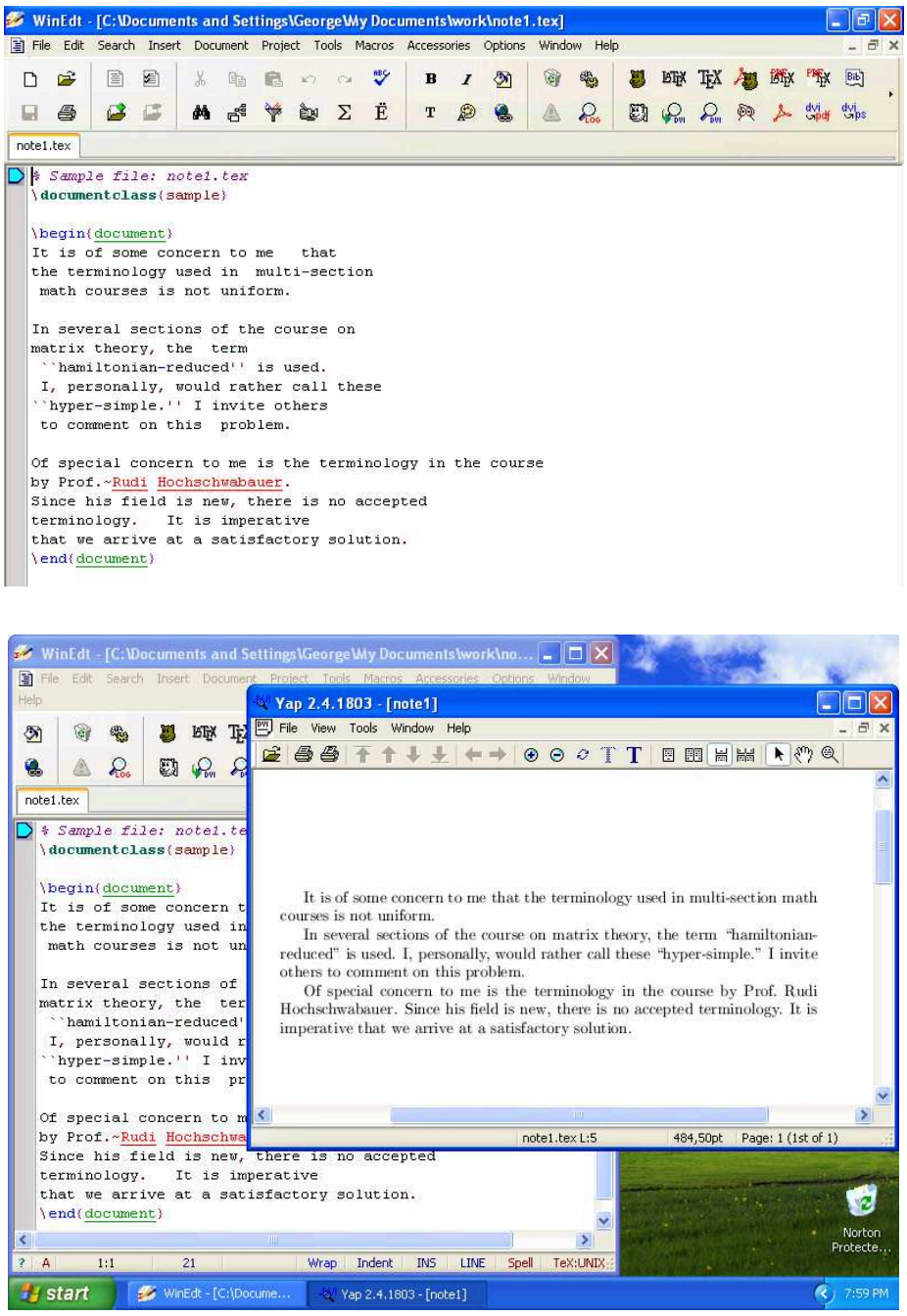


Figure A.1: `note1.tex` opened in `WinEdt` and typeset.

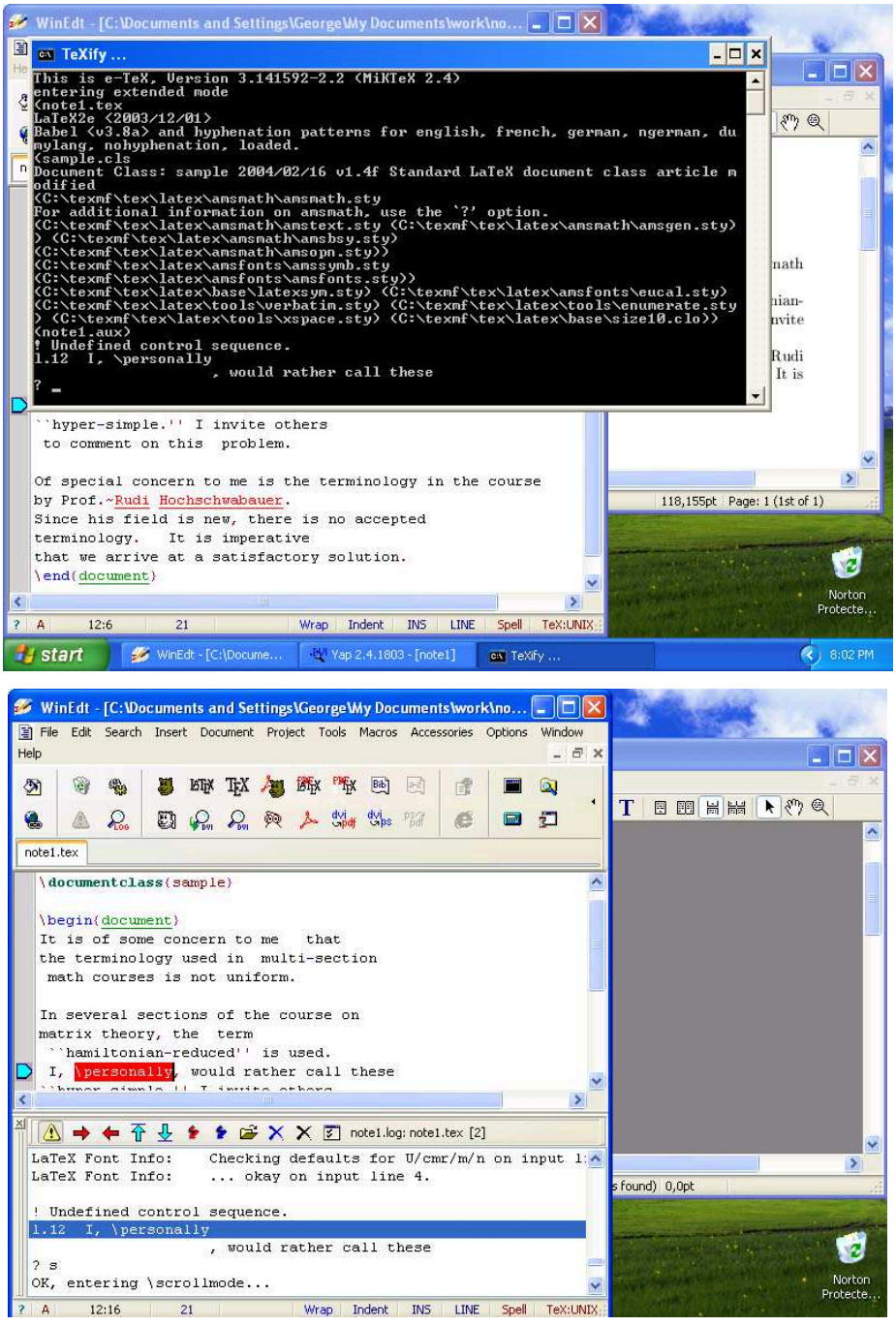


Figure A.2: The mistake identified and localized (showing the `log` window).

A.1.5 Three productivity tools

Now we see how the three productivity tools introduced in Section 1.4 are implemented in `WinEdt` and `MiKTeX`.

Synchronization This is known as *inverse search* in `WinEdt` and `MiKTeX`. To set it up, open `Yap`, for example, by `Texifying` the document `note1.tex`. Choose the menu item `View>Options>Inverse DVI Search`. A list now displays the editors detected. Select `WinEdt`, click `Apply`, and click `OK`. You are done.

- To jump from the typeset file in `Yap` to the source file in `WinEdt`, double click a word in the typeset file.
- To jump from the source file to the typeset file, click on the `DVI Search` icon, the magnifying glass with the green leaf attached to the handle. You then jump to the typeset file, where a marker indicates the beginning of the typeset version of the source line the cursor is on.

Block comment Select a number of lines in a source document and choose the menu option `Edit>Move/Fill>Insert comment`. Block uncomment is done with `Edit>Move/Fill>Remove comment`.

Go to line This is done with `Search>Go to Line...`

The icons and menu options all have keyboard equivalents. For instance, `Go to line` is Ctrl+`G` and `TeXify` is Ctrl+Shift and then press `x`.

A.1.6 An important folder

When using `MikTeX`, where do you put new style files and packages?

You can always place these files in your working folder. This is the simplest, but they will not be available when working with files outside that folder.

If you want these files accessible everywhere (from all folders), then make a new folder in the `LocalTeXFM` folder tree, say

```
C:\LocalTeXMF\MyStyles
```

and place these files there. In order for `MiKTeX` to find these files, you now need to update the `MiKTeX FileName Database`. Do this:

```
All Programs>MikTeX>MikTeX Options>Refresh FNDB
```

or access it directly via the drop down menus from within `WinEdt`:

```
Accessories>MikTeX Options
```

The best way to install new packages from CTAN is via the `Package Manager`, which is accessed via

All Programs>MikTeX>Package Manager

The Package Manager downloads the files from a CTAN mirror of your choice, then installs all files in their correct folders and updates MikTeX's filename database.

It is a good idea to run the MiKTeX Update Wizard on a regular basis to keep your system up to date:

All Programs>MikTeX>MikTeX Update Wizard

A.2 LaTeX on a Mac

A.2.1 Installations

For the Mac, we install MacTeX,[3] which consists of the TUG's TeX Live and Richard Koch's TeXShop. If you have TeX Live (see Section E.2), put the DVD in your computer's DVD drive and follow the simple instructions. In a few minutes you are done. Otherwise, go to

http://www.tug.org/mactex/

and in the Downloading section, click on MacTeX-2007. After downloading about 700 MBs, you get the MacTeX-2007 "disk image" that contains the mactex installer package. The disk image should open and the application MacTeX-2007.mpkg should start automatically. A few more clicks—as in all Mac installations—and you are done.

The spelling checker

Finally, get the spell checker cocoAspell by Anton Leuski by going to

http://people.ict.usc.edu/~leuski/cocoaspell/home.html

and clicking on the download link, cocoAspell. This downloads the disk image cocoAspell.dmg, containing the installation package cocoAspell.mpkg. Follow the same process as above to mount this. Then double click on the installer package and follow the instructions.

In the Apple menu, choose System Preferences..., where you find a new one, Spelling. Double click on it, and choose a dictionary. I use the dictionary English(United States). You should also select the filters you need. They are explained on the page you obtain by clicking on the Filter button. I selected Texinfo and TeX/LaTeX. You may have to restart the computer for the spelling checker to work.

To invoke the spelling checker, select a word and press Command+Shift and :. It suggests a correct spelling. You can also add words to the dictionary.

To learn more about the dictionaries, read the documents in

/Library/Application Support/cocoAspell/ aspell6-en-6.0-0/doc/

[3] We follow the instructions for MacTeX-2007. Hopefully, this will also assist you with later versions.

A.2.2 Working with TeXShop

Custom settings

In due course, you can fully customize TeXShop as an advanced user. But to begin, there are just a few things to do. In the TeXShop menu choose Preferences... to open the TeXShop Preferences. To set the default font for the source files in TeXShop, click on the Document tab. Under Document Font click Set..., which brings up the Font window. Choose a font and size for the source files that is easy on your eyes. I use Courier and 12. Close the Font window.

Make sure that under Editor all items except Auto Complete are selected.

Now click the Preview tab and in the Preview Window Magnification enter a number for the magnification to be used for viewing the typeset version—I use 150. Once you type in the number, press Set. For Default Mouse Mode, choose Select Text. For Default Page Style, choose Multi-Page.

Under After Window Resize, choose Fixed Magnification. Then click on OK. Close the TeXShop Preferences window.

When you become more familiar with TeXShop, you may want to revisit these settings.

Changing a document for TeXShop

If

- you use the graphicx package, see intrart.tex in the samples folder as an example, and
- your document contains illustrations that have been saved as EPS graphics and included with the \includegraphics command,

then add the line

```
\usepackage{epstopdf}
```

in the preamble below the line

```
\usepackage{graphicx}
```

If you have many illustrations, it is preferable to open all the illustrations with TeXShop or Preview and save them in PDF format. Also, make sure that the extensions are not given in the \includegraphics commands, that is,

```
\usepackage{graphicx}
\includegraphics{products.eps}
```

is changed to

```
\usepackage{graphicx}
\includegraphics{products}
```

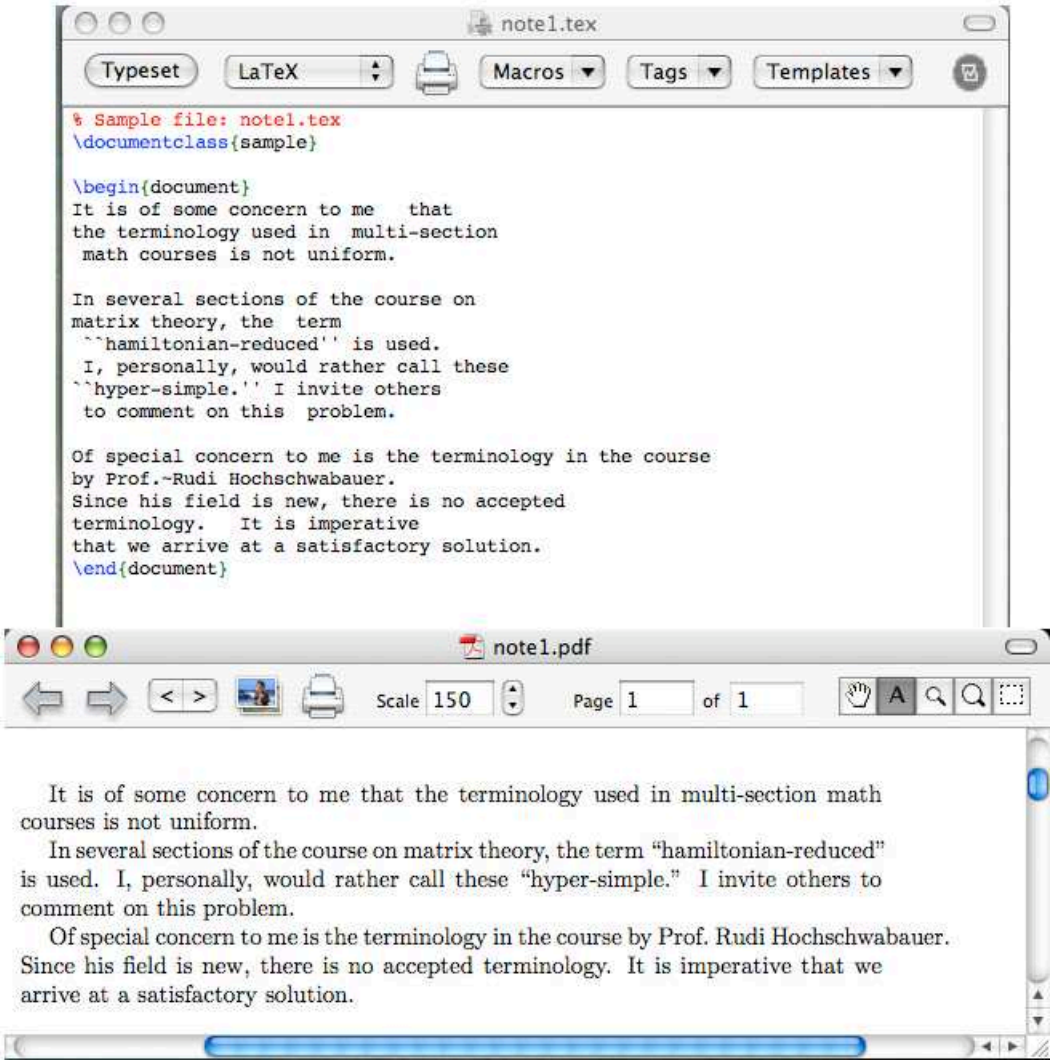


Figure A.3: The document `note1.tex`: the source and the typeset version.

A.2.3 The editing cycle

In your `Document` folder, you created the `work` folder (see page 4). We are going to work with the document `note1.tex` in the `work` folder.

To start `TeXShop`, double click on `TeXShop.app` in the `Applications` folder, select the menu `File>Open...`, and navigate to the folder

`Documents/work/note1.tex`

Open the document.

In the upper left corner of the source window, click the `Typeset` button. A second window opens, the *preview window*, showing `note1.pdf`, the typeset version of `note1.tex`. Unlike `WinEdt`, discussed in Section A.1, which produces a file called `note1.dvi`, `TeXShop` produces a PDF file, `note1.pdf`.

Figure A.3 shows the two windows. At the top, you see `TeXShop`'s seven menus. For this introduction we ignore all but two menu options. You should use the `Help` menu to learn more and the `Macros Help` in the `Help` window along with the `Macro Editor` (open it with `Macros>Open Macro Editor...`) to become more productive.

A.2.4 Making a mistake

`TeXShop` tells you if there is a mistake in your source file. Open `note1.tex` again and introduce a silly error, say, in the line `\documentclass{amsart}`, delete the closing brace, so it reads `\documentclass{amsart`. A new window—the third!—pops up, called `note1 console`, see Figure A.4. This is the `log` window (see Section 1.3). Click on the button `Goto Error` and the cursor is placed in the source document pretty close to the error. Now you can correct the error and typeset again.

A.2.5 Three productivity tools

Now we see how the three productivity tools introduced in Section 1.4 are implemented in `TeXShop`.

Synchronization Command-click on a word in the source window. The preview window shows the corresponding typeset phrase circled in red. Similarly, command-click on a word in the preview window and the corresponding source phrase is highlighted in yellow—it helps to click on text with no LaTeX commands close by.

Block comment Select a number of lines in a source document, and choose the menu option `Format>Comment`. All the lines, the whole block, are commented out. The reverse is done with `Format>Uncomment`.

Go to line This is done with `Edit>Line Number...`

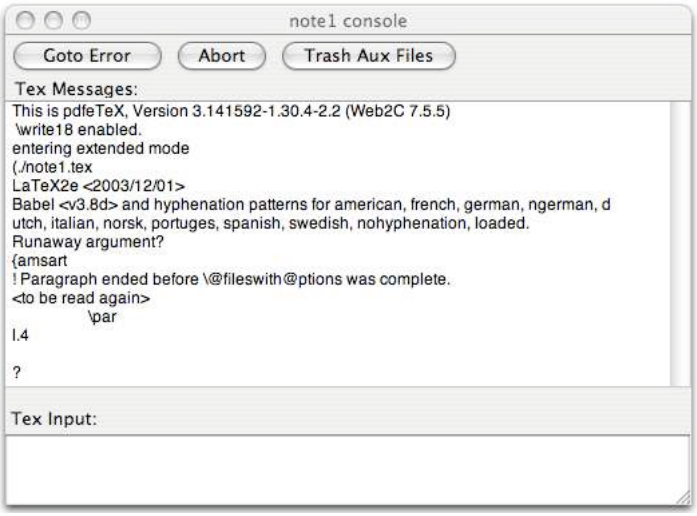


Figure A.4: The `note1.tex` console (`log` window).

Buttons and menu options all have keyboard equivalents. For instance, `Block comment` is Command + { and `Go to line` is Command + L.

A.2.6 An important folder

Create the `texmf` folder in the `Library` folder of your home folder—**not** the other `Library` folder, which is in the same folder as `Applications`. In `texmf`, create the `tex` folder, wherein you create the `latex` folder. Put all your personal (see Section 15.3) and additional `sty` files here.