More

Math Into LaTeX

5th Edition

George Grätzer

More Math Into LaTeX

5th Edition

Foreword by
Rainer Schöpf
LaTeX3 team

SPRINGER

To the **Volunteers**

without whose dedication,
over 25 years,
this book could not have been done

and to the young ones

Emma (10),

Kate (8),

Jay (3)

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 AMS-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 AMS-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 AMS-TEX code to LATEX-NFSS, thus giving birth to AMS-LATEX, released by the AMS at the August 1990 meeting of the International Mathematical Union in Kyoto.

AMS-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 AMS-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 LATEXe 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 AMS-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 AMS-LaTeX in Kyoto. The documentation he received explained that AMS-LaTeX is a LaTeX variant—read Lamport's LaTeX book to get the proper background. AMS-LaTeX is not AMS-TeX either—read Spivak's AMS-TeX book to get the proper background. The rest of the document explained in what way AMS-LaTeX differs from LaTeX and AMS-TeX. Talk about a steep learning curve ...

Luckily, George's frustration working through this nightmare was eased by his 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 AMS-LaTeX". This edition is more mature, but preserves what made his first book such a success. Just as in the first book, Part I, *Mission Impossible,* is a short introduction for the beginner. Chapter 1, *Short Course*, dramatically reducing the steep learning curve of a few weeks to a few hours in only 30 pages. Chapter 2, *And a few more things...* adds a few more advanced topics useful already at this early stage.

The rest of the book is a detailed presentation of everything 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. For formulas, he discusses the building blocks with examples, presents a *Formula Gallery,* and a *Visual Guide* for 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.

- 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

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

Rules which you must follow

Tips on what to be careful about and 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 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 as of 2004, also buy a second book, the LaTeX *Companion,* 2nd edition.

Rainer Schöpf
LaTeX3 team

Preface to the fifth edition

My book *Practical* LaTeX [42] was published last year. Many of the changes in this fifth edition are based on *Practical* LaTeX and on my articles "What Is New in LaTeX?" in the Notices of the American Mathematical Society [36]–[41] and [43].

Part I. Short Course of the fourth edition was revised under the title *Chapter 1. Short Course*. I renamed Part I: *Mission Impossible*. This part now has a second chapter: *And a few more things ...* The new Chapter 1 is what you absolutely, unquestionably must know to write your first TeX document. It's only 30 pages long, should not take more than a few hours to read and understand. No typing is necessary, the files you need are provided for you.

The new Chapter 2 adds a few more topics that is helpful to know such as the `aux` files, what is their role, how to handle them. It deals in some detail with error messages. Finally, it contains Brian Davey's list of LaTeX mistakes most often made by authors.

To create "vector graphics" illustrations (see page 349 for an example), many users switched to Till Tantau's Ti*k*Z package. We introduce Ti*k*Z in Chapter 13. We hope that the few commands we discuss are sufficient to get you started.

I carefully revised all the material in this book. One would think that this is not necessary in a fifth edition. But as Fred says, there are infinitely many typos in any book, and even our best efforts remove only finitely many. And so many of the links have changed...

Finally, I should mention that I renamed the awkward *user-defined commands* to *custom commands*. How come I have not thought of this before?

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 under the guidance of 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 page 4 and creating your first presentation such as the sample presentation—four slides of which you find in Figure 1.5—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 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 the Short Contents, the detailed Contents, and the Index.

Mathematicians find LaTeX very strange. A typical article in mathematics deals with a field defined by a few axioms, and the topic of the article needs only a few more. In contrast, LaTeX has hundreds of axioms. We try to ease the transition by introducing at the start as few commands as possible. For instance, we introduce presentations with only five new commands.

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, consider the formula $\int \sqrt{\alpha^2 + x^2}\, dx$. To mark it up 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.

The three layers

The markup language we shall discuss comes in three layers: TeX, LaTeX, and the AMS packages, described in detail in Appendix C. Most LaTeX installations 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 Windows computer, a computer running Microsoft Windows
- A Mac, 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.

What's in the book?

Part I is *Mission Impossible;* it helps you to get started quickly with LaTeX, to type your first articles, to make your first presentations, and it prepares you to tackle LaTeX in more depth in the subsequent parts.

Chapter 1 is the *Short Course*. You start writing your *first article*—as typeset on page 4—and prepare your *first presentation*—see some of the slides typeset on page 28. This chapter 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 also learn math markup, which is not so straightforward. 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 should cover most of your everyday needs. We also cover the elements of presentations with a simple example.

In **Chapter 2**, we explain how things work, the structure of LaTeX, the auxiliary files, the logical and visual design of an article, LaTeX error messages. Finally, we present a long list of dos and don't to help you write good LaTeX.

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

Chapters 3 and **4** introduce *text* and *displayed text.* Chapter 3 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 4 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 5** 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 6**. How to space them, how to build new ones; we introduce the new set of some 2,000 STIX math symbols. 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 7** 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 8**, you learn about the *structure* of a LaTeX document. The most important topics are *sectioning* and *cross-referencing*. In **Chapter 9**, we discuss the `amsart` *document class* for articles. In particular, I present the title page information. Chapter 9 also features `secondarticle.tex`,

a sample article for `amsart`, somewhat more advanced than `firstarticle.tex` typeset on page 4. 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 conclude this chapter with a brief description of the AMS distribution, the packages and document classes, of which `amsart` is a part.

In **Chapter 10** the most commonly used *legacy document classes* are presented, `article`, `report`, and `letter` (the `book` class is discussed in Chapter 17), 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 11**, discussing PDF *files, hyperlinks,* and the `hyperref` package. This prepares you for *presentations,* which are PDF files with hyperlinks. In **Chapter 12** we utilize the `beamer` *package* for making LaTeX presentations and **Chapter 13** introduces its sister package Ti*k*Z for illustrations.

Part V (**Chapter 14**) introduces techniques to *customize* LaTeX: custom commands and environments created by users, and command files. We present a sample command file, `newlattice.sty`, and a version of the second 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 15** and **16**), 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 17**. Better quality books have to use document classes designed by professionals.

You will probably find yourself referring to **Appendices A** and **B** time and again. They contain the *math and text symbol tables.* You can also find them in the `samples` file.

Appendix C relates some historical background material on LaTeX. It gives you some insight into how LaTeX developed and how it works. **Appendix D** discusses the many ways we can find LaTeX material on the *Internet.* **Appendix E** is a short introduction to the use of *PostScript fonts* in a LaTeX document. **Appendix F** briefly describes the use of LaTeX for languages other than American English.

LaTeX on an iPad is introduced in **Appendix G**.

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

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 custom 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 to create your own packages
- to discuss how to design beautiful documents by writing document classes

The definitive book on the first topic, as of 2004, is Frank Mittelbach and Michel Goosens's *The* LaTeX *Companion,* 2nd edition [56] (in collaboration 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 (`\newpage`)
  - Environments (`\align`)
  - Documents (`firstarticle.tex`)
  - Document classes (`amsart`)
  - Document class options (`draft`)
  - Folders or directories (`work`)
  - The names of *packages*—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. Compare: iff, iff, `iff`, and a larger version: iff, iff, `iff`.

- 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@me.com
Home page:
http://server.maths.umanitoba.ca/homepages/gratzer/

PART I

Mission Impossible

CHAPTER

1

Short course

It happens to most of us. We live a happy life without LaTeX and then, all of a sudden, we have to do something urgent that requires it.

If you are a student, maybe your professor turned to you and said "I need the solutions to these exercises typed up and distributed to the class by tomorrow" and the solutions are chock-full of formulas, difficult to do in Word.

Or you are a researcher whose documents have always been typed up by a secretary. You have to attend a conference and give a presentation. Your secretary is on vacation.

In my case, it was a letter (this was before e-mail) from the American Mathematical Society, in which they informed me that my paper, written in Word, was accepted for publication. The AMS will publish the paper in nine months. However, a LaTeX version would be published in three months! So I had to learn LaTeX in a hurry.

> The mission, should you choose to accept it, is to get started really fast in LaTeX. Our goal is to produce in LaTeX the little article printed on the next page.

Relax, this chapter will not self-destruct in five seconds.

A TECHNICAL RESULT FOR CONGRUENCES OF FINITE LATTICES

G. GRÄTZER

Abstract. We present a technical result for congruences on finite lattices.

1. Introduction

In some recent research, G. Czédli and I, see [1] and [2], spent quite an effort in proving that some equivalence relations on a planar semimodular lattice are congruences. The number of cases we had to consider was dramatically cut by the following result.

Theorem 1. *Let L be a finite lattice. Let δ be an equivalence relation on L with intervals as equivalence classes. Then δ is a congruence relation iff the following condition and its dual hold:*

(C_+) *If x is covered by $y, z \in L$ and $x \equiv y \pmod{\delta}$, then $z \equiv y + z \pmod{\delta}$.*

2. The proof

We prove the join-substitution property: if $x \leq y$ and $x \equiv y \pmod{\delta}$, then

$$x + z \equiv y + z \pmod{\delta}. \tag{1}$$

Let $U = [x, y + z]$. We induct on $\operatorname{length} U$, the length of U.

Let $I = [y_1, y + z]$ and $J = [z_1, y + z]$. Then $\operatorname{length} I$ and $\operatorname{length} J < \operatorname{length} U$. Hence, the induction hypothesis applies to I and $\delta\restriction I$, and we obtain that $w \equiv y + w \pmod{\delta}$. By the transitivity of δ, we conclude that

$$z_1 \equiv y + w \pmod{\delta}. \tag{2}$$

Therefore, applying the induction hypothesis to J and $\delta\restriction J$, we conclude (1).

References

[1] G. Czédli, *Patch extensions and trajectory colorings of slim rectangular lattices.* Algebra Universalis **88** (2013), 255–280.

[2] G. Grätzer, *Congruences of fork extensions of lattices.* Acta Sci. Math. (Szeged), **57** (2014), 417–434.

Department of Mathematics, University of Manitoba, Winnipeg, MB R3T 2N2, Canada
E-mail address, G. Grätzer: gratzer@me.com
URL, G. Grätzer: http://tinyurl.com/gratzerhomepage

Date: March 21, 2015.
2010 *Mathematics Subject Classification.* Primary: 06B10.
Key words and phrases. finite lattice, congruence.

1.1 Getting started

1.1.1 Your LaTeX

Are you sitting in front of your computer that has a LaTeX implementation? If you use a UNIX computer, you surely are. If you are in front of a Windows computer or a Mac, point your Internet browser at `tug.org`. Choose to download MikTeX for a Windows computer and MacTeX for a Mac. Follow the easy instructions (and be patient, these are big downloads) and you are done.

Even better, find a friend who can help.

1.1.2 Sample files

We work with a few sample documents. Download them from `CTAN.org`, search for `MiL5`, or go to the Springer page for this book, and click on the

`Extra files`

I suggest you create a folder, `samples`, on your computer 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 created.*

One of the sample files is `sample.cls`. Make sure it is in the `work` folder when you typeset a sample document.

1.1.3 Editing cycle

Watch a friend type a document in LaTeX and learn the basic steps.

1. *A text editor is used to create a* LaTeX *source file*. A source file might look like this:

```
\documentclass{amsart}
\begin{document}
Then $\delta$ is a congruence relation. I can type formulas!
\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*:

   Then δ is a congruence relation. I can type formulas!

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 viewed/printed.* View the typeset version as a pdf file or print it to get a paper version.

If LaTeX finds a mistake when typesetting the source file, it records this in the *log file*. The *log window* (some call it *console*) displays a shorter version.

Various LaTeX implementations have different names for the source file, the text editor, the typeset file, the typeset window, the log file, and the log window. Become familiar with these names, so you can follow along with our discussions.

1.1.4 Typing the source file

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

This typesets as

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

In this sentence, the first part

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

is text. Then

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

is a formula

```
), and
```

is text again. Finally,

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

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{center}
\end{center}
```

enclose a center environment; the *contents* (the text typed between these two commands) are centered when typeset.

In practice, text, formulas, 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 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.

1.2 The keyboard

The following keys are used to type the source file:

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

You can also use the following punctuation marks:

```
,  ;  .  ?  !  :  `  '  -
```

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

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, the dollar sign, $, is typed as `\$`, the underscore, _, is typed as `\_`, and the percent sign, %, is typed as `\%`. Only @ requires no special command, type `@` to print @; see Sections 3.1.2 and B.4.

There are also commands to produce composite characters, such as accented characters, for example ä, which is typed as `\"{a}`. LaTeX prohibits the use of other keys on your keyboard unless you have special support for it. See the text accent table in Sections 3.4.7 and B.2. If you want to use accented characters in your source file, then you must use the `inputenc` package.

Tip The text accent table looks formidable. Don't even dream of memorizing it. You will need very few. When you need a text accent, look it up. I know only one: `\"a` (LOL). If you use a name with accented characters, figure out once how to type it, and then any time you need it you can just copy and paste (chances are that the name is in your list of references).

1.3 Your first text notes

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.

To produce this typeset document, create a new file in your `work` folder with the name `textnote1.tex`. Type the following, including the spacing and linebreaks shown, but not the line numbers:

```
1   % Sample file: textnote1.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  \end{document}
```

Alternatively, copy the `textnote1.tex` file from the `samples` folder (see page 5).

The first line of `textnote1.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 document, 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*, `sample` (the special class we provided for the sample documents), which controls how the document is formatted.

The text of the note is typed within the `document` environment, that is, between `\begin{document}` and `\end{document}`.

Now typeset `textnote1.tex`. You should get the typeset document as shown. 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 document class and the markup commands. LaTeX notices when you put a blank space in the text, but it ignores *how many blank spaces* have been typed. 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. There is also a command for a *new paragraph*: `\par`.

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). The `~` (tilde) command also forces an interword space, but with a difference: it keeps the words on the same line. This command produces a *tie* or *nonbreakable space*.

Note that on lines 11 and 13, the left double quotes is typed as two left single quotes and the right double quote is typed as two right single quotes, apostrophes.

We numbered the lines of the source file for easy reference. Sometimes you may want the same for the typeset file. This is really easy. Just add the two lines

```
\usepackage{lineno}
\linenumbers
```

after the `\documentclass` line and you get:

1 It is of some concern to me that the terminology used in multi-section math
2 courses is not uniform.
3 In several sections of the course on matrix theory, the term "hamiltonian-
4 reduced" is used. I, personally, would rather call these "hyper-simple". I invite
5 others to comment on this problem.

Next, we produce the following note:

January 5, 2015

From the desk of George Grätzer

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

`George_Gratzer@yahoo.com`

Type the source file, without the line numbers. Save it in your `work` folder as `textnote2.tex` (`textnote2.tex` can also be found in the `samples` folder):

```
1   % Sample file: textnote2.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}
9
10  February 7--21 \emph{please} use my
11  temporary e-mail address:
12  \begin{center}
13     \texttt{George\_Gratzer@yahoo.com}
14  \end{center}
15  \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 (see also Section 3.4.8).
- The environments to *right justify* (lines 5–7) and *center* (lines 12–14) text.
- The commands to change the text style, including the `\emph` command (line 10) to *emphasize* text, the `\textbf` command (line 8) for **bold** text (text bold font), and the `\texttt` command (line 13) to produce `typewriter style` text. These are *commands with arguments.*
- The form of the LaTeX commands. As we have noted already, 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.

Tip `textnote2.tex` is a file name but `textbf1` is not a command name. `\textbf1` typesets as **1**. Let's look at this a bit more closely. `\textbf` is a valid command. If a command needs an argument and it is not followed by braces, then it takes the next character as its argument. So `\textbf1` is the command `\textbf` with the argument `1`; it typesets as **1**.

- The multiple role of hyphens: Double hyphens are used for number ranges. For example, `7--21` (in line 10) 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.
- Special rules for special characters (see Section 1.2), for *accented characters*, and for some *European characters.* For instance, the accented character ä is typed as `\"{a}`. (But I confess, I always type my name as `Gr\"atzer` without the braces.)

See Section 3.4 for more detail. In Appendix B, all the text symbols are organized into tables. We also have the `SymbolTables.pdf` in the `samples` folder.

Tip Keep `SymbolTables.pdf` handy on your computer!

1.4 Lines too wide

LaTeX reads the text in the source file one line at a time and typesets the entire paragraph when the end of a paragraph is reached. Occasionally, LaTeX gets into trouble when trying to split the paragraph into typeset lines. To illustrate this situation, modify `textnote1.tex`. In the second sentence, replace `term` by `strange term`. Now save this modified file in your `work` folder using the name `textnote1bad.tex` (or copy the file from the `samples` folder).

Typesetting `textnote1bad.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.

The first line of paragraph two is too wide. 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-
```

It informs you that 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. Then it identifies 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.*

What to do, when a line is too long?

Tip Your first line of defense: reword the offending line. Write

```
The strange term ‘‘hamiltonian-reduced’’ is used
in several sections of the course on matrix theory.
```

and the problem goes away.

Your second line of defense: insert one or more *optional hyphen commands* (`\-`), which tell LaTeX where it can hyphenate the word. Write:

```
hamil\-tonian-reduced
```

1.5 A note with formulas

In addition to the regular text keys and the 13 special keys discussed in Section 1.2, two more keys are used to type formulas: < and >. The formula $2 < |x| > y$ (typed as `$2 < |x| >y$`) uses both. Note that such a formula, called *inline*, is enclosed by a pair of $ symbols.

We begin typesetting formulas 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 formula note, create a new document with your text editor. Name it `formulanote.tex`, place it in the `work` folder, and type the following, without the line numbers (or simply copy `formulanote.tex` from the `samples` folder):

```
1  % Sample file: formulanote.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 formulas in LaTeX.

- There are two kinds of math formulas and environments in `formulanote.tex`:
  - *Inline* formulas; they open and close with `$` or open with `\(` and close with `\)`.
  - *Displayed* math environments; they open with `\[` and close with `\]`. (We will introduce many other displayed math environments in Section 1.7 and Chapter 7.)
- LaTeX uses its own spacing rules within math environments, and completely ignores the white spaces 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 1.6 and 5.4.6.

  The white space that you add when typing formulas is important only for the readability of the source file.
- 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 A; 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.

- There is no blank line before a displayed formula!

Tip Keep in mind that many spaces equal one space in text, whereas your spacing is ignored in formulas, unless the space terminates a command.

1.6 The building blocks of a formula

A formula (inline or displayed) is built from components. We group them as follows:

Arithmetic
Binomial coefficients
Congruences
Delimiters
Ellipses
Integrals
Math accents
Matrices
Operators
Roots
Text

In this section, I describe each of these groups, and provide examples illustrating their use. Read carefully the groups you need!

Arithmetic We type the arithmetic operations $a + b$, $a - b$, $-a$, a/b, and ab in the natural way: `$a + b$`, `$a - b$`, `$-a$`, `$a / b$`, and `$a b$` (the spaces are typed only for readability).

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 formulas $a \cdot b$ and $a \times b$ are typed as `$a \cdot b$` and `$a \times b$`.

Displayed fractions, such as

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

are typed with `\frac`:

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

Subscripts and superscripts Subscripts are typed with _ 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 $a_1 0$.

There is one symbol, the prime (`'`), that is automatically superscripted in a formula. To get $f'(x)$, just type `$f'(x)$`. (On many keyboards, the symbol on the key looks like this: `)

See Section 5.4.1 for more detail.

Binomial coefficients Binomial coefficients are typeset with the `\binom` command. `\binom{a}{b + c}` is here inline: $\binom{a}{b+c}$, whereas

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

is the displayed version.

See Section 5.4.2 for more detail.

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	`$a \equiv v \pod{\theta}$`

See Section 5.6.2 for more detail.

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 formula inside the parentheses; sometimes the result is pleasing, sometimes not.

We dedicate Section 5.5 to this topic.

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

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

or as *centered dots*:

$x_1 + \cdots + x_n$ is typed as

```
$x_{1} + \dots + x_{n}$
```

Use `\cdots` and `\ldots` if `\dots` does not work as expected.

See Section 5.4.3 for more detail.

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.

The formula looks bad without the spacing command: $\int_0^\pi \sin x dx = 2$.

See Section 5.4.4 for more complicated integrals.

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 Section 5.7 for more detail. See Sections 5.7 and A.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, `\\`.

Tip Do not end the last row with a new line command.

The `matrix` environment has to appear within a formula, as a rule, in a displayed formula. It can be used in the `align` environment discussed in Sections 1.7.3 and 7.5.

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. There is also `bmatrix` for square brackets.

See Section 7.7.1 for a listing of all the matrix variants and Sections 5.5 and A.6 for lists of delimiters.

Operators To typeset the sine function, $\sin x$, type `$\sin x$`. Note that `$sin x$` would be typeset as $sinx$—-how awful. LaTeX calls `\sin` an *operator*. Sections 5.6 and A.7 list a number of operators. Some are just like `\sin`. Others produce a more complex display, for example,

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

is typed as

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

See Section 5.6 for more detail.

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

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

are typed as

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

Sum and product are examples of *large operators.* They are typeset larger in displayed math than in an inline formula. They are listed in Sections 5.6.3 and A.7.1. See Section 5.6.3 for more detail.

Roots `\sqrt` produces a square root. `$\sqrt{a + 2b}$` typesets as $\sqrt{a+2b}$. The n-th root, $\sqrt[n]{5}$, requires the use of an *optional argument,* which is specified in brackets: `$\sqrt[n]{5}$`. See Section 5.4.5.

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

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

is typed as

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

where `\quad` is a spacing command.

See Section 5.4.6 for more detail.

1.7 Displayed formulas

1.7.1 Equations

The `equation` environment creates a displayed 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:firstIntegral}
  \int_{0}^{\pi} \sin x \, dx = 2
\end{equation}
```

The equation number, which is automatically generated, depends on how many numbered displayed formulas occur before the given equation. You can choose to have equations numbered within each section—(1.1), (1.2), ..., in Section 1; (2.1), (2.2), ..., in Section 2; and so on—by including, in the preamble (see Sections 1.8 and 5.3), the command

```
\numberwithin{equation}{section}
```

You can choose to have the equation numbers on the right; see the `reqno` option of the `amsart` document class in Section 10.1.2.

The `equation*` environment is the same as the displayed formula opened with `\[` and closed with `\]` we discussed in Section 1.5. Sometimes you may want to use `equation*` for the ease of deleting the `*`-s if you wish.

1.7.2 Symbolic referencing

To reference a formula without having to remember a number—which can 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 `firstIntegral`, and used the convention that the label of an equation starts with `E:`, so that the complete `\label` command is `\label{E:firstIntegral}`.

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 18.

type (see Sections 1.3 and Section 3.4.3 for `~`)

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

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

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

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

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. For bibliographic references, LaTeX uses the `\bibitem` command to define a bibliographic item and the `\cite` command to cite it.

Tip For renumbering to work, you have to typeset **twice**.

Tip It is a good idea to check the LaTeX warnings periodically in the `log` file. If you forget to typeset the source file twice when necessary, LaTeX issues a warning.

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

```
LaTeX Warning: Reference 'E:FirstIntegral' 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 the argument of `\cite` is misspelled, you get **[?]** and similar warnings.

Check the **Tip** on page 70.

Absolute referencing

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 are *absolute.* This equation is *always* referred to as (Int). Equation numbers, on the other hand, are *relative,* they may change when the file is edited.

1.7.3 Aligned formulas

LaTeX has many ways to typeset multiline formulas. We discuss three constructs in this section: *simple alignment, annotated alignment,* and *cases*. For more constructs, see Chapter 7.

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

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

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

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

```
\begin{align}
  r^{2}  &= s^{2} + t^{2},       \label{E:Pyth}\\%\eqref{E:Pyth}
  2u + 1 &= v + w^{\alpha}.      \label{E:alpha}%\eqref{E:alpha}
\end{align}
```

Figure 1.1 may help visualize the placements of the ampersands.

Tip In this displayed formula, \\ is a *line separator*, not a new line command. Do not place a \\ to terminate the last line!

```
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}
```

alignment points of formulas

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

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

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

Figure 1.1: Simple alignment: source and typeset.

These formulas are numbered (2) and (3) 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,

$$h(x) = \int \left(\frac{f(x) + g(x)}{1 + f^2(x)} + \frac{1 + f(x)g(x)}{\sqrt{1 - \sin x}} \right) dx \tag{4}$$
$$= \int \frac{1 + f(x)}{1 + g(x)}\, dx - 2\tan^{-1}(x - 2)$$

is typed as follows:

```
\begin{align}
  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\label{E:longInt}\\%\eqref{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 the `&` *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 can want to reference with `\ref`, `\eqref`, or `\pageref`.

Annotated alignment

Annotated alignment allows you to align formulas and their annotations, that is, explanatory text, separately:

$$
\begin{aligned}
x &= x \wedge (y \vee z) && \text{(by distributivity)}\\
&= (x \wedge y) \vee (x \wedge z) && \text{(by condition (M))}\\
&= y \vee z
\end{aligned}
\tag{5}
$$

This is typed as

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

Figure 1.2 may help visualize the placements of the ampersands.

Rule ■ **Annotated alignment**
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: `&&`.

1.7.4 Cases

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

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

It 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}
\]
```

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

1.8 The anatomy of a document

To begin, we use the sample document `firstarticle.tex` (in the `samples` folder) to examine the anatomy of an document.

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

```
\begin{document}
```

The *body* is the contents of the `document` environment. For a schematic view of a document, see Figure 1.3.

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, see Section 8.2) that must be placed in the preamble if they are used, but such commands do not have to be present in every document.

Here is the preamble and top matter of `firstarticle`:

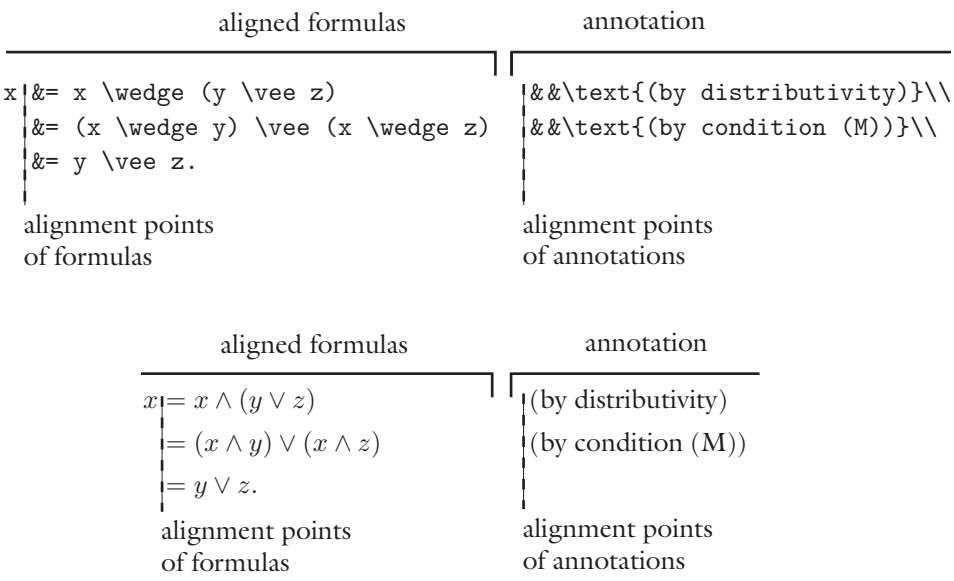


Figure 1.2: Annotated alignment: source and typeset.

```
%First document, firstarticle.tex
\documentclass{amsart}
\usepackage{amssymb,latexsym}

\newtheorem{theorem}{Theorem}

\begin{document}
\title{A technical result\\ for congruences of finite lattices}
\author{G. Gr\"atzer}
\address{Department of Mathematics\\
  University of Manitoba\\
  Winnipeg, MB R3T 2N2\\
  Canada}
\email[G. Gr\"atzer]{gratzer@me.com}
\urladdr[G. Gr\"atzer]{http://tinyurl.com/gratzerhomepage}
\date{March 21, 2015}
```

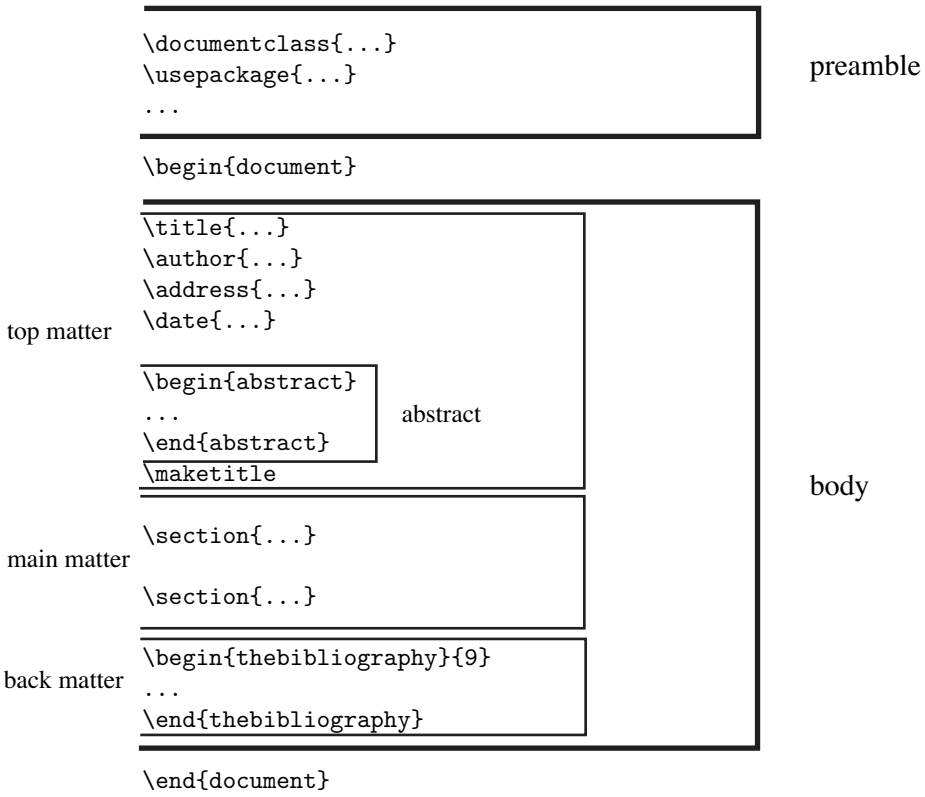


Figure 1.3: A schematic view of a document.

```
\subjclass[2010]{Primary: 06B10.}
\keywords{finite lattice, congruence.}
\maketitle

\begin{abstract}
We present a technical result for congruences on finite lattices.
\end{abstract}
```

You find the source file, `firstarticle.tex`, in the `samples` folder and the typeset document on page 4.

To simplify the discussion in Part I, we discuss only one document class for articles: `amsart`. You may come across its predecessor, `article`, which handles a limited set of commands for the preamble and the top matter and displays them differently. We shall discuss in detail the `amsart` document class in Chapter 9. For the `article` document class, see Section 10.1.

1.9 Your own commands

Over time, LaTeX can be adjusted to fit your needs. You add packages to enable LaTeX to do new things (such as the `graphicx` package, see Sections 1.10 and 8.4.3) and introduce your own commands to facilitate typing and make the source file more readable.

We can add two new commands to the sample article `firstarticle.tex`:

```
\newcommand{\pdelta}{\pmod{\delta}}
\DeclareMathOperator{\length}{length}
```

So instead of

```
$x \equiv y \pmod{\delta}$+
```

we can type

```
$x \equiv y \pdelta$
```

and instead of `length\,$U$`, we can type `$\length U$` (see Section 14.1.6). Notice how the spacing is now done by LaTeX!

We'll dedicate Chapter 14 to customizing LaTeX.

1.10 Adding an illustration

"And what is the use of a book," thought Alice, "without pictures or conversations?" I am not sure what to suggest about conversations, but illustrations we can tackle with ease. Let us add an illustration, `covers.pdf` to `firstarticle`. First, add

```
\usepackage{graphicx}
```

as the fourth line of the document, to the preamble. This will enable LaTeX to tackle illustrations. Secondly, add the following lines to `firstarticle.tex`, say, as the second paragraph of the introduction:

```
\begin{figure}[hbt]
{\centering\includegraphics{covers}}
\caption{Theorem~\ref{T:technical} illustrated}\label{F:Theorem}
\end{figure}
```

We place the illustration `covers.pdf` in the same folder as `firstarticle.tex`. That's it. You find `covers.pdf` and `firstarticleill.tex` in the `samples` folder.

Tip Make sure that the `\label` command follows the `\caption` command! You may have hard to explain troubles otherwise.

See Section 8.4.3 for more information.

Most people in my field used the vector graphics application Adobe Illustrator to produce the PDF files for illustrations. Quite recently, it became prohibitively expensive. Luckily, many reasonably priced alternatives are available. In Chapter 13, we discuss an alternative, Ti*k*Z, built for LaTeX. Inkspace is an alternative, available for all platforms.

1.11 The anatomy of a presentation

Chances are, one of your first exposures to LaTeX was watching a *presentation*. The presenter used a pdf document produced by LaTeX and opened it with Adobe Reader. He went from "slide" to "slide" by pressing the space bar. Figures 1.4 and Figure 1.5 show four slides of a presentation.

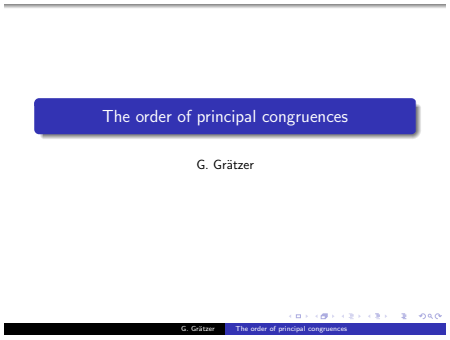
In LaTeX, you use a presentation package—really, a document class—to prepare the PDF file. We use Till Tantau's BEAMER.

Here are the first few lines—the preamble and the Title slide—of the source file of our sample presentation, `firstpresentation.tex` (see `firstpresentation.tex` in the `samples` folder, along with `Louisville.tex`, the full presentation):

```
\documentclass[leqno]{beamer}
\usetheme{Warsaw}

\DeclareMathOperator{\Princ}{Princ}

\begin{document}
\title{The order of principal congruences}
\author{G. Gr\"atzer}
\date{}
\maketitle
```



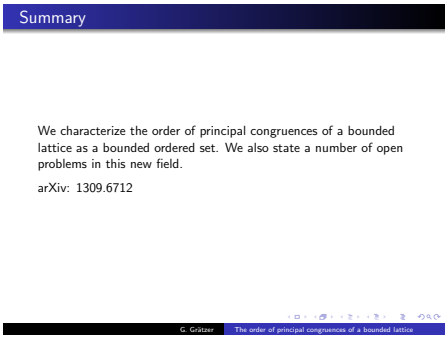
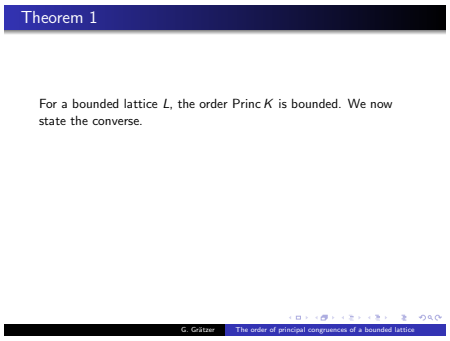


Figure 1.4: The Title slide (Slide 1) and Slide 2



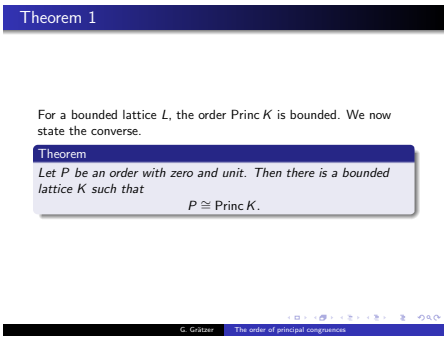


Figure 1.5: Slides 3 and 4

`\usetheme{Warsaw}` provides a flavor. It is followed by the Title slide, providing the title and the author.

The `\title` command may be longer, it may contain all the additional information you may want to display. Here is the `\title` command of `Louisville.tex`:

```
\title[The order of principal congruences of a bounded lattice]
{The order of principal congruences\\
of a bounded lattice.\\
AMS Fall Southeastern Sectional Meeting\\
University of Louisville, Louisville, KY\\
October 5-6, 2013}
```

Note that the `\title` has two parts. The first, in `[]`, is the short title, repeated in the bottom line on every slide. The second, in `{}`, is the title for the front page.

The rest of the presentation source file is divided into two *frames* with the structure:

```
\begin{frame}
\frametitle{}
\end{frame}
```

Each frame produces a "slide" (or more). Here is the first frame:

```
\begin{frame}
\frametitle{Summary}
We characterize the order of principal congruences
of a bounded lattice
as a bounded ordered set.
We~also state a number of open problems in this new field.
\medskip

arXiv: 1309.6712
\end{frame}
```

The command `\frametitle` gives the slide its title: Summary, see Slide 2 in Figure 1.4. In the body of the `frame`, you type regular LaTeX.

To produce Slides 3 and 4, it would be natural to try

```
\begin{frame}
\frametitle{Theorem 1}
For a bounded lattice $L$, the order $\Princ K$ is bounded.
We now state the converse.
\end{frame}

\begin{frame}
\frametitle{Theorem 1}
For a bounded lattice $L$, the order $\Princ K$ is bounded.
We now state the converse.
\begin{theorem}
Let $P$ be an order with zero and unit.
Then there is a bounded lattice~$K$ such that
\[
   P \cong \Princ  K.
\]
If $P$ is finite, we can construct $K$ as a finite lattice.
\end{theorem}
\end{frame}
```

which produces the two frames of Figure 1.6.

This is really jarring to watch. The two lines of the new Slide 3 jump up more than two lines as they transition to Slide 4.

Here is how we produce Slides 3 and 4 of Figure 1.5:

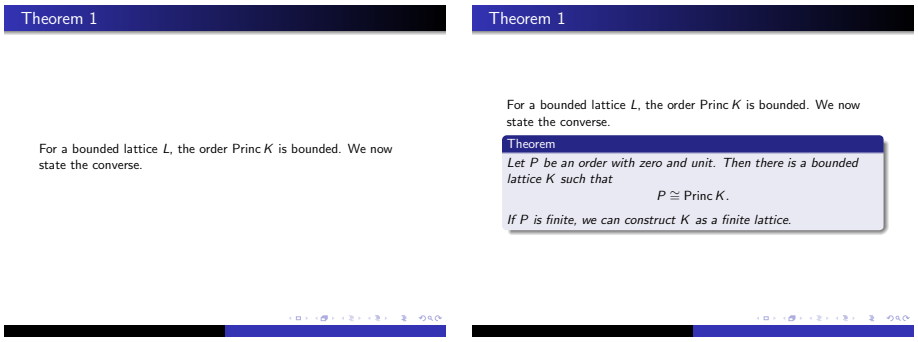


Figure 1.6: Slides 3 and 4, first try

```
\begin{frame}
\frametitle{Theorem 1}
For a bounded lattice $L$, the order $\Princ K$ is bounded.
We now state the converse.
\pause
\begin{theorem}
Let $P$ be an order with zero and unit.
Then there is a bounded lattice~$K$ such that
\[
   P \cong \Princ  K.
\]
If $P$ is finite, we can construct $K$ as a finite lattice.
\end{theorem}
\end{frame}
\end{document}
```

There is only one new command to learn: `\pause`; it produces from this frame **two** slides.

The `\pause` in this frame splits the contents of the frame into two parts. The first slide is typeset from the first part as if the second part was also present. The second slide is typeset from both parts. So the transition from the first slide to the second is smooth, see Figure 1.5.

You can have more than one `\pause` in a frame. Use `\pause` also to display a list one item at a time.

Chapter 12 discusses BEAMER in more detail.

CHAPTER

2

And a few more things...

If life was perfect, we would not need this chapter. You would write perfect LaTeX, based on Chapter 1, no need to study how LaTeX works, what error messages mean... But life is not perfect, you will make mistakes, LaTeX will send messages, plain and mysterious.

In this chapter, we briefly explain how things work, the structure of LaTeX, the auxiliary files, the logical and visual design of an article, LaTeX error messages. See Appendix C for more detail. Finally, we present a long list of dos and don't to help you write good LaTeX.

2.1 Structure

LaTeX's core is a programming language called TeX, created by Donald E. Knuth, which provides low-level typesetting instructions. TeX comes with a set of fonts called *Computer Modern* (CM). The CM fonts and the TeX programming language form the foundation of a typical TeX system. TeX is extensible—new commands can be defined in terms of more basic ones. LaTeX is one of the best known extensions of TeX.

The visual layout of a LaTeX document is primarily determined by the *document class*, such as `amsart`, `article` for articles, `amsbook`, `book` for books. Many journals,

Figure 2.1: The structure of LaTeX.

publishers, and schools have their own document classes for formatting articles, books, and theses.

Extensions of LaTeX are called *packages*. They provide additional functionality by adding new commands and environments, or by changing the way previously defined commands and environments work. It is essential that you find the packages that make your work easier. *The* LaTeX *Companion,* 2nd edition [56] discusses a large number of the most useful packages as of 2004.

The structure of LaTeX is illustrated in Figure 2.1. This figure suggests that in order to work with a LaTeX document, you first have to install TeX and the CM fonts, then LaTeX, and finally specify the document class and the necessary packages. The packages must include `amsmath`, `amsthm`, and so on. Of course, your LaTeX installation already includes all of these.

2.2 Auxiliary files

Figure 2.2 illustrates the steps in the production of a typeset document.

You start by opening an existing LaTeX source file or creating a new one with a text editor. For this discussion, the source file is called `myart.tex`. Once the source file is ready, you typeset it. Depending on the document class options you choose and the packages the document loads, you end up with at least three additional files:

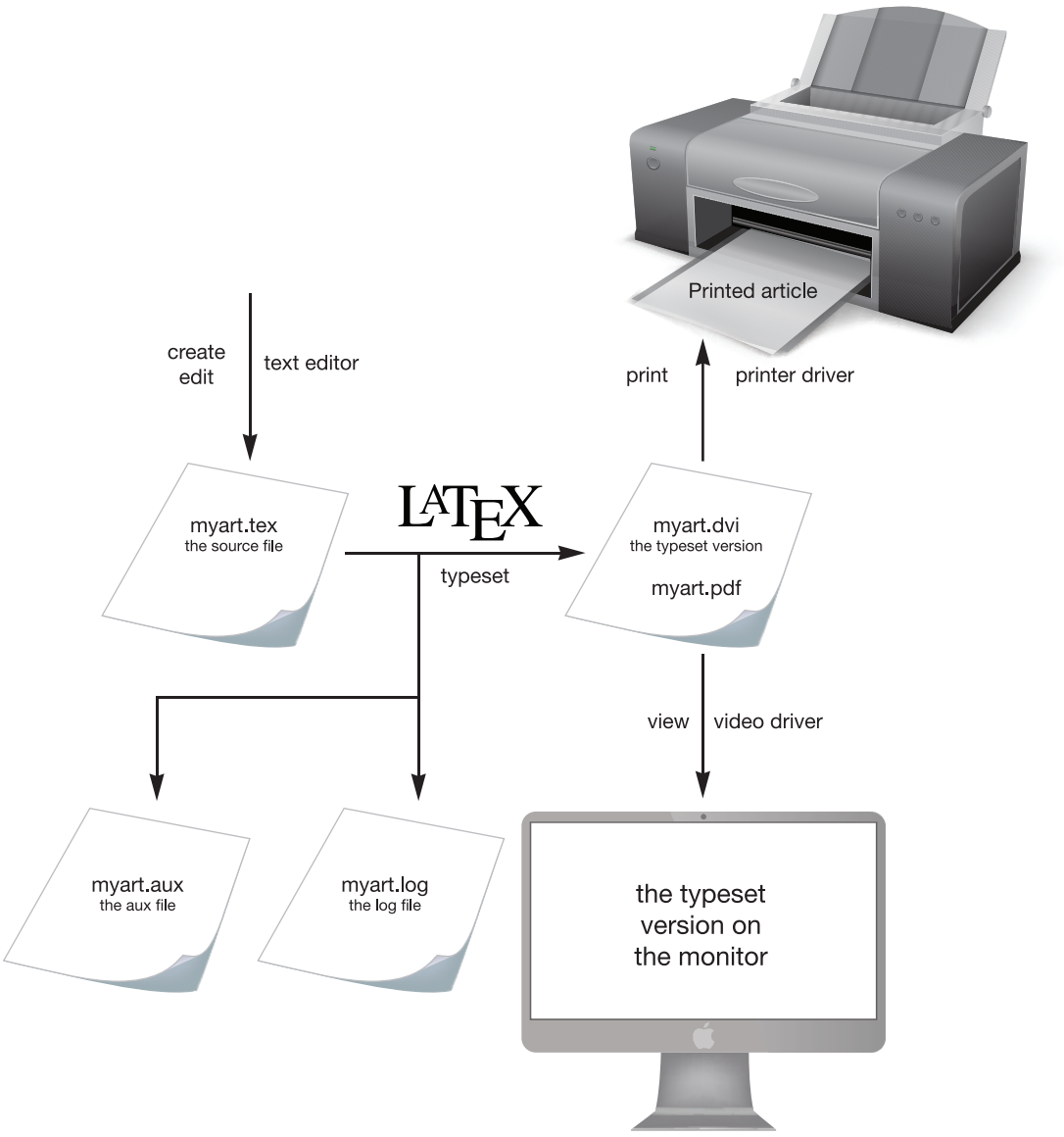


Figure 2.2: Using LaTeX.

1. `myart.pdf` The typeset article in PDF format.
2. `myart.aux` The auxiliary file, used by LaTeX for internal bookkeeping, including cross-references and bibliographic citations.
3. `myart.log` The `log` file. LaTeX records the typesetting session in the `log` file, including any warnings and messages that appear on your monitor in the `log` window.

Your computer uses a *video driver* to display the typeset article on your monitor and a *printer driver* to print the typeset article on a printer. The video and printer drivers are computer and LaTeX implementation dependent.

It should be emphasized that of the three applications used, only one is the same for all computers and all implementations.

LaTeX always uses the `aux` file from the last typesetting. Here is an example. Your article has Theorems 1 (with `\label{T:first}`) and 2 (with `\label{T:main}`). The `aux` file has the two lines:

```
\newlabel{T:first}{{1}{1}}
\newlabel{T:main}{{2}{1}}
```

`\newlabel{T:first}{{1}{1}}` means that the label `T:first` is assigned the value 1 and appears on page 1. `\newlabel{T:main}{{2}{1}}` means that the label `T:main` is assigned the value 2 and appears on page 1. So the reference

```
see Theorems \ref{T:first} and \ref{T:first}.
```

is typeset as

see Theorems 1 and 2.

Now add a new theorem between Theorems 1 and 2. Typeset the article. In the typeset article, the three theorems are properly numbered, but it still contains the same typeset line:

see Theorems 1 and 2.

The `aux` file has the lines:

```
\newlabel{T:first}{{1}{1}}
\newlabel{T:main}{{3}{1}}
```

So at the next typesetting, the reference is displayed as

see Theorems 1 and 3.

2.3 Logical and visual design

The typeset version of `firstarticle.tex` looks impressive on p. 4. To produce such articles, you need to understand that there are two aspects of article design: *visual* and *logical*.

As an example, let us look at a theorem from `firstarticle.tex` (see the typeset form of the theorem on page 4). You tell LaTeX that you want to state a theorem by using a `theorem` environment:

```
\begin{theorem}\label{T:technical}
Let $L$ be a finite lattice.

...
\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 9.5, 10.1.2, and 17.1). Section 9.1 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.

For more on this topic, see Section C.4.

2.4 General error messages

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

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

1. Typographical errors, which LaTeX blindly typesets.
2. Errors in 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 source file, finding the errors, and then editing the typeset file. Mistakes in the second and

third categories may trigger errors during the typesetting process, such as lines too wide of Section 1.4.

We now look at some examples of the third class of errors by deliberately introducing a number of mistakes into `firstarticle.tex` and examining the messages.

Experiment 1. In `firstarticle.tex`, go to line 19 (use the `Go to Line` command of your editor) and remove the closing brace so that it reads `\begin{abstract`

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

```
{abstract We present a technical result for congruences on\ETC.
./firstarticle.tex:23:
Paragraph ended before \begin was complete.
<to be read again>
                   \par
l.23
```

Line 23 of the file is the line after `\maketitle`. The message informs you that the environment name was not completed.

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

Experiment 2. Now restore line 19, then go to line 21 and change `\end{abstract}` to `\end{abstrac}` and typeset again. LaTeX informs you of another error:

```
./firstarticle.tex:21: LaTeX Error: \begin{abstract}
on input line 19 ended by \end{abstrac}.


See the LaTeX manual or LaTeX Companion for explanation.
Type  H <return>  for immediate help.
 ...
l.21 \end{abstrac}
```

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

Experiment 3. Correct the error in line 21, and introduce a new error in line 61. This line reads

```
z_1 \equiv y+ w \pmod{\delta}.
```

Change `\delta` to `\deta`. Now, when you typeset the document, LaTeX reports

```
./firstarticle.tex:61: Undefined control sequence.
<argument> {\operator@font mod}\mkern 6mu\deta


l.61 z_1 \equiv y+ w \pmod{\deta}
```

This mistake is easy to identify: `\deta` is a misspelling of `\delta`.

Experiment 4. In line 38, delete the closing brace of the `\label` command. This results in a message:

```
Runaway definition?
->E:cover\text {If $x$ is covered by $y,z \in L$ and\ETC.
! File ended while scanning definition of \df@label.
<inserted text>
                }
<*> firstarticle.tex
```

Undo the change to line 38.

Experiment 5. Add a blank line following line 61:

```
    x+ z = z + z_1 \equiv z + (y+ w) = y+ z \pmod{\delta},
```

This change results in the message

```
./firstarticle.tex:62: Missing $ inserted.
<inserted text>
                $
l.62
```

There can be no blank lines within a displayed math environment. LaTeX catches the mistake, but the message itself is misleading.

Experiment 6. Add a `$` before `\pmod` in line 61 (such an error often occurs when cutting and pasting a formula). You get the message:

```
./firstarticle.tex:61: Display math should end with $$.
<to be read again>
                   \penalty
l.61 z_1 \equiv y+ w $\pmod{\delta}
```

Maybe this could be more to the point?

Tip LaTeX's messages are not very useful with displayed formulas. Comment out some of the lines to try to localize the problem.

Tip Typeset often.

Typesetting my book *First Steps into* LaTeX with the closing brace of the first `\caption` command on line 480 of the source file missing, I 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 changes and then typeset, you may not know where to start.

2.5 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 errors in formulas, we introduce mistakes into `formulanote.tex`.

Experiment 1 In line 6 of `formulanote.tex`, delete the third `$` symbol; save the file under the name `formulanotebad1.tex` in the `work` folder.

Typeset `formulanotebad1.tex`. LaTeX generates the following message:

```
! Missing $ inserted.
<inserted text>
                $
l.6 as $(u, v)$ and  (u, \infty
                                )$. Such an interval
```

LaTeX reads `(u, \infty)` as text; but the `\infty` command instructs LaTeX to typeset a math symbol, which can only be done in a 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 formula *should* start just before `(u`.

Experiment 2 In line 16 of `formulanote.tex`, delete the second `}` symbol and save it under the name `formulanotebad2.tex` in the `work` folder. This introduces an error: the closing brace of the subscript (see page 15) is missing. Now typeset the note. You get the message

```
Missing } inserted.
<inserted text>
                }
l.12 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 started with `{`, but it reached the end of the 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 In `mathnote.tex`, delete the two $ signs in line 19, 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 message by LaTeX and the typeset version looks good. Notice the difference in the shape of the letter a in the two cases. You need sharp eyes to catch such an error.

Tip After an error is corrected, LaTeX can refuse to typeset your document. If your document is `document.tex`, look in the same folder for the *auxiliary file* `document.aux` that was created by LaTeX. Delete `document.aux` and **typeset twice**. See Section 2.2.

2.6 Your errors: Davey's Dos and Don'ts

Based on his many years of experience correcting LaTeX articles for the journal *Algebra Universalis*, Brian Davey collected the LaTeX mistakes most often made by authors. Here are some items from his list, divided into three categories.

Commands

1. Place ALL custom commands and environments in the preamble!
   If you have trouble with custom commands, then you know where to find them.

2. Don't use `\def`; rather use `\newcommand` or `\renewcommand`.
   `\def` is a TeX command. It is like `\newcommand` (see Sections 1.9 and 14.1), but it can redefine an existing command. Redefining your own commands is bad enough, redefining a TeX command can be a disaster.

3. Do not simply type the name of an operator into a formula. Declare the appropriate operator; see Sections 1.9 and 14.1.
   For instance, do not type `$length I$`; it typesets as $lengthI$. It should be length I, typed as `$\length I$`. Of course, you have to add

   ```
   \DeclareMathOperator{\length}{length}
   ```

   to the preamble (see Section 1.8).

4. When you send a document to a coauthor or submit an article to a journal, remove all the custom commands not used.
   This is a real time saver for your coauthor and editor.

Text

1. Do not produce a list with horizontal and vertical spacing commands. Use a list environment; see Sections 3.8 and 4.2.

2. Do not type numbers for citations and internal references. Use `\cite{...}` for citations and `\ref{...}` for references. For references to equations, use `\eqref`; see Sections 1.7.1 and 5.3.

3. Do not number proclamations (see Section 4.4). Use the standard `amsart` environments for theorems, and so on, and let LaTeX number them.

4. When writing a document for a journal requiring a document class file, **do not**
   (a) change any of the size parameters: for instance, do not use options like `12pt` to change the font size or the `\setlength` command to change any parameter of the page size;

   (b) insert vertical white space via `\bigskip`, `\smallskip`, `\vskip`, `\vspace`, etc, nor via your own custom commands. Do not adjust horizontal space without a very good reason.
   So if you want to display some text:

> Please, display this text.

don't do this:

```
\medskip
\hspace*{6pt} Please, display this text.
\medskip
```

but rather

```
\begin{itemize}
\item[] Please, display this text.
\end{itemize}
```

or

```
\begin{quote}
Please, display this text.
\end{quote}
```

5. Do not leave a blank line before `\end{proof}` or before a text environment (see Section 4.1).

6. Do not use the `geometry` package.

Formulas

1. Do not leave a blank line before a displayed formula.

2. Don't use the symbol `|` in a set description, use the binary relation `\mid`; see Section 5.5.4.
   For instance, `\{ x | x^2 < 2 \}` typesets as $\{x|x^2 < 2\}$. The correct form is $\{x \mid x^2 < 2\}$, typed as `$\{x \mid x^2 < 2\}$`.

3. Don't put punctuation marks inside an inline math environment.
   For instance, $\sin x.$ typed as `$\sin x.$`; use `$\sin x$`. This typesets as $\sin x$. Notice the smaller space between "$\sin x$." and "typed" and the wider space between "`$\sin x$.`" and "This"; see Sections 1.3 and 3.2.2.

4. Don't use two or more displayed formulas one after another. Use an appropriate environment such as `\align`, `\alignat`, `\gather`, and so on (see Section 7.1.1).

5. Don't use `\left\{`, `\right\}`, `\left(`, `\right)`, and so on, by default (see page 15 and Section 5.5.1 for the commands `\left` and `\right`). Even when `\left` and `\right` do not change the size of the symbol, they add extra space after the closing delimiter.

6. Use `\colon` for functions. For instance, `$f(x) \colon x \to x^2$` typesets as $f(x) \colon x \to x^2$. If you type `$f(x) : x \to x^2$`, you get $f(x) : x \to x^2$; the spacing is bad.

7. Use `\[` and `\]` (or `equation*`) to type a displayed math environment (see Section 1.7) rather than the old TeX `$$` matched by `$$`. While display math produced via the latter does work properly most of the time, there are some LaTeX commands that do not; for example, `\qedhere`.

8. Do not use the `center` environment to display formulas.

9. Use `\dots` first and let LaTeX make the decision whether to use `\dots` or `\cdots`; see page 16 and Section 5.4.3. If LaTeX gets it wrong, then use `\cdots` or `\ldots`.

10. If you can, avoid constructs (for instance, $\overset{\text{up}}{\to}$) in inline formulas that disrupt the regular line spacing. Although LaTeX automatically leaves room for it, it does not look good, as a rule.

http://www.springer.com/978-3-319-23795-4

More Math Into LaTeX
Grätzer, G.
2016, XXX, 609 p. 76 illus., 23 illus. in color., Softcover
ISBN: 978-3-319-23795-4