The UK \TeX Users’ Group FAQ
Your 136 Questions Answered

GELLMU Edition

Maintained for the UK \TeX Users Group Committee
by Robin Fairbairns

version 2.4.7a, date 2000/07/07

Note

This document is an updated and extended version of the FAQ article that was pub-
lished as the December 1994 and 1995, and March 1999 editions of the UKTUG magazine
Baskerville (which weren’t formatted like this). The article is also available via the World
Wide Web.

Table of Contents

Part A Introduction .................................................. 6
Part B The Background ................................................. 7
    1 What is \TeX? .................................................. 7
    2 How should I pronounce “\TeX”? ............................. 7
    3 What is \METAFONT? .......................................... 7
    4 What is MetaPost? ........................................... 8
    5 How can I be sure it’s really \TeX? .......................... 8
    6 Are \TeX and friends Y2K compliant? ....................... 9
    7 What is \LaTeX? ................................................ 9
    8 How should I pronounce “\LaTeX(2\epsilon)”? ............ 10
    9 Should I use plain \TeX or \LaTeX? .......................... 10
   10 What is \CONTEXT? ........................................... 10
   11 What are the AMS packages (\AMS-\TeX, etc.)? ........... 10
   12 What is Eplain? ............................................. 11
   13 What is Lollipop? ............................................ 11
   14 What is Texinfo? ............................................. 11
   15 If \TeX is so good, how come it’s free? ................... 12
<table>
<thead>
<tr>
<th>Page</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>What is the future of \TeX{}?</td>
</tr>
<tr>
<td>17</td>
<td>What are TUG and \texttt{TUGboat}?</td>
</tr>
<tr>
<td>18</td>
<td>Are there nationally-based user groups, too?</td>
</tr>
<tr>
<td>19</td>
<td>TUG Technical Working Groups</td>
</tr>
<tr>
<td></td>
<td>Part C Documentation and Help</td>
</tr>
<tr>
<td>20</td>
<td>Books on \TeX{} and its relations</td>
</tr>
<tr>
<td>21</td>
<td>Books on Type</td>
</tr>
<tr>
<td>22</td>
<td>Where to find this article</td>
</tr>
<tr>
<td>23</td>
<td>Where to get help</td>
</tr>
<tr>
<td>24</td>
<td>(B)\TeX{} Tutorials</td>
</tr>
<tr>
<td>25</td>
<td>\texttt{METAFONT} and MetaPost Tutorials</td>
</tr>
<tr>
<td>26</td>
<td>\texttt{BIB\TeX} Documentation</td>
</tr>
<tr>
<td>27</td>
<td>The \texttt{PT\TeX} manual</td>
</tr>
<tr>
<td>28</td>
<td>Finding (I)\TeX{} macro packages</td>
</tr>
<tr>
<td>29</td>
<td>Finding files in the CTAN archives</td>
</tr>
<tr>
<td></td>
<td>Part D Bits and pieces of \TeX{}</td>
</tr>
<tr>
<td>30</td>
<td>What is a DVI file?</td>
</tr>
<tr>
<td>31</td>
<td>What is a driver?</td>
</tr>
<tr>
<td>32</td>
<td>What are PK files?</td>
</tr>
<tr>
<td>33</td>
<td>What are TFM files?</td>
</tr>
<tr>
<td>34</td>
<td>Virtual fonts</td>
</tr>
<tr>
<td>35</td>
<td>\texttt{\special}s commands</td>
</tr>
<tr>
<td>36</td>
<td>Documented \texttt{I\TeX} sources (.dtx files)</td>
</tr>
<tr>
<td>37</td>
<td>What are the EC fonts?</td>
</tr>
<tr>
<td>38</td>
<td>What is TDS?</td>
</tr>
<tr>
<td></td>
<td>Part E Acquiring the Software</td>
</tr>
<tr>
<td>39</td>
<td>Repositories of \TeX{} material</td>
</tr>
<tr>
<td>40</td>
<td>What's the CTAN nonfree tree?</td>
</tr>
<tr>
<td>41</td>
<td>Contributing a file to the archives</td>
</tr>
<tr>
<td>42</td>
<td>Finding new fonts</td>
</tr>
<tr>
<td>43</td>
<td>\TeX{} CD-ROMs</td>
</tr>
<tr>
<td></td>
<td>Part F \TeX{} Systems</td>
</tr>
<tr>
<td>44</td>
<td>(B)\TeX{} for different machines</td>
</tr>
<tr>
<td>45</td>
<td>\TeX{}-friendly editors and shells</td>
</tr>
<tr>
<td>46</td>
<td>Commercial \TeX{} implementations</td>
</tr>
<tr>
<td></td>
<td>Part G DVI Drivers and Previewers</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DVI to PostScript conversion programs ........................................ 33
DVI drivers for HP LaserJet ..................................................... 33
DVI previewers .................................................................. 33

Part H  Support Packages for TeX .................................................. 34
Fig, a TeX-friendly drawing package ........................................... 34
TeXCAD, a drawing package for LaTeX ....................................... 34
Spelling checkers for work with TeX ........................................... 34

Part I  Literate programming ....................................................... 35
What is Literate Programming? .................................................. 35
WEB for C, FORTRAN, and other languages ............................... 35

Part J  Format conversions ........................................................ 35
Conversion between \LaTeX and others ......................................... 35
Conversion from \LaTeX to plain ASCII ....................................... 36
Conversion from SGML or HTML to TeX .................................... 37
\LaTeX conversion to HTML .................................................... 38
Making hypertext documents from TeX ...................................... 38
Making Acrobat documents from \LaTeX ...................................... 39
Using TeX to read SGML or XML directly ................................... 40

Part K  METAFONT ................................................................. 40
Getting METAFONT to do what you want ................................. 40
Which font files should be kept ............................................... 42
Getting bitmaps from the archives .......................................... 42

Part L  PostScript and TeX ........................................................ 43
Using PostScript fonts with TeX ............................................... 43
Previewing files using PostScript fonts ...................................... 43
TeX font metric files for PostScript fonts .................................... 44
Problems using PostScript fonts .............................................. 44
Choice of scalable outline fonts .............................................. 45
Including a PostScript figure in \LaTeX ..................................... 47

Part M  Special sorts of typesetting .............................................. 47
Drawing with TeX ................................................................. 47
Double-spaced documents in \LaTeX ......................................... 48
Formatting a thesis in \LaTeX .................................................... 48
Flowing text around figures in \LaTeX ....................................... 49
Alternative head- and footlines in \LaTeX ................................. 50
Part Q Why does it do that? ........................................ 69
121 What’s going on in my \include commands?................. 69
122 Why does it ignore paragraph parameters?.................... 69
123 What’s the reason for ‘protection’? .......................... 70
124 Why doesn’t \verb work within...? .......................... 71
125 Case-changing oddities ........................................ 71
126 Why are # signs doubled in macros? ........................ 72
127 Why does L\TeX split footnotes across pages? ............... 72
128 Getting \marginpar on the right side ........................ 73
129 Why does \pictex run out of ‘\dimen’s’? ....................... 73
130 Why won’t L\TeX ‘fixed’ names stay changed? ............... 73

Part R Current \TeX Projects ........................................... 74
131 L\TeX 2ε (the ‘new’ standard L\TeX) ........................ 74
132 The \crtex project ............................................. 75
133 The Omega project ............................................. 75
134 The \n \s project ............................................. 75
135 The \pictex project ............................................ 76

Part S Perhaps There isn’t an Answer .............................. 76
136 What to do if you find a bug ................................. 76
Part A Introduction

This article was prepared by the Committee of the UK \TeX\ Users Group (UK TUG)\(^1\) as a development of a regular posting to the Usenet newsgroup \texttt{comp.text.tex} that was maintained for some time by Bobby Bodenheimer (\texttt{bobby@hot.caltech.edu}).

Usenet is a mechanism for exchanging articles between people who share interests or needs\(^2\) a newsgroup is an area within Usenet carrying a particular class of articles. Since a common sort of article asks for help, advice or information, and since certain of these questions are regularly repeated (often with monotonous regularity), some public-spirited souls took to writing articles which listed “Frequently Asked Questions” and answers to them. Many members of UK TUG do not have access to Usenet, but could be expected to value the answers about \TeX\ that have accumulated over the years; so we decided to update the list and publish it in \textit{Baskerville}; we are grateful to Bobby for his permission to use his article in this way. As a \textit{quid pro quo}, we are making the source of the article freely available \texttt{(usergrps/uktug-faq)}, and it can be compiled by anyone who runs reasonably current \TeX\ (question 131), and has the required fonts. It was the committee’s original intention that it would also be possible for the content of this article to feed back to the world-wide \TeX\ community via regular posting to Usenet, but since Bobby was forced to abandon his work in the area, no-one has been able to take his place.

Therefore, a translation of the article has been made available on the World-Wide Web, via URL \url{http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes}

A hypertext version is also to be found on the \TeX\ Live CD-ROM (question 43).

When we started, we rearranged Bobby’s original, and we have since added new questions and answers on the basis of our experience of answering questions about \TeX, writing documents in \TeX, and developing macros for \TeX, over the years.

The committee is grateful for help and advice, from the following outside its number: Donald Arsenneau, Barbara Beeton, Karl Berry, Damian Cugley, Michael Downes, John Hobby, Berthold Horn, Werner Icking, David Kastrup, Ted Nieland, Pat Rau, Peter R. Wilson, Piet van Oostrum, Oren Patashnik, Joachim Schrod, Ulrik Vieth, Rick Zaccone and Reinhard Zierke.

Further, Rosemary Bailey and Chris Rowley (who resigned from the committee in 1995), Alan Jeffrey and Carol Hewlett (who resigned from the committee in 1996), David Carlisle (who resigned from the committee in 1997), and Robin Fairbairns, Jonathan Fine and Sebastian Rahtz (who resigned from the committee in 1998) all made significant contributions to the conception, development and subsequent revision of this FAQ while they remained on the committee, and we are grateful to them for their contributions to it. In particular, Alan Jeffrey developed the technique whereby the Web version of the FAQ is generated.

And lastly, the editor has regularly used the work of other groups for ‘inspiration’, notably the excellent (German-language) FAQ published by Dante e.V. (see \url{http://www.dante.de/faq/de-tex-faq/}).

Finding the Files

\(^1\)For 1998–99: Peter Abbott, Kaveh Bazargan, Malcolm Clark, Roy Everett, James Foster, David Hardy, Hong Ji, Phil Molyneux, John Palmer, Kim Roberts, Philip Taylor and Dominik Wajastyk.

\(^2\)Usenet, as its name implies, is a means of using some sort of network; in the earliest days the network was made by stringing together a series of telephone lines, but nowadays Usenet is most often carried over the Internet.
Unless otherwise specified, all files mentioned in this article are available from a CTAN archive, or from one of their mirrors. Question 39 gives details of the CTAN archives, and how to retrieve files from them. If you don’t have access to the Internet, question 43 tells you of sources of CD-ROMs that offer snapshots of the archives.

The reader should also note that the first directory name of the path name of every file on CTAN has been elided from what follows, for the simple reason that it’s always the same (`tex-archive`).

To avoid confusion, we’ve also elided the full stop from the end of any sentence whose last item is a path name (note that such sentences only occur at the end of paragraphs). Though the path names are set in a different font from running text, it’s not easy to distinguish the font of a single dot!

Part B  The Background

1  What is T\textsc{e}X?

T\textsc{e}X is a typesetting system written by Donald E. Knuth, who says in the Preface to his book on T\textsc{e}X (see question 20) that it is “intended for the creation of beautiful books —and especially for books that contain a lot of mathematics”.

Knuth developed a system of ‘literate programming’ to write T\textsc{e}X, and he provides the literate (WEB) source of T\textsc{e}X free of charge, together with tools for processing the web source into something that can be compiled and something that can be printed; there’s never any mystery about what T\textsc{e}X does. Furthermore, the WEB system provides mechanisms to port T\textsc{e}X to new operating systems and computers; in order that one may have some confidence in the ports, Knuth supplied a test by means of which one may judge the fidelity of a T\textsc{e}X system. T\textsc{e}X and its documents are therefore highly portable.

T\textsc{e}X is a macro processor, and offers its users a powerful programming capability. For this reason, T\textsc{e}X on its own is a pretty difficult beast to deal with, so Knuth provided a package of macros for use with T\textsc{e}X called plain T\textsc{e}X plain T\textsc{e}X is effectively the minimum set of macros one can usefully employ with T\textsc{e}X, together with some demonstration versions of higher-level commands (the latter are better regarded as models than used as-is). When people say they’re “programming in T\textsc{e}X”, they usually mean they’re programming in plain T\textsc{e}X.

2  How should I pronounce “T\textsc{e}X”?

The ‘X’ stands for the Greek letter Chi (\(\chi\)), and is pronounced by English-speakers either a bit like the ‘ch’ in ‘loch’ ([x] in the IPA) or like ‘k’. It definitely is not pronounced ‘ks’.

3  What is METAFONT?

METAFONT was written by Knuth as a companion to T\textsc{e}X whereas T\textsc{e}X defines the layout of glyphs on a page, METAFONT defines the shapes of the glyphs and the relations between

---

3‘Full stop’ (British English) == ‘period’ (American English)
them. METAFONT details the sizes of glyphs, for \TeX’s benefit, and details the rasters used to represent the glyphs, for the benefit of programs that will produce printed output as post processes after a run of \TeX.

METAFONT’s language for defining fonts permits the expression of several classes of things: first (of course), the simple geometry of the glyphs; second, the properties of the print engine for which the output is intended; and third, ‘meta’-information which can distinguish different design sizes of the same font, or the difference between two fonts that belong to the same (or related) families.

Knuth (and others) have designed a fair range of fonts using METAFONT, but font design using METAFONT is much more of a minority skill than is \TeX macro-writing. The complete \TeX-user nevertheless needs to be aware of METAFONT, and to be able to run METAFONT to generate personal copies of new fonts.

4 What is MetaPost?

The MetaPost system (by John Hobby) implements a picture-drawing language very much like that of METAFONT except that it outputs Encapsulated PostScript files instead of run-length-encoded bitmaps. MetaPost is a powerful language for producing figures for documents to be printed on PostScript printers, either directly or embedded in (L)\TeX documents. It includes facilities for directly integrating \TeX text and mathematics with the graphics. The PostScript output is of such a simple form that MetaPost output files can be directly included in PDF/L\TeX documents (question 135). (Knuth tells us that he uses nothing else but MetaPost for diagrams in text that he is writing.)

Much of MetaPost’s source code was copied from METAFONT’s sources with Knuth’s permission.

5 How can I be sure it’s really \TeX?

\TeX (and METAFONT and MetaPost) are written in a ‘literate’ programming language called Web (question 53) which is designed to be portable across a wide range of computer systems. How, then, is a new version of \TeX checked?

Of course, any sensible software implementor will have his own suite of tests to check that his software runs: those who port \TeX and its friends to other platforms do indeed perform such tests.

Knuth, however, provides a ‘conformance test’ for both \TeX (trip) and METAFONT (trap). He characterises these as ‘torture tests’: they are designed not to check the obvious things that ordinary typeset documents, or font designs, will exercise, but rather to explore small alleyways off the main path through the code of \TeX. They are, to the casual reader, pretty incomprehensible!

Once an implementation of \TeX has passed its \texttt{trip}, or and implementation of METAFONT has passed its \texttt{trap}, test it may reasonably be distributed as a working version.

\footnote{PDF/L\TeX cannot normally handle PostScript inclusions}
6  Are \TeX{} and friends Y2K compliant?

**Crashing:** None of \TeX{}, METAFONT or MetaPost can themselves crash due to any change whatever in the date of any sort.

**Timestamps:** As Knuth delivers the sources, a 2-digit year is stored as the creation time for format files and that value is printed in logfiles. These items should not be of general concern, since the only use of the date format file is to produce the log output, and the log file is designed for human readers only.

Knuth’s distributed source does not designate the code, which generates this 2-digit date, as a valid area where implementations may differ. However, he announced in 1998 that implementors can alter this code without fear of being accused of non-compliance. Nearly all implementations that are being actively maintained had been modified to generate 4-digit years in the format file and the log, by the end of 1998.

**The \texttt{\texttt{\texttt{year}}} primitive:** Certification of a \TeX{} implementation (see question 5) does not require that \texttt{\texttt{\texttt{\texttt{year}}}} return a meaningful value (which means that \TeX{} can, in principle, be implemented on platforms that don’t make the value of the clock available to user programs). The \TeX{}\texttt{}\textit{book} (see question 20) defines \texttt{\texttt{\texttt{year}}} as “the current year of our Lord”, which is the only correct meaning for \texttt{\texttt{\texttt{year}}} for those implementations which can supply a meaningful value, which is to say nearly all of them.

In short, \TeX{} implementations should provide a value in \texttt{\texttt{\texttt{year}}} giving the 4-digit year Anno Domini, or the value 1776 if the platform does not support a date function.

Note that if the system itself fails to deliver a correct date to \TeX{}, then \texttt{\texttt{\texttt{year}}} will of course return an incorrect value. \TeX{} cannot be considered Y2K compliant, in this sense, on a system that is not itself Y2K compliant.

**Macros:** \TeX{} macros can in principle perform calculations on the basis of the value of \texttt{\texttt{\texttt{year}}}.

The \L A\TeX{} suite (question 7) performs such calculations in a small number of places; the calculations performed in the current (supported) version of \L A\TeX{} are known to be Y2K compliant.

Other macros and macro packages should be individually checked.

**External software:** Software such as DVI translators needs to be individually checked.

7  What is \L A\TeX{}?

\L A\TeX{} is a \TeX{} macro package, originally written by Leslie Lamport, that provides a document processing system. \L A\TeX{} allows markup to describe the structure of a document, so that the user need not think about presentation. By using document classes and add-on packages, the same document can be produced in a variety of different layouts.

Lamport says that \L A\TeX{} “represents a balance between functionality and ease of use”. This shows itself as a continual conflict that leads to the need for such as the present article: \L A\TeX{} \textit{can} meet most user requirements, but finding out \textit{how} is often tricky.
8 How should I pronounce “\LaTeX(2\epsilon)”? 

Lamport never recommended how one should pronounce \LaTeX, but a lot of people pronounce it ‘Lay \TeX’ or perhaps ‘Lah \TeX’ (with \TeX pronounced as the program itself; see question 2). The ‘epsilon’ in ‘\LaTeX 2\epsilon’ is supposed to be suggestive of a small improvement over the old \LaTeX 2.09. Nevertheless, most people pronounce the name as ‘\LaTeX-two-ee’.

9 Should I use plain \TeX or \LaTeX? 

There’s no straightforward answer to this question. Many people swear by plain \TeX, and produce highly respectable documents using it (Knuth is an example of this, of course). But equally, many people are happy to let someone else take the design decisions for them, accepting a small loss of flexibility in exchange for a saving of brain power.

The arguments around this topic can provoke huge amounts of noise and heat, without offering much by way of light; your best bet is to find out what those around you are using, and to go with the crowd. Later on, you can always switch your allegiance; don’t bother about it.

If you are preparing a manuscript for a publisher or journal, ask them what markup they want before you develop your own; many big publishers have developed their own \LaTeX styles for journals and books, and insist that authors stick closely to their markup.

10 What is CONT\TeX? 

CONT\TeX is a macro package developed by Hans Hagen, originally to serve the needs of the Dutch firm, Pragma. It was designed with the same general-purpose aims as \LaTeX, but (being younger) reflects much more recent thinking about the structure of markup, etc. In particular, CONT\TeX can customise its markup to an author’s language (customising modules for Dutch, English and German are provided, at present).

CONT\TeX is well integrated, in all of its structure, with the needs of hypertext markup, and in particular with the facilities offered by PDF\TeX.

CONT\TeX doesn’t yet have quite such a strong developer community as does \LaTeX, but those developers who are active seem to have prodigious energy.

Try a copy, from macros/context

11 What are the AMS packages (\AMS-\TeX, etc.)? 

\AMS-\TeX is a \TeX macro package, originally written by Michael Spivak for the American Mathematical Society (AMS) during 1983–1985. It is described in “The Joy of \TeX” by Michael D. Spivak (second edition, AMS, 1990, ISBN 0-821-82997-1). It is based on plain \TeX, and provides many features for producing more professional-looking maths formulas with less burden on authors. It pays attention to the finer details of sizing and positioning that mathematical publishers care about. The aspects covered include multi-line displayed equations, equation numbering, ellipsis dots, matrices, double accents, multi-line subscripts, syntax checking (faster processing on initial error-checking \TeX runs), and other things.
As \LaTeX increased in popularity, authors asked to submit papers to the AMS in \LaTeX, and so the AMS developed \AMS-\LaTeX, which is a collection of \LaTeX packages and classes that offer authors most of the functionality of \AMS-\TeX.

12 What is Eplain?

The Eplain macro package expands on and extends the definitions in plain \TeX. Eplain is not intended to provide “generic typesetting capabilities”, as do \LaTeX or Texinfo (question 14). Instead, it provides definitions that are intended to be useful regardless of the high-level commands that you use when you actually prepare your manuscript.

For example, Eplain does not have a command \textbackslash section, which would format section headings in an “appropriate” way, as \LaTeX’s \textbackslash section. The philosophy of Eplain is that some people will always need or want to go beyond the macro designer’s idea of “appropriate”. Such canned macros are fine — as long as you are willing to accept the resulting output. If you don’t like the results, or if you are trying to match a different format, you are out of luck.

On the other hand, almost everyone would like capabilities such as cross-referencing by labels, so that you don’t have to put actual page numbers in the manuscript. Karl Berry, the author of Eplain, says he is not aware of any generally available macro packages that do not force their typographic style on an author, and yet provide such capabilities.

13 What is Lollipop?

Lollipop is a macro package written by Victor Eijkhout; it was used in the production of his book “\TeX by Topic” (see question 20). The manual says of it:

Lollipop is ‘\TeX made easy’. Lollipop is a macro package that functions as a toolbox for writing \TeX macros. It was my intention to make macro writing so easy that implementing a fully new layout in \TeX would become a matter of less than an hour for an average document, and that it would be a task that could be accomplished by someone with only a very basic training in \TeX programming.

Lollipop is an attempt to make structured text formatting available for environments where previously only WYSIWYG packages could be used because adapting the layout is so much more easy with them than with traditional \TeX macro packages.

The manual goes on to talk of ambitions to “capture some of the \LaTeX market share”; it’s a very witty package, but little sign of it taking over from \LaTeX is detectable… An article about Lollipop appeared in \textit{TUGboat} 13(3).

14 What is Texinfo?

Texinfo is a documentation system that uses one source file to produce both on-line information and printed output. So instead of writing two different documents, one for the on-line help and the other for a typeset manual, you need write only one document source file. When the work is revised, you need only revise one document. You can read the on-line information, known as an “Info file”, with an Info documentation-reading program. By convention, Texinfo source
file names end with a .texi or .texinfo extension. You can write and format Texinfo files into Info files within GNU emacs, and read them using the emacs Info reader. If you do not have emacs, you can format Texinfo files into Info files using makeinfo and read them using info.

The Texinfo distribution, including a set of \TeX\ macros for formatting Texinfo files is available as macros/texinfo-dist

15 If \TeX is so good, how come it’s free?

It’s free because Knuth chose to make it so. He is nevertheless apparently happy that others should earn money by selling \TeX-based services and products. While several valuable \TeX-related tools and packages are offered subject to restrictions imposed by the GNU General Public Licence (‘Copyleft’), \TeX itself is not subject to Copyleft.

There are commercial versions of \TeX available; for some users, it’s reassuring to have paid support. What is more, some of the commercial implementations have features that are not available in free versions. (The reverse is also true: some free implementations have features not available commercially.)

Usually, this article does not describe commercial versions; Question 46 lists the major vendors.

16 What is the future of \TeX?

Knuth has declared that he will do no further development of \TeX he will continue to fix any bugs that are reported to him (though bugs are rare). This decision was made soon after \TeX version 3.0 was released; at each bug-fix release the version number acquires one more digit, so that it tends to the limit $\pi$ (at the time of writing, Knuth’s latest release is version 3.14159). Knuth wants \TeX to be frozen at version $\pi$ when he dies; thereafter, no further changes may be made to Knuth’s source. (A similar rule is applied to \METAfont; its version number tends to the limit $e$, and currently stands at 2.718.)

There are projects (some of them long-term projects: see, for example, question 132) to build substantial new macro packages based on \TeX. For the even longer term, there are various projects to build a successor to \TeX see questions 133 and 134.

17 What are TUG and \texttt{TUGboat}?

TUG is the \TeX Users Group. \texttt{TUGboat} is TUG’s main journal, containing useful articles about \TeX and \METAfont. TUG also produces a newsletter for members (\TeX and TUG News), organises a yearly conference, runs training courses, sells almost all \TeX-related books, and distributes \TeX-related microcomputer software on disk. TUG has a Technical Council to coordinate \TeX-related developments (question 19). Enquiries should be directed to:

\TeX Users Group
1466 NW Front Avenue, Suite 3141
Portland, OR 97209
USA
Tel: +1 503-223-9994
18  Are there nationally-based user groups, too?

The following groups publish their membership (etc.) information electronically on CTAN archives:

DANTE, Deutschsprachige Anwendervereinigung TeX e.V.
Postfach 10 18 40
D-69008 Heidelberg
Germany
Tel: +49 06221 2 97 66
Fax: +49 06221 16 79 06
Email: dante@dante.de
Web: http://www.dante.de/
CTAN details: usergrps/dante

Association GUTenberg,
BP 10,
93220 Gagny principal,
France
Email: secretariat@gutenberg.eu.org
Web: http://www.gutenberg.eu.org/
CTAN details: usergrps/gut

NTG
Postbus 394, 1740AJ Schagen,
The Netherlands
Email: ntg@nic.surfnet.nl
Web: http://www.ntg.nl/
CTAN details: usergrps/ntg

UK TeX Users' Group,
Membership enquiries: c/o Dr James Foster,
School of Mathematical Sciences
University of Sussex
Falmer, Brighton BN1 9QH
United Kingdom
Tel: +44 1273 678781
Fax: +44 1273 678097
Email: j.foster@sussex.ac.uk
General enquiries: uktug-enquiries@tex.ac.uk
Web: http://uk.tug.org/uk-tug

A listing of all known groups is available as usergrps-list
19 TUG Technical Working Groups

TUG (question 17) has an autonomous Technical Council which can appoint Technical Working Groups related to the specific areas of interest to the T\TeX{} community. Each group has a chair/contact person and establishes its own working methods and membership; anyone interested in taking part should contact the group’s chair person. Suggestions for new groups should be addressed to tech-council@tug.org; a list of members of the Technical Council can be found at http://tug.org/committees.html

Below is a brief list of currently active groups:

**WG-92-01 T\TeX{} Extended Mathematics Font Encoding.**
To create font encoding standards for Mathematical fonts used in T\TeX{} systems.
Contact: Barbara Beeton (bnb@math.ams.org)

**WG-92-05 T\TeX{} Archive Guidelines.**
To develop guidelines for the effective management and utilisation of major T\TeX{} archives, and to initiate communication among the maintainers of the existing archives for the purpose of coordination and synchronisation.
Contact: Sebastian Rahtz (s.rahtz@elsevier.co.uk)

**WG-94-07 T\TeX{} Directory Structure.**
To identify a universal directory structure for macros, fonts and other related T\TeX{} software so that recommendations can be made to all suppliers of T\TeX{} software.
The group’s current set of proposals are to be found on CTAN at tds/draft-standard/tds-0.9996.
Contact: Karl Berry (kb@cs.umb.edu)

**WG-94-08 DVI Driver Implementation and Standardisation Issues.**
To study the issues in the requirements of DVI Drivers imposed by changing needs and technologies, and to make recommendations for implementation and standardisation of such drivers to enhance the uniformity of their use. Work will include, but not be limited to, the examination of the use, syntax, and semantics of \special{..} commands.
Contact: Michael Sofka (sofkam@rpi.edu)

**WG-94-10 T\TeX{} and Linguistics.**
To study and discuss the requirements for typesetting linguistics in T\TeX{} and as a means of identifying, examining, testing, and comparing macros, fonts, style files and other aids for typesetting linguistics.
Contact: Christina Thiele (cthiele@ccs.carleton.ca)

Part C Documentation and Help

20 Books on T\TeX{} and its relations

While Knuth’s book is the definitive reference for T\TeX{}, there are other books covering T\TeX{}:


Introduction to \TeX{} by Norbert Schwarz (Addison-Wesley, 1989, ISBN 0-201-51141-X)

A Plain \TeX{} Primer by Malcolm Clark (Oxford University Press, 1993, ISBNs 0-198-53724-7 (hardback) and 0-198-53784-0 (paperback))

\TeX{} by Topic by Victor Eijkhout (Addison-Wesley, 1992, ISBN 0-201-56885-3)

\TeX{} for the Beginner by Wynter Snow (Addison-Wesley, 1992, ISBN 0-201-54799-6)


\TeX{}: Starting from Square One by Michael Doob (Springer Verlag, 1993, ISBN 3-540-56441-1)


A collection of Knuth’s publications about typography has recently been published:


For \LaTeX{}, see:


The \LaTeX{} Web Companion Integrating \TeX{}, HTML and XML by Michel Goossens and Sebastian Rahtz (Addison-Wesley, 1999, ISBN 0-201-43311-7)


Math into \LaTeX{}: An Introduction to \LaTeX{} and \AMS-L\TeX{} by George Grätzer (Birkhäuser, 1996, ISBN 0-817-63805-9)

First Steps in \LaTeX{} by George Grätzer (Birkhäuser, 1999, ISBN 0-8176-4132-7)
\LaTeX for Linux: A Vade Mecum by Bernice Sacks Lipkin (Springer-Verlag, 1999, ISBN 0-387-98708-8, second printing)
A list of errata for the first printing is available from:
A sample of George Grätzer’s book, in Adobe Acrobat format, is also available (info/mil/mil.pdf).
Example files for the \LaTeX Graphics and Web Companions are available in info/lgc (Graphics) and info/lwc (Web). Example files for George Grätzer’s ‘First Steps’ are available in info/FirstSteps.
The list for METAFONT is rather short:
Alan Hoenig’s ‘\TeX Unbound’ includes some discussion and examples of using METAFONT.
This list only covers books in English: UK TUG cannot hope to maintain a list of books in languages other than our own.

21 Books on Type

The following is a partial listing of books on typography in general. Of these, Bringhurst seems to be the one most often recommended.
There are many catalogues of type specimens but the following books provide a more interesting overall view of types in general and some of their history.
The Alphabet and Elements of Lettering by Frederick Goudy (Dover, 1963, ISBN 0-486-20792-7)
8)  

The above lists are limited to books published in English. Typographic styles are somewhat language-dependent, and similarly the ‘interesting’ fonts depend on the particular writing system involved.

22 Where to find this article

Bobby Bodenheimer’s article, from which the present one was developed, used to be posted (nominally monthly) to newsgroup comp.text.tex and cross-posted to newsgroups news.answers and comp.answers. The most recently posted copy of that article is kept on CTAN in directory obsolete/help; it is no longer kept in the news.answers archives.

A version of the present article may be browsed via the World-Wide Web, at URL

http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes;

the sources of the article are available from usergrps/uktug-faq

Both the Francophone \TeX\ usergroup Gutenberg and the Czech/Slovak usergroup CS-TUG have published translations of this FAQ, with extensions appropriate to their languages.

In addition, the German usergroup Dante posts a FAQ in German to de.comp.tex, which is archived as usergrps/dante-faq, and Marie-Paule Kluth posts a FAQ in French to fr.comp.text.tex, which is archived as help/LaTeX-FAQ-francaise

23 Where to get help

First . . . read any FAQ you can find. (Which is what you’re doing now, isn’t it?)

The tutorials and other on-line documentation can get you started but for more in-depth understanding you should get and read at least one of the many good books on the subject (question 20). The definitive source for (\LaTeX) is the source code itself, but that is not something to be approached lightly (if at all).

If you are seeking a particular package or program, look on your own system first: you might already have it.

If you have access to the internet, and in particular newsgroups, then (\LaTeX) discussions, including METAFONT and MetaPost, are on comp.text.tex. It is best to spend a little time familiarising yourself with the current threads before asking a question. The group is normally very responsive but asking a question that has just been answered is likely to dampen people’s enthusiasm to help you. In case you are not familiar with some of the abbreviations used in newsgroup postings, RTFM\(^5\) stands for ‘Read The . . . Manual’, and you don’t really want to get that response, do you?

\(^5\)The F stands for ‘Fill in the blank’
http://www.deja.com/usenet makes a good job of archiving Usenet news discussions, and comp.text.tex may be found there. If your internet access doesn’t include direct access to Usenet news, you can participate in discussions via Deja.

The few people who can’t use the World Wide Web, can’t access Usenet news, but can use electronic mail can seek help through mailing lists.

The TeXhax digest is operated as a mailing list. Send a message ‘subscribe texhax’ to texhax-request@tex.ac.uk to join it. Its turn-around is not rapid, but questions submitted to it do eventually get answered.

Many mailing lists exist that cover some small part of the TeX arena. A good source of pointers is http://www.tug.org/

Announcements of TeX-related installations on the CTAN archives are sent to the mailing list ctan-ann. Subscribe to the list by sending a message subscribe ctan-ann <your name>’ to listserv@urz.Uni-Heidelberg.de

Issues related to METAFONT (and, increasingly, MetaPost) are discussed on the metafont mailing list; subscribe by sending a message ‘subscribe metafont <your name>’ to listserv@ens.fr

Several other TeX-related lists may be accessed via listserv@urz.uni-heidelberg.de. Send a message containing the line ‘help’ to this address.

24 (La)TeX Tutorials

Some very fine tutorials have been written, over the years. Michael Doob’s splendid ‘Gentle Introduction’ to plain TeX has been stable for a very long time. See info/gentle/gentle.pdf

More dynamic is Tobias Oetiker’s ‘(Not so) Short Introduction to LaTeX 2ε’, which is regularly updated, as people suggest better ways of explaining things, etc. See info/lshort/english/lshort.pdf

A recent entrant is Harvey Greenberg’s ‘Simplified Introduction to LaTeX’; this was written for a lecture course, and is available (PostScript only, unfortunately) as info/simplified-latex/latex.ps

25 METAFONT and MetaPost Tutorials

Unfortunately there appear to be no tutorials on how to use either METAFONT or MetaPost, except for the information provided by their respective authors. There are, though, a couple of articles on how to run the programs.

Geoffrey Tobin has provided METAFONT for Beginners which is available as info/metafont-for-beginners.tex (question 62). This describes how the METAFONT system works and how to avoid some of the potential pitfalls.

Peter Wilson’s Some Experiences in Running METAFONT and MetaPost is available at info/metafp.ps (PostScript) or info/metafp.pdf (PDF format). On the METAFONT side the article is more geared towards testing and installing new METAFONT fonts than on the system, while on the other side it describes how to use MetaPost illustrations in LaTeX and PDFLaTeX documents, with an emphasis on how to use appropriate fonts for any text or mathematics.
26 Bib\TeX Documentation

\LaTeX, a program originally designed to produce bibliographies in conjunction with L\TeX, is explained in Section 4.3 and Appendix B of Leslie Lamport’s L\TeX manual (see question 20). The document “Bib\TeXing”, contained in the file \texttt{btxdoc.tex}, gives a more complete description. The \texttt{\LaTeX Companion} (see question 20) also has information on Bib\TeX and writing Bib\TeX style files.

The document “Designing Bib\TeX Styles”, contained in the file \texttt{btxhak.tex}, explains the postfix stack-based language used to write Bib\TeX styles (.bst files). The file \texttt{btxbst.doc} is the template for the four standard styles (plain, abbrv, alpha, unsrt). It also contains their documentation. The complete Bib\TeX documentation set (including the files above) is in \texttt{support/lsedit/bibtex-doc}.

There is a Unix Bib\TeX \texttt{man} page in the \texttt{web2c} package (see question 44). Any copy you may find of a \texttt{man} page written in 1985 (before “Bib\TeXing” and “Designing Bib\TeX Styles” appeared) is obsolete, and should be thrown away.

27 The PICTEX manual

PICTEX is a set of macros by Michael Wichura for drawing diagrams and pictures. The macros are freely available in \texttt{graphics/pictex}; however, the PICTEX manual itself is not free. Unfortunately, \texttt{TUG} is no longer able to supply copies of the manual (as it once did), and it is now available only through Personal \TeX Inc, the vendors of PCTEX (\texttt{http://www.pctex.com/}). The manual is not available electronically.

28 Finding (\LaTeX)\TeX macro packages

Before you ask for a \TeX macro or \LaTeX class or package file to do something, try searching Graham Williams’ catalogue, available as \texttt{help/Catalogue/catalogue.html}, or for efficient interactive searching via \texttt{http://www.tex.ac.uk/tex-archive/help/Catalogue/catalogue.html}; this lists many macro packages together with brief descriptive texts.

Having learnt of a file that seems interesting, search a CTAN archive for it (see question 29). For packages listed in \texttt{The \LaTeX Companion} (question 20), the file \texttt{info/companion.ctan} may be consulted as an alternative to searching the archive’s index. It lists the current location in the archive of such files.

29 Finding files in the CTAN archives

To find software at a CTAN site, you can use anonymous ftp to the host with the command ‘quote site index <term>’, or the searching script at \texttt{http://www.dante.de/cgi-bin/ctan-index}.

To get the best use out of the \texttt{ftp} facility you should remember that <term> is a \textit{Regular Expression} and not a fixed string, and also that many files are distributed in source form with an extension different to the final file. (For example \TeX packages are often distributed sources with extension \texttt{dtx} rather than as package files with extension \texttt{sty}.)
One should make the regular expression general enough to find the file you are looking for, but not too general, as the `ftp` interface will only return the first 20 lines that match your request.

The following examples illustrate these points. To search for the \LaTeX\ package `caption', you might use the command:

```
quote site index caption.sty
```

but it will fail to find the desired package (which is distributed as `caption.dtx') and does return unwanted 'hits' (such as `hangcaption.sty'). Also, although this example does not show it the `\.' in `caption.sty' is used as the regular expression that matches any character. So

```
quote site index doc.sty
```

matches such unwanted files as `language/swedish/slatex/doc2sty/makefile'

Of course if you know the package is stored as `.dtx' you can search for that name, but in general you may not know the extension used on the archive. The solution is to add `/' to the front of the package name and `\' to the end. This will then search for a file name that consists solely of the package name between the directory separator and the extension. The two commands:

```
quote site index /caption/\.
quote site index /doc/\.
```

do narrow the search down sufficiently. (In the case of doc, a few extra files are found, but the list returned is sufficiently small to be easily inspected.)

If the search string is too wide and too many files would match, the list will be truncated to the first 20 items found. Using some knowledge of the CTAN directory tree you can usually narrow the search sufficiently. As an example suppose you wanted to find a copy of the `dvips' driver for MS-DOS. You might use the command:

```
quote site index dvips
```

but the result would be a truncated list, not including the file you want. (If this list were not truncated 412 items would be returned!) However we can restrict the search to MS-DOS related drivers as follows.

```
quote site index msdos.*dvips
```

Which just returns relevant lines such as `systems/msdos/dviware/dvips/dvips5528.zip'

A basic introduction to searching with regular expressions is:

- Most characters match themselves, so "a" matches "a" etc.;
- "." matches any character;
- "[abcD-E]" matches any single character from the set {"a","b","c","D","E"};
- "*" placed after an expression matches zero or more occurrences so "a*" matches "a" and "aaaa", and "[a-zA-Z]*" matches a 'word';
- "\" 'quotes' a special character such as "." so "\." just matches ".";
- "^" matches the beginning of a line;
- "$" matches the end of a line.
For technical reasons in the quote site index command, you need to ‘double’ any \ hence the string /caption\ in the above example. The quote site command ignores the case of letters. Searching for caption or CAPTION would produce the same result.

**Part D  Bits and pieces of TeX**

**30  What is a DVI file?**

A DVI file (that is, a file with the type or extension .dvi) is TeX’s main output file, using TeX in its broadest sense to include \LaTeX, etc. ‘DVI’ is supposed to be an acronym for DeVice-Independent, meaning that the file can be printed on almost any kind of typographic output device. The DVI file is designed to be read by a driver (question 31) to produce further output designed specifically for a particular printer (e.g., a LaserJet) or to be used as input to a previewer for display on a computer screen. DVI files use TeX’s internal coding; a TeX input file should produce the same DVI file regardless of which implementation of TeX is used to produce it.

A DVI file contains all the information that is needed for printing or previewing except for the actual bitmaps or outlines of fonts, and possibly material to be introduced by means of \texttt{\special} commands (question 35).

The canonical reference for the structure of a DVI file is the source of dvitype (systems/knuth/texware/dvitype.web).

**31  What is a driver?**

A driver is a program that takes as input a dvi file (question 30) and (usually) produces a file that can be sent to a typographic output device, called a printer for short.

A driver will usually be specific to a particular printer, although any PostScript printer ought to be able to print the output from a PostScript driver.

As well as the DVI file, the driver needs font information. Font information may be held as bitmaps or as outlines, or simply as a set of pointers into the fonts that the printer itself ‘has’.

Each driver will expect the font information in a particular form. For more information on the forms of fonts, see questions 32, 33, 34 and 65.

**32  What are PK files?**

PK files (packed raster) contain font bitmaps. The output from METAfont (question 62) includes a generic font (GF) file and the utility gftopk produces the PK file from that. There are a lot of PK files, as one is needed for each font, that is each magnification (size) of each design (point) size for each weight for each family. Further, since the PK files for one printer do not necessarily work well for another, the whole set needs to be duplicated for each printer family at a site. As a result, they are often held in an elaborate directory structure, or in ‘font library files’, to regularise access.
33 What are TFM files?

TFM stands for TeX font metrics, and TFM files hold information about the sizes of the characters of the font in question, and about ligatures and kerns within that font. One TFM file is needed for each font used by TeX, that is for each design (point) size for each weight for each family; one TFM file serves for all magnifications, so that there are (typically) fewer TFM files than there are PK files. The important point is that TFM files are used by TeX (\LaTeX, etc.), but are not, generally, needed by the printer driver.

34 Virtual fonts

Virtual fonts for TeX were first implemented by David Fuchs in the early days of TeX, but for most people they started when Knuth redefined the format, and wrote some support software, in 1989. Virtual fonts provide a way of telling TeX about something more complicated than just a one-to-one character mapping. The entities you define in a virtual font look like characters to TeX (they appear with their sizes in a font metric file), but the DVI processor may expand them to something quite different. You can use this facility just to remap characters, to make a composite font with glyphs drawn from several sources, or to build up an effect in arbitrarily complicated ways — a virtual font may contain anything which is legal in a DVI file. In practice, the most common use of virtual fonts is to remap PostScript fonts (see question 67) or to build 'fake' maths fonts.

It is important to realise that TeX itself does not see virtual fonts; for every virtual font read by the DVI driver there is a corresponding TFM file read by TeX. Virtual fonts are normally created in a single ASCII vpl (Virtual Property List) file, which includes both sets of information. The vptovf program is then used to the create the binary TFM and VF files. The commonest way (nowadays) of generating vpl files is to use the fontinst package, which is described in detail in question 67. fonts/utilities/qdtexvpl is another utility for creating ad-hoc virtual fonts.

35 \special commands

TeX provides the means to express things that device drivers can do, but about which TeX itself knows nothing. For example, TeX itself knows nothing about how to include PostScript figures into documents, or how to set the colour of printed text; but some device drivers do.

Such things are introduced to your document by means of \special commands; all that TeX does with these commands is to expand their arguments and then pass the command to the DVI file. In most cases, there are macro packages provided (often with the driver) that provide a comprehensible interface to the \special; for example, there’s little point including a figure if you leave no gap for it in your text, and changing colour proves to be a particularly fraught operation that requires real wizardry. \LaTeX 2ε has standard graphics and colour packages that make file inclusion, rotation, scaling and colour via \specials all easy.

The allowable arguments of \special depend on the device driver you’re using. Apart from the examples above, there are \special commands in the emTeX drivers (e.g., dvihplj, dviscr, etc.) that will draw lines at arbitrary orientations, and commands in dvito1n03 that permit the page to be set in landscape orientation.
36 Documented \LaTeX{} sources (\.dtx files)

\LaTeX{} 2ε, and many support macro packages, are now written in a literate programming style (question 53), with source and documentation in the same file. This format, known as ‘doc’, was originated by Frank Mittelbach. The documented sources conventionally have the suffix \texttt{.dtx}, and should normally be stripped of documentation before use with \LaTeX{}. Alternatively you can run \LaTeX{} on a \texttt{.dtx} file to produce a nicely formatted version of the documented code. An installation script (with suffix \texttt{.ins}) is usually provided, which needs the standard \LaTeX{} 2ε \texttt{docstrip} package (among other things, the installation process strips all the comments that make up the documentation for speed when loading the file into a running \LaTeX{} system). Several packages can be included in one \texttt{.dtx} file, with conditional sections, and there facilities for indices of macros etc. Anyone can write \texttt{.dtx} files; the format is explained in \textit{The \LaTeX{} Companion} (see question 20). There are no programs yet to assist in composition.

\texttt{.dtx} files are not used by \LaTeX{} after they have been processed to produce \texttt{.sty} or \texttt{.cls} (or whatever) files. They need not be kept with the working system; however, for many packages the \texttt{.dtx} file is the primary source of documentation, so you may want to keep \texttt{.dtx} files elsewhere.

37 What are the EC fonts?

A font consists of a number of glyphs. In order that the glyphs may be printed, there has to be some way of accessing them; in \TeX{} they’re arranged in a numerical order called an encoding, and their number in the encoding is used. For various reasons, Knuth chose rather eccentric encodings; in particular, he chose different encodings for different fonts.

When \TeX{} version 3 arrived, some at least of the reasons for the eccentricity of Knuth’s encodings went away, and at TUG’s Cork meeting, an encoding for a set of 256 glyphs, for use in \TeX{} text, was defined. The intention was that these glyphs should cover ‘most’ European languages, in the sense of including all accented letters needed. (Knuth’s CMR fonts missed things necessary for Icelandic, Polish and Sami, for example, but the Cork fonts have them.) \LaTeX{} 2ε (question 131) refers to the Cork encoding as T1, and provides the means to use fonts thus encoded to avoid problems with the interaction of accents and hyphenation (see question 108).

The only \texttt{METAFONT}-fonts that conform to the Cork encoding are the EC fonts (available as \texttt{fonts/ec}). They look CM-like, and are now regarded as ‘stable’ (in the same sense that the CM fonts are stable: their metrics are unlikely ever to change). Their serious disadvantage for the casual user is that they are large — each EC font is roughly twice the size of the corresponding CM font; what’s more, until corresponding fonts for mathematics are produced, the CM fonts must be retained.

The EC fonts supersede the experimental DC fonts, which have now been removed from the archives. They are distributed with a set of ‘Text Companion’ (TC) fonts that provide glyphs for symbols commonly used in text. The TC fonts are encoded according to the \LaTeX{} TS1 encoding, and are not viewed as ‘stable’ in the same way as are the EC fonts are.

The Cork encoding is also implemented by the PSNFSS system (question 65), for PostScript fonts.
38 What is TDS?

TDS stands for the \TeX\ Directory Structure, which is a standard way of organising all the \TeX\-related files on a computer system. The TDS specification can be found on CTAN in tds/draft-standard/tds-0.9996

Most modern distributions conform to the TDS, which provides for both a ‘standard’ hierarchy and a ‘local’ hierarchy. The TDS reserves the name \texttt{texmf} as the name of the root directory (folder) of the hierarchies. Files supplied as part of the distribution are put into the standard hierarchy. The location of the standard hierarchy is system dependent, but on a Unix system it might be at /usr/local/texmf, or /usr/local/share/texmf, or /opt/texmf, or a similar location, but in each case the \TeX\ files will be under the /texmf subdirectory.

There can be multiple ‘local’ hierarchies in which additional files can be put. In the extreme an installation can have a local hierarchy and each user can also have an individual local hierarchy. The location of any local hierarchy is not only system dependent but also user dependent. Again, though, all files should be put under a local /texmf directory.

Part E Acquiring the Software

39 Repositories of \TeX\ material

To aid the archiving and retrieval of of \TeX\-related files, a TUG working group developed the Comprehensive \TeX\ Archive Network (CTAN). Each CTAN site has identical material, and maintains authoritative versions of its material. These collections are extensive; in particular, almost everything mentioned in this article is archived at the CTAN sites, even if its location isn’t explicitly stated.

The CTAN sites are currently dante.ctan.org (Mainz, Germany), cam.ctan.org (Cambridge, UK) and tug.ctan.org (Colchester, Vermont, USA). Two of the CTAN web servers offer a search facility: http://www.tex.ac.uk/search and http://www.ctan.org/search; you can look for a file whose name you already know, or you can do a keyword-based search of the catalogue. The search script requires that you choose an appropriate CTAN site or mirror to retrieve files from, and stores details of that site in a cookie on your machine. Choose a site that is close to you, to reduce network load.

The organisation of \TeX\ files on all CTAN sites is identical and starts at \texttt{tex-archive/}. Again, to reduce network load, please use the CTAN site or mirror closest to you. A complete and current list of CTAN sites and known mirrors can be obtained by using the \texttt{finger} utility on ‘user’ ctan@cam.ctan.org, ctan@dante.ctan.org or ctan@tug.ctan.org; it is also available as file CTAN.sites

To find software at a CTAN site using anonymous ftp to the host, execute the command ‘quote site index <term>’ (see question 29 for details).

The email servers ftpmail@dante.ctan.org and ftpmail@tug.ctan.org provide an ftp-like interface through mail. Send a message containing just the line ‘help’ to your nearest server, for details of use.

There is also the DECUS \TeX\ collection of material for VMS, Unix, MS-DOS, and the Macintosh. It is available via anonymous ftp from wuarchive.wustl.edu (128.252.135.4) in
decus/tex/. It can also be obtained from the DECUS Library (reference number VS0058) in the US, or through your DECUS office outside of the US. To contact the DECUS Library, send mail or telephone:

DECUS
LIBRARY ORDER PROCESSING
334 South Street, SHR3-1/T25
Shrewsbury, MA 01545-4195
USA
Tel: 800-DECUS55 (within the USA, for information)
Fax: +1 508-841-3373 (for inquiries)

or send electronic mail for information to the DECUS TeX Collection Editor, Ted Nieland (nieland@ted.hcst.com).

Finally, of course, the TeX user who has no access to any sort of network may buy a copy of the archive on CD-ROM (see question 43).

40 What’s the CTAN nonfree tree?

The CTAN archives are currently restructuring their holdings so that files that are ‘not free’ are held in a separate tree. The definition of what is ‘free’ (for this purpose) is influenced by, but not exactly the same as the Debian Free Software Guidelines (DFSG: see The Debian Social Contract).

Material is placed on the nonfree tree if it is not freely-usable (e.g., if the material is shareware, commercial, or if its usage is not permitted in certain domains at all, or without payment). Users of the archive should check that they are entitled to use material they have retrieved from the nonfree tree.

For details of the licence categories, see


41 Contributing a file to the archives

Use anonymous ftp to any CTAN archive (see question 39) and retrieve the file README.uploads in the root directory. It contains instructions for uploading files and notifying the appropriate people for that site.

If you cannot use ftp, mail your contribution to ctan@urz.Uni-Heidelberg.de and it will be passed along. You will make everyone’s life easier if you choose a descriptive and unique name for your submission, so it’s probably a good idea to check that your style file’s name is not already in use by means of the ‘site index’ command (question 29).

42 Finding new fonts

A comprehensive list of METAFONT fonts is posted to comp.fonts and to comp.text.tex, roughly every six weeks, by Lee Quin (lee@sq.sq.com); it is available as info/metafont-list.

URI: http://www.debian.org/social_contract#guidelines
The list contains details both of commercial fonts and of fonts available via anonymous ftp. Most of the fonts are available via anonymous ftp from the CTAN archives (see question 39).

43 \TeX\ CD-ROMs

If you don’t have access to the Internet, there are obvious attractions to \TeX\ collections on a CD-ROM. Even those with net access will find large quantities of \TeX\-related files to hand a great convenience.

Ready-to-run \TeX\ systems on CD-ROM are available:

- A consortium of User Groups (notably TUG, UK TUG and GUTenberg) distribute the \TeX\ Live CD-ROM, now in its third edition. All members of several User Groups are receiving copies, and additional copies may be purchased, for example, from UK TUG for £25, including an edition of Baskerville which serves as a manual.

  On line details of \TeX\ Live are available at \texttt{http://www.tug.org/texlive.html}

- The Dutch \TeX\ Users Group (NTG) publish the whole 4All\TeX\ workbench on a 2-CD-ROM set packed with all the MS-DOS \TeX\ software, macros and fonts you can want. It is available from NTG direct (see question 18), from TUG for $40 and from UK TUG for £30 (a manual is included). It is a useful resource for anyone to browse, not just for MS-DOS users.

An alternative to the ready-to-run system is the CTAN archive snapshot; in general one would expect that such systems would be harder to use, but that the volume of resources offered would balance this extra inconvenience.

Walnut Creek CDROM provide a two-disc CD-ROM set, holding 1000Mb of \TeX\-related material. Information about the CD-ROM is available at

\texttt{http://www.cdrom.com/titles/prog/tex.html},

which also has a link to an ordering page. Walnut Creek’s address, etc., are:

Walnut Creek CDROM
4041 Pike Lane, Ste D-www
Concord, CA 94520
USA
Tel: +1 510 674-0783 or
800 786-9907 (within the USA and Canada)
Fax: +1 510 674-0821
Email: \texttt{info@cdrom.com} (for questions) and
\texttt{orders@cdrom.com} (for orders)

Walnut Creek (who run one of the major CTAN mirrors) are rumoured to be producing a new release of their disc set in 1997.

Prime Time Freeware produced \TeX\cetera 1.1 in July 1994, which was a snapshot of CTAN taken in June 1994. Regular updates were planned (but have not apparently been forthcoming). The material is all compressed in ZIP format to fit it all on one CD, and to avoid the limitations of the ISO 9660 file system directory. You can buy the CD from:
or from many CD-ROM resellers, or the question 17 office. Price will be around $60. Please note that PTF is not a big commercial firm, and is a good friend of the TeX community.

Part F   TeX Systems

44  (Iß)TeX for different machines

We list here the free or shareware packages; see question 46 for details of commercial packages.

Unix  Instructions for retrieving the Unix TeX distribution via anonymous ftp are available in the document systems/unix/unixtex.ftp, though nowadays the sensible installer will take (and possibly customise) one of the packaged distributions such as teTeX, or the TeX Live CD-ROM (see question 43).

For teTeX, browse systems/unix/teTeX/1.0/distrib/sources for relevant files: you need at most one each of the .tar.gz files for teTeX-src, teTeX-texmf and teTeX-texmfsrc.

Sets of binaries for many common Unix systems are to be found as part of the teTeX distribution, or on the TeX Live CD-ROM. For teTeX binaries, browse systems/unix/teTeX/1.0/distrib/sources-bin — there's a compressed .tar archive for each supported architecture in the directory. In default of a precompiled version, teTeX will compile on most Unix systems, though it was originally developed for use under Linux (see below).

Linux  There are at least two respectable implementations of TeX to run on Linux, NTTeX (available as systems/unix/linux/ntex) and teTeX (browse systems/unix/teTeX/1.0/distrib/sources).

Beware the Slackware '96 CD-ROM distribution of NTTeX: it includes a version of the CM fonts that has deeply offended Don Knuth (since it contravenes his distribution conditions). The Slackware updates now offer teTeX, as do most Linux distributions.

PC  The emTeX package for PCs running OS/2, MS-DOS or Windows includes lOTeX, BibTeX, previewers, and drivers, and is available in systems/msdos/emtex as a series of zip archives. The package was written by Eberhard Mattes, and documentation is available in both German and English. Appropriate memory managers for using emTeX with 386 (and better) processors and under Windows, are included in the distribution.

The most recent offering is an MS-DOS port of the Web2C 7.0 implementation, using the GNU djgpp compiler. It is available from systems/msdos/djgpp.

PC: Win32  fpTeX, by Fabrice Popineau, is a version of TeX for Windows systems. As such, it is particularly attractive to those who need to switch back and forth between Windows
and Unix environments, and to administrators who need to maintain both (fpTeX can use the same \texttt{texmf} tree as a \texttt{tex} installation). \texttt{fpTeX}'s previewer (\texttt{windvi}) is based on \texttt{xdvi}, and takes advantage of extra facilities in the Win32 environment. Printing is available via \texttt{dvips} only. \texttt{fpTeX} is available from \texttt{systems/win32/fptex}.

\texttt{MikTeX}, by Christian Schenk, is also a comprehensive distribution, developed separately from the \texttt{tcTeX} work. It has its own previewer, YAP, which is itself capable of printing, though the distribution also includes a port of \texttt{dvips}. The current version is in \texttt{systems/win32/miktex}.

\textbf{Windows NT, other platforms} A Power PC port of \texttt{MikTeX} is available from \texttt{systems/win32/miktexppc}, and an AXP port is available from \texttt{systems/win32/miktex-AXP}; these versions have not been updated for version 1.2 (and later) of \texttt{MikTeX} — they may not be satisfactory.

\textbf{Mac} \texttt{OzTeX} is a shareware version of \texttt{TeX} for the Macintosh. A DVI previewer and PostScript driver are also included. It should run on any Macintosh Plus, SE, II, or newer model, but will not work on a 128K or 512K Mac. It was written by Andrew Trevorrow, and is available in \texttt{nonfree/systems/mac/oztex}.

UK TUG prepaies the shareware fee, so that its members may acquire the software without further payment. Questions about \texttt{OzTeX} may be directed to \texttt{oztex@midway.uchicago.edu}.

Another partly shareware program is \texttt{CMacTeX} (available as \texttt{systems/mac/cmactex}), put together by Tom Kiffe. This is much closer to the Unix \texttt{TeX} setup (it uses \texttt{dvips}, for instance). \texttt{CMacTeX} includes a port of the latest version of Omega (question 133).

A useful resource for Mac users is to be found at \texttt{http://www.esm.psu.edu/mac-tex/}; it has a news and ‘help’ section, as well as details of systems and tools.

\textbf{OpenVMS} \texttt{TeX} for OpenVMS is available as \texttt{systems/OpenVMS/TEX97_CTAN.ZIP}.

Standard tape distribution is through DECUS (see question 39).

\textbf{Atari} \texttt{TeX} is available for the Atari ST in \texttt{systems/atari}.

If anonymous \texttt{ftp} is not available to you, send a message containing the line ‘\texttt{help}’ to \texttt{atari@atari.archive.umich.edu}.

\textbf{Amiga} Full implementations of \texttt{TeX} 3.1 (Pas\texttt{TeX}) and \texttt{METAFONT} 2.7 are available in \texttt{systems/amiga}.

You can also order a CD-ROM containing this and other Amiga software from Walnut Creek CDROM, telephone +1 510-947-5997.

\textbf{TOPS-20} \texttt{TeX} was originally written on a DEC-10 under WAITS, and so was easily ported to TOPS-20. A distribution that runs on TOPS-20 is available via anonymous \texttt{ftp} from \texttt{ftp.math.utah.edu} (128.110.198.34) in \texttt{pub/tex/pub/web}.

\section{\texttt{TeX}-friendly editors and shells}

There are good \texttt{TeX}-writing environments and editors for most operating systems; some are described below, but this is only a personal selection:
Unix  Try GNU emacs, and the AUCTeX mode (support/auctex). This provides menu items and control sequences for common constructs, checks syntax, lays out markup nicely, lets you call \TeX and drivers from within the editor, and everything else like this that you can think of. Complex, but very powerful.

VMS An lsedit mode for editing \TeX source is available from TUG (question 17) as \TeXniques 1, VAX Language-Sensitive Editor, by Kent MacPherson (1985).

MS-DOS There are several choices:

- The (shareware) 4All\TeX workbench (systems/msdos/4alltex) provides a very comprehensive environment written in 4DOS which lets you access most \TeX-related software in a friendly way. You can choose your own editor; something such as Q\Edit or Brief is suitable. This whole package is available in easy-to-use form on CD-ROM from \TeX user groups.

- \TeXshell (systems/msdos/texshell) is a simpler, easily-customisable environment, which can be used with the editor of your choice.

- Eddi4\TeX (systems/msdos/e4t; also shareware) is a specially-written \TeX editor which features intelligent colouring, bracket matching, syntax checking, online help and the ability to call \TeX programs from within the editor. It is highly customisable, and features a powerful macro language.

You can also use GNU emacs and AUCTeX under MS-DOS.

Windows 3.1 Your best public domain bet is probably to use MicroEmacs as an editor and control centre for \TeX programs.

\TeXtelmExtel (systems/msdos/emtex-contrib/TeXtelmExtel) is a Shell for cm\TeX or WT\TeX and related tools under Windows. It includes a simple multiple-document editor, a built-in spelling checker, automatic OEM/ANSI character conversion, user-definable point-and-click Templates, support for the forward and inverse search mechanism of DVI driver for Windows and for automatic font generation. Besides the predefined tools, up to 10 user-defined tools can be set up.

On a PC with large enough memory, a version of GNU emacs, that will run under Windows, is available; thus you can also use AUCTeX under Windows.

Y&Y’s commercial (and high-quality) Windows previewer, dviwindo, can be used as a good \TeX shell, calling programs such as \TeX, drivers, and editors (Y&Y supply the public domain PE, and recommend the commercial Epsilon) from customisable menus (see question 46 for details of Y&Y).

Scientific Word is a WYSIWYG editing program, strong on maths, which uses \LaTeX for output (see question 46 for contact address).

Windows ’9x, NT Winedt, a shareware package (systems/win32/winedt/winedt32.exe), is highly spoken of. It provides a shell for the use of \TeX and related programs, as well as a powerful and well-configured editor.

OS/2 Eddi4\TeX works under OS/2; look also at systems/os2/epmtex for a specific OS/2 shell.

Macintosh The commercial Textures provides an excellent integrated Macintosh environment with its own editor. More powerful still (as an editor) is the shareware Alpha (systems/
mac/support/alpha) which is extensible enough to let you perform almost any \TeX-
related job. It works well with Oz\TeX.

Atari, Amiga and NeXT users also have nice environments. \LaTeX users who like make should try support/latexmk

There is another set of shell programs to help you manipulate Bib\TeX databases.

### 46 Commercial \TeX implementations

There are many commercial implementations of \TeX. The first appeared not long after \TeX itself appeared. Of the vendors, ArborText (formerly Textset) and Personal \TeX are those who have survived longest (since the mid or early 80s).

What follows is probably an incomplete list. Naturally, no warranty or fitness for purpose is implied by the inclusion of any vendor in this list. The source of the information is given to provide some clues to its currency.

In general, a commercial implementation will come ‘complete’, that is, with suitable previewers and printer drivers. They normally also have extensive documentation (i.e., not just the \TeXbook!) and some sort of support service. In some cases this is a toll free number (probably applicable only within the USA and or Canada), but others also have email, and normal telephone and fax support.

**Unix; \TeX** Silicon Graphics Iris/Indigo, Solaris 2.1, IBM RS/6000, DEC/RISC-Ultrix, HP 9000. “Complete \TeX packages. Ready to use, fully documented and supported.”

ArborText Inc
1000 Victors Way
Suite 400
Ann Arbor MI 48108
USA
Tel: +1 313-996-3566
Fax: +1 313-996-3573

Source: *TUGboat* 15(1) (1994)

**VAX/VMS; Convergent \TeX** Complete system for VAX/VMS machines (a version for Alphas is in preparation); includes \LaTeX, multinational typesetting support, METAFONT and Web.

Northlake Software, Inc.
812 SW Washington, Ste 1100
Portland, OR 97201
USA
Tel: +1 503-228-3383
Fax: +1 503-228-5662
Email: rau@nls.com

Source: Email from Pat Rau, November 1994

**PC; True\TeX** Runs on Windows 3.1, Window NT and Windows 95.

Richard J. Kinch
PC; \TeX  \textit{“}Bitmap free \TeX \textit{for Windows.”}\textit{”}

Y\&Y, Inc.
45 Walden Street
Concord MA 01742
USA
Tel: 800-742-4059 (within the USA)
Tel: +1 508-371-3286
Fax: +1 508-371-2004
Email: sales-help@YandY.com and
tech-help@YandY.com
Web: \url{http://www.YandY.com/}

Source: Y\&Y announcement, February 1995

\textbf{pcT\TeX} Long-established: \textbf{pcT\TeX}32 is a Windows implementation.

Personal \TeX Inc
12 Madrona Street
Mill Valley, CA 94941
USA
Tel: 800-808-7906 (within the USA)
Fax: +1 415-388-8865
Email: texsales@pctex.com and
texsupp@pctex.com
Web: \url{http://www.pctex.com/}

Source: Mail from Personal \TeX Inc, September 1997

\textbf{PC; VT\TeX} DVI, PDF and HTML backends, Visual Tools and Type 1 fonts

MicroPress Inc
68-30 Harrow Street
Forest Hills, NY 11375
USA
Tel: +1 718-575-1816
Fax: +1 718-575-8038
Email: support@micropress-inc.com
Web: \url{http://www.micropress-inc.com/}

Source: Mail from MicroPress, Inc., July 1999

\textbf{PC; Scientific Word} Scientific Word and Scientific Workplace offer a mechanism for near-
WYSIWYG input of \LaTeX \textit{documents}; they ship with True\TeX from Kinch (see above).
Queries within the UK and Ireland should be addressed to Scientific Word Ltd., others should be addressed directly to the publisher, MacKichan Software Inc.

Dr Christopher Mabb
Scientific Word Ltd.
49 Queen Street
Peterhead
Aberdeen, AB42 1TU
UK
Tel: 0845 766 0340 (within the UK)
Tel: +44 1779 490500
Fax: 01779 490600 (within the UK)
Email: christopher@sciword.demon.co.uk
Web: http://www.sciword.demon.co.uk

MacKichan Software Inc.
600 Ericksen Ave. NE, Suite 300
Bainbridge Island WA 98110
USA
Tel: +1 206 780 2799
Fax: +1 206 780 2857
Email: info@mackichan.com
Web: http://www.mackichan.com

Source: Mail from Christopher Mabb, May 1999

**Macintosh: Textures** “A TeX system ‘for the rest of us’”; also gives away a METAFont implementation and some font manipulation tools.

Blue Sky Research
534 SW Third Avenue
Portland, OR 97204
USA
Tel: 800-622-8398 (within the USA)
Tel: +1 503-222-9571
Fax: +1 503-222-1643
Email: sales@bluesky.com
Web: http://www.bluesky.com/

Source: *TUGboat* 15(1) (1994)

**AmigaTeX** A full implementation for the Commodore Amiga, including full, on-screen and printing support for all PostScript graphics and fonts, IFF raster graphics, automatic font generation, and all of the standard macros and utilities.

Radical Eye Software
PO Box 2081
Stanford, CA 94309
USA

Source: Mail from Tom Rokicki, November 1994
Part G  DVI Drivers and Previewers

47  DVI to PostScript conversion programs

The best public domain DVI to PostScript conversion program which runs under many operating systems is Tom Rokicki’s dvips. dvips is written in C and ports easily to other operating systems; it is available as dviware/dvips

VMS versions are available through the DECUS library (see question 39), and also as part of the CTAN distribution of T\TeX for VMS (systems/OpenVMS/TEX97_CTAN.ZIP).

A precompiled version for MS-DOS is available from systems/msdos/dviware/dvips

Karl Berry’s version of dvips (called dvipsk) has a configure script and path searching code similar to that in his other programs (e.g., web2c); it is available from dviware/dvipsk

Another good portable program is dvitops by James Clark, which is also written in C and will compile under Unix, MS-DOS, VMS, and Primos; however, it does not support virtual fonts. It is available from obsolete/dviware/dvitops

Macintosh users can use either the excellent drivers built into OzT\TeX or Textures, or a port of dvips in the C\MacT\TeX package.

48  DVI drivers for HP LaserJet

The emT\TeX package (see question 44) contains a driver for the LaserJet, dvihplj.

Version 2.10 of the Beebe drivers supports the LaserJet. These drivers will compile under Unix, VMS, and on the Atari ST and DEC-20’s, and are available from dviware/beebe

Karl Berry’s dviljk, which has the same path-searching code as his dvipsk (see question 47), is available in dviware/dviljk

49  DVI previewers

EmT\TeX for PCs running MS-DOS or OS/2, MikT\TeX and fpT\TeX for PCs running Windows and OzT\TeX for the Macintosh, all come with previewers that can be used on those platforms. EmT\TeX’s previewer can also be run under Windows 3.1, as can the public domain Windows previewer dviwin (dviware/dviwin).

Commercial PC T\TeX packages (see question 46) have good previewers for PCs running Windows, or for Macintoshes. For Unix systems, there is one ‘canonical’ viewer, xdvi. The basic distribution is available as dviware/xdvi, and a version integrated with the current state of web2c is available from dviware/xdvik; Unix T\TeX distributions (such as teT\TeX or NT\TeX) include a version of xdvik which is compatible with the version of web2c that they use.

Alternatives to previewing include

- conversion to ‘similar’ ASCII text (using dviware/dvi2tty or something of the sort) and using a conventional text viewer to look at that,
• generating a PostScript version of your document and viewing it with a ghostscript previewer, and
• generating PDF output, and viewing that with Acrobat Reader or one of the substitutes for that.

Part H Support Packages for \TeX

50 Fig, a \TeX-friendly drawing package

\(\text{(X)Fig}\) is a menu driven tool that allows you to draw objects on the screen of an X workstation. \text{transfig} is a set of tools which translate the code fig produces to other graphics languages including PostScript and the \LaTeX picture environment. They are available in \texttt{graphics/xfig} and \texttt{graphics/transfig}.

\texttt{Fig} is supported by Micah Beck (beck@cs.cornell.edu) and \texttt{transfig} is maintained by Brian Smith (bvsmith@lbl.gov). Another tool for \texttt{fig} conversion is \texttt{fig2mf} which generates \texttt{META-FONT} code from \texttt{fig} input. It is available in \texttt{graphics/fig2mf}.

51 \TeX\textsc{CAD}, a drawing package for \LaTeX

\TeX\textsc{CAD} is a program for the PC which enables the user to draw diagrams on screen using a mouse or arrow keys, with an on-screen menu of available picture-elements. Its output is code for the \LaTeX picture environment. Optionally, it can be set to include lines at all angles using the em\TeX driver-family \texttt{\textbackslash specials} (question 35). \TeX\textsc{CAD} is part of the em\TeX distribution.

A Unix port of the program exists, \texttt{graphics/xtexcad/xtexcad-2.4.tar.gz}

52 Spelling checkers for work with \TeX

For Unix, \texttt{ispell} is probably the program of choice. Browse \texttt{support/ispell} for a version, but beware of any with a number 4.x — such versions represent a divergent version of the source which lacks many useful facilities of the 3.x series.

For MS-DOS, there are several programs. \texttt{amspell} can be called from within an editor (available as \texttt{support/amspell}). \texttt{jspell} is an extended version of \texttt{ispell} (available as \texttt{support/jspell}).

For the Macintosh, Excalibur is the program of choice. It will run in native mode on both sorts of Macintosh, and is available as \texttt{systems/mac/support/excalibur/Excalibur-2.6-sit.hqx} (there are other dictionaries in the same directory).

For VMS, a spell checker can be found in \texttt{support/vmspell}
Part I  Literate programming

53  What is Literate Programming?

Literate programming is the combination of documentation and source together in a fashion suited for reading by human beings. In general, literate programs combine source and documentation in a single file. Literate programming tools then parse the file to produce either readable documentation or compilable source. The WEB style of literate programming was created by D. E. Knuth during the development of his T\TeX\ typesetting software.

Discussion of literate programming is conducted in the newsgroup comp.programming.literate. The literate programming FAQ is stored as help/LitProg-FAQ

54  WEB for C, FORTRAN, and other languages

T\TeX\ is written in the programming language WEB; WEB is a tool to implement the concept of “literate programming”.

CWEB, a WEB for C programs, written by Silvio Levy, is available as web/c_cpp/cweb

Spidery WEB supports many languages including Ada, awk, and C. It was written by Norman Ramsey and, while not in the public domain, is usable free. It is available in web/spiderweb

FWEB is a version for Fortran, Ratfor, and C written by John Krommes. It is available in web/fweb

SchemeWEB is a Unix filter that translates SchemeWEB into \LaTeX\ source or Scheme source. It was written by John Ramsdell and is available in web/schemeweb

APLWEB is a version of WEB for APL and is available in web/apl/aplweb

FunnelWeb is a version of WEB that is language independent. It is available in web/funnelweb

Other language independent versions of WEB are nuweb (which is written in ANSI C), available in web/nuweb, and noweb, available in web/noweb

A WEB for plain T\TeX\ macro files, using noweb, has recently been made available in web/tweb

Part J  Format conversions

55  Conversion between (\LaTeX) and others

troff troff-to-latex (available as indexing/makeindex/troff-to-latex), written by Kamal Al-Yahya at Stanford University (California, USA), assists in the translation of a troff document into \LaTeX\ format. It recognises most -ms and -man macros, plus most eqn and some tbl preprocessor commands. Anything fancier needs to be done by hand. Two style files are provided. There is also a man page (which converts very well to \LaTeX\...). The program is copyrighted but free. An enhanced version of this program, tr2latex, is available in support/tr2latex

The DECUS T\TeX\ distribution (see question 39) also contains a program which converts troff to T\TeX\.
WordPerfect \texttt{wp2latex} (available as \texttt{support/wp2latex}) has recently been much improved, and is now available either for MS-DOS or for Unix systems, thanks to its current maintainer Jaroslav Fojtik.

PC-Write \texttt{pcwritex.arc}, from \texttt{support/pcwritex}, is a print driver for PC-Write that “prints” a PC-Write V2.71 document to a \TeX-compatible disk file. It was written by Peter Flynn at University College, Cork, Republic of Ireland.

\texttt{runoff} Peter Vanroose’s (vanroose@esat.kuleuven.ac.be) conversion program is written in VMS Pascal. The sources and a VAX executable are available from \texttt{support/rnototex}.

\texttt{refer/tib} There are a few programs for converting bibliographic data between \BibTeX and \texttt{refer/tib} formats. They are in \texttt{bibliobibdex/utilis/refer-tools}.

In spite of the directory name, it also contains a shell script to convert \BibTeX to \texttt{refer} as well. The collection is not maintained.

\texttt{RTF} A program for converting Microsoft’s Rich Text Format to \TeX is available in \texttt{support/rtf2tex}, which was written and is maintained by Robert Lupton (rhl@astro.princeton.edu). There is also a convertor to \LaTeX by Erwin Wechtl, in \texttt{support/rtf2latex}.

Translation to RTF may be done (for a somewhat constrained set of \LaTeX documents) by \TeX2RTF, which can produce ordinary RTF, Windows Help RTF (as well as HTML, question 58). \TeX2RTF is supported on various Unix platforms and under Windows 3.1; it is available from \texttt{support/tex2rtf}.

\texttt{Microsoft Word} A rudimentary program for converting MS-Word to \LaTeX is \texttt{wd2latex}, for MS-DOS (\texttt{dviware/wd2latex}); a better idea, however, is to convert the document to RTF format and use one of the RTF converters mentioned above.

A FAQ that deals specifically with conversions between \TeX-based formats and word processor formats is regularly posted to \texttt{comp.text.tex}, is available via

\begin{verbatim}
http://www.kfa-juelich.de/isr/1/texconv/texconv.html
\end{verbatim}

and is archived as \texttt{help/wp-conv/wp-conv.zip}.

A group at Ohio State University (USA) is working on a common document format based on SGML, with the ambition that any format could be translated to or from this one. \texttt{FrameMaker} provides “import filters” to aid translation from alien formats (presumably including \TeX) to \texttt{Framemaker}’s own.

56 Conversion from \(\mathbb{B}\)\TeX to plain ASCII

The aim here is to emulate the Unix \texttt{nroff}, which formats text as best it can for the screen, from the same input as the Unix typesetting program \texttt{troff}.

Ralph Droms (droms@bucknell.edu) has a style file and a program that provide the \BIBTeX equivalent of \texttt{nroff}, though it doesn’t do a good job with tables and mathematics. The software is available in \texttt{support/txt}; the original \texttt{dvi2tty} often does an acceptable job and is available in \texttt{dviware/dvi2tty}.

36
Another possibility is to use `screen.sty`, available as `macros/latex209/contrib/misc/screen.sty`. Use a `dvi2tty` program of some kind; you might try `dviware/crudetype` as well. Another possibility is to use the `\LaTeX`-to-ASCII conversion program, `l2a` (`support/l2a`), although this is really more of a de-\TeX{}ing program.

The canonical de-\TeX{}ing program is `detex` (`support/detex`), which removes all comments and control sequences from its input before writing it to its output. Its original purpose was to prepare input for a dumb spelling checker.

57 Conversion from SGML or HTML to \TeX{}

SGML is a very important system for document storage and interchange, but it has no formatting features; its companion ISO standard DSSSL (http://www.jclark.com/dsssl/) is designed for writing transformations and formatting, but this has not yet been widely implemented. Some SGML authoring systems (e.g., SoftQuad Author/Editor) have formatting abilities, and there are high-end specialist SGML typesetting systems (e.g., Miles33’s Genera). However, the majority of SGML users probably transform the source to an existing typesetting system when they want to print. \TeX{} is a good candidate for this. There are three approaches to writing a translator:

1. Write a free-standing translator in the traditional way, with tools like `yacc` and `lex`; this is hard, in practice, because of the complexity of SGML.

2. Use a specialist language designed for SGML transformations; the best known are probably Omnimark and Balise. They are expensive, but powerful, incorporating SGML query and transformation abilities as well as simple translation.

3. Build a translator on top of an existing SGML parser. By far the best-known (and free!) parser is James Clark’s `nsgmls`, and this produces a much simpler output format, called ESIS, which can be parsed quite straightforwardly (one also has the benefit of an SGML parse against the DTD). Two good public domain packages use this method:


Both of these allow the user to write ‘handlers’ for every SGML element, with plenty of access to attributes, entities, and information about the context within the document tree.

If these packages don’t meet your needs for an average SGML typesetting job, you need the big commercial stuff.

Since HTML is simply an example of SGML, we do not need a specific system for HTML. However, Nathan Torkington (Nathan.Torkington@vuw.ac.nz) developed `html2latex` from the HTML parser in NCSA’s Xmosaic package. The program takes an HTML file and generates a \TeX{} file from it. The conversion code is subject to NCSA restrictions, but the whole source is available as `support/html2latex`.

37
Michel Goossens and Janne Saarela published a very useful summary of SGML, and of public domain tools for writing and manipulating it, in *TUGboat* 16(2).

58 *(L)TEX conversion to HTML*

*TEX* is a typesetting language, not a markup system. With properly-used *TEX*, you may be luckier, but don’t expect a free lunch. Remember that a) if you want a really good Web document, you had better redesign it from scratch, and b) HTML (even HTML3) has pretty poor ‘typesetting’ facilities, and anything beyond the trivial will probably need to end up a graphic.

*TEX2HTML* ([support/latex2html](#)) is a package by Nikos Drakos (mostly of *perl* scripts) that breaks up a *TEX* document into one or more components, and links them together so that they can be read over the World-Wide Web as an hypertext document. It defines a mapping between *TEX* intra-document references and hyperlinks, and extends the mechanisms to permit reference to other (possibly remote) documents and other Internet resources. It translates *TEX* accented and other characters (as best it can) to things that World-Wide Web browsers can display, and translates mathematics (and other things that browsers can’t deal with) to images that can be loaded in-line into the hypertext document.

*TEX2HTML* needs *perl*, the PBM utilities, *dvips*, *Ghostscript*, and other sundries; it assumes it is running on a Unix system. Michel Goossens and Janne Saarela published a detailed discussion of *TEX2HTML*, and how to tailor it, in *TUGboat* 16(2).

There are two alternative strategies:

1. Free-standing *TEX* to HTML translations. Hard, but not impossible. Julian Smart’s [latex2rtf](#) (available from [support/latex2rtf](#)) does a plausible job on a subset of *TEX*

2. Writing an HTML-output backend in *TEX* itself. See Sebastian Rahtz’ paper in *TUGboat* 16(3) for a discussion of how to go about this for the general case of SGML.

59 Making hypertext documents from *TEX*

If you want on-line hypertext with a *(L)TEX* source, probably on the World Wide Web, consider four technologies (which overlap):

1. Try direct *TEX* conversion to HTML; see question 58;

2. Rewrite your document using Texinfo (see question 14), and convert that to HTML;

3. Look at Adobe Acrobat, an electronic delivery system guaranteed to preserve your typesetting perfectly. See question 60;

4. Invest in the hyper*TEX* conventions (standardised `\special` commands); there are supporting macro packages for plain *TEX* and *TEX*).

The Hyper*TEX* project aims to extend the functionality of all the *TEX* cross-referencing commands (including the table of contents) to produce `\special` commands which are parsed by DVI processors conforming to the Hyper*TEX* guidelines; it provides general hypertext links, including those to external documents.

38
The HyperTeX specification says that conformant viewers/translators must recognize the following set of \special commands:

```
href: html:<a href = "href_string">
name: html:<a name = "name_string">
end: html:</a>
image: html:<img src = "href_string">
base_name: html:<base href = "href_string">
```

The `href`, `name` and `end` commands are used to do the basic hypertext operations of establishing links between sections of documents.

Further details are available on http://xxx.lanl.gov/hypertex/; there are two commonly-used implementations of the specification, a modified xdvi and (recent releases of) dvips. Output from the latter may be used in recent releases of Ghostscript or Acrobat Distiller.

60 Making Acrobat documents from \LaTeX

There are three general routes to Acrobat output: Adobe's original 'distillation' route (via PostScript output), conversion of an DVI file, and the use of a direct PDF generator such \TeX (see question 135) or MicroPress's VTeX (see question 46).

For simple documents (with no hyper-references), you can either

- process the document in the normal way, produce PostScript output and distill it,
- (on a Windows or Macintosh machine with Acrobat Exchange installed) pass the output through the PDFwriter in place of a printer driver. This route is a dead end: the PDFwriter cannot create hyperlinks.
- process the document in the normal way and process the DVI with dvipdfm (available from dviware/dvipdfm, and on the latest \TeX-live disc), or
- process the document direct to PDF with PDF\TeX or VTeX. PDF\TeX has the advantage of availability for a wide range of platforms, VTeX (available commercially for Windows, or free of charge for Linux — systems/linux/micropress) has wider graphics capability, dealing with encapsulated PostScript and some in-line PostScript.

To translate all the \LaTeX cross-referencing into Acrobat links, you need a \LaTeX package to suitably redefine the internal commands. There are two of these for \LaTeX, both capable of conforming to the HyperTeX specification (see question 59): Sebastian Rahtz's hyperref (available from macros/latex/contrib/supported/hyperref), and Michael Mehlich's hyper (available from macros/latex/contrib/supported/hyper). Hyperref uses a configuration file to determine how it will generate hypertext; it can operate using PDF\TeX primitives, the hyper\TeX \specials, or DVI driver-specific \special commands. Both dvips and Y&Y's DVIPSONE to translate the DVI into PostScript acceptable to Distiller.

There is no free implementation of all of Adobe Distiller's functionality, but Ghostscript (version 4.00 onwards) provides some restricted distilling capability (note the restrictions on
the fonts it can use). However, Distiller itself is now remarkably cheap (for academics at least).

For viewing (and printing) the resulting files, Adobe’s Acrobat Reader is available for a wide range of platforms (see ftp://ftp.adobe.com/pub/adobe/acrobatreader). For those platforms for which Adobe’s reader is unavailable, GhostScript (versions 3.51 onwards) can display and print PDF files.

61 Using \TeX to read SGML or XML directly

This can nowadays be done, with a certain amount of clever macro programming. David Carlisle’s xmltex (macros/xmltex/base) and Jonathan Fine’s SGMLbase (see http://www.active-tex.demon.co.uk) are two examples of packages that have been developed and are publicly available. At the time of writing, xmltex is more mature than SGMLbase, and offers a practical solution to typesetting XML files. SGMLbase is designed to cope with SGML as well as XML, but at present only has ‘toy’ typesetting examples.

One use of a \TeX that can typeset XML files is as a backend processor for XSL formatting objects, serialized as XML. Sebastian Rahtz’s Passive\TeX (macros/xmltex/contrib/passivetex) uses xmltex to achieve this end.

Part K \ \ METAFONT

62 Getting METAFONT to do what you want

METAFONT allows you to create your own fonts, and most \TeX users will never need to use it. METAFONT, unlike \TeX, requires some customisation: each output device for which you will be generating fonts needs a mode associated with it. Modes are defined using the \texttt{mode_def} convention described on page 94 of \textit{The METAFONT Book} (see question 20). You will need a file, which conventionally called \texttt{local.mf}, containing all the \texttt{mode_def}s you will be using. If \texttt{local.mf} doesn’t already exist, Karl Berry’s collection of modes, available as \texttt{fonts/modes-file}, is a good starting point (it can be used as a ‘\texttt{local.mf}’ without modification in a ‘big enough’ implementation of METAFONT). Lists of settings for various output devices are also published periodically in \textit{TUGboat} (see question 17). Now create a \texttt{plain} base file using \texttt{inimf, plain.mf}, and \texttt{local.mf}:

```
% inimf
This is METAFONT...  
**plain # you type plain
 (output)
*input local # you type this
 (output)
*dump # you type this
Beginning to dump on file plain...
 (output)
```

This will create a base file named \texttt{plain.base} (or something similar; for example, it will be \texttt{PLAIN.BAS} on MS-DOS systems) which should be moved to the directory containing the base
files on your system (note that some systems have two or more such directories, one for each 'size' of \texttt{METAFONT} used).

Now you need to make sure \texttt{METAFONT} loads this new base when it starts up. If \texttt{METAFONT} loads the \texttt{plain} base by default on your system, then you’re ready to go. Under Unix (using the default \texttt{web2c} distribution\footnote{The \texttt{command\_name} is symbolically linked to \texttt{virmf}, and \texttt{virmf} loads \texttt{command\_name\_base}.}) this does indeed happen, but we could for instance define a command \texttt{mf} which executes \texttt{virmf &plain} loading the \texttt{plain} base file.

The usual way to create a font with \texttt{plain} \texttt{METAFONT} is to start it with the line

\begin{verbatim}
\mode=<mode name>; mag=<magnification>; input <font file name>
\end{verbatim}

in response to the ‘***’ prompt or on the \texttt{METAFONT} command line. (If \texttt{<mode name>} is unknown or omitted, the mode defaults to ‘proof’ and \texttt{METAFONT} will produce an output file called \texttt{<font file name>.2602gf}) The \texttt{<magnification>} is a floating point number or ‘magstep’ (magsteps are defined in \texttt{The METAFONT Book} and \texttt{The \TeX \book}). If \texttt{mag=<magnification>} is omitted, then the default is 1 (magstep 0). For example, to generate \texttt{cmr10} at 12pt for an epson printer you would type

\begin{verbatim}
mf \mode=epson; mag=magstep 1; input cmr10
\end{verbatim}

Note that under Unix the \ and ; characters must usually be quoted or escaped, so this would typically look something like

\begin{verbatim}
mf '\\mode=epson; mag=magstep 1; input cmr10'
\end{verbatim}

If you don’t have \texttt{inimf} or need a special mode that isn’t in the base, you can put its commands in a file (e.g., \texttt{ln03.mf}) and invoke it on the fly with the \texttt{\smode} command. For example, to create \texttt{cmr10.300gf} for an LN03 printer, using the file

\begin{verbatim}
% This is ln03.mf as of 1990/02/27
% mode\_def courtesy of John Sauter
proofing:=0;
fontmaking:=1;
tracingtitles:=0;
pixels\_per\_inch:=300;
blacker:=0.65;
fillin:=-0.1;
o\_correction:=.5;
\end{verbatim}

(note the absence of the \texttt{mode\_def} and \texttt{enddef} commands), you would type

\begin{verbatim}
mf \smode="ln03"; input cmr10
\end{verbatim}

This technique isn’t one you should regularly use, but it may prove useful if you acquire a new printer and want to experiment with parameters, or for some other reason are regularly editing the parameters you’re using. Once you’ve settled on an appropriate set of parameters, you should use them to rebuild the base file that you use.

A summary of the above written by Geoffrey Tobin, and tips about common pitfalls in using \texttt{METAFONT}, is available as \texttt{info/metafont-for-beginners.tex}.
63 Which font files should be kept

METAFONT produces from its run three files, a metrics (TFM) file, a generic font (GF) file, and a log file; all of these files have the same base name as does the input (e.g., if the input file was cmr10.mf, the outputs will be cmr10.tfm, cmr10.nnngf and cmr10.log).

For TeX to use the font, you need a TFM file, so you need to keep that. However, you are likely to generate the same font at more than one magnification, and each time you do so you’ll (incidentally) generate another TFM file; these files are all the same, so you only need to keep one of them.

To preview or to produce printed output, the DVI processor will need a font raster file; this is what the GF file provides. However, while there used (once upon a time) to be DVI processors that could use GF files, modern processors use packed raster (PK) files. Therefore, you need to generate a PK file from the GF file; the program gftopk does this for you, and once you’ve done that you may throw the GF file away.

The log file should never need to be used, unless there was some sort of problem in the METAFONT run, and need not be ordinarily kept.

64 Getting bitmaps from the archives

Most people these days start using TeX with a 300 dots-per-inch (dpi) laser printer, and Computer Modern bitmap fonts for this resolution are supplied with most TeX packages. There are also two such sets available on CTAN: fonts/cm/pk/pk300.zip (for write-black printer engines) and fonts/cm/pk/pk300w.zip (for write-white engines). However, some users want to send their work to high quality typesetting machines (typically with a resolution of 1270 dpi or greater); it is also becoming more common to use a 600 dpi laser printer. Why don’t the archives or suppliers provide bitmap fonts at these sizes? There are two reasons:

1. When a bitmap font is created with METAFONT, it needs to know the characteristics of the device; who knows what 600 or 1270 dpi device you have? (Of course, this objection applies equally well to 300 dpi printers.)

2. Bitmap fonts get big at high resolutions. Who knows what fonts at what sizes you need?

It would be possible to provide some set of 1270 dpi bitmap fonts in the archives, but it would take a lot of space, and might not be right for you.

So what to do? You can build the fonts you need yourself with METAFONT: this isn’t at all hard, and some drivers help you (dvips, and the emTeX drivers construct the METAFONT commands). You might need to look at Karl Berry’s collection of METAFONT modes (fonts/modes-file). Alternatively, if it is a PostScript device you have, consider using the fonts in Type 1 font format. You can buy all the Computer Modern fonts in outline form from Blue Sky Research, Kinch or Y&Y (question 46 for addresses), or you can use Basil Malyshev’s public domain versions in fonts/cm/ps-type1 (the Paradissa collection is complete, but has largely been replaced by the better BaKoMa collection).

---

8Note that the file name may be transmuted by such operating systems as MS-DOS, which don’t permit long file names.
Part L  PostScript and \TeX

65  Using PostScript fonts with \TeX

In order to use PostScript fonts, \TeX needs metric (called TFM) files. Several sets of metrics are available from the archives; for mechanisms for generating new ones, see question 67. You also need the fonts themselves; PostScript printers come with a set of fonts built in, but to extend your repertoire you almost invariably need to buy from one of the many commercial font vendors (see, for example, question 69).

If you use \LaTeX\,\2\v, the best way to get PostScript fonts into your document is to use the PSNFSS package maintained by Sebastian Rahtz and Alan Jeffrey (available in macros/latex/required/psnfss); it's supported by the \LaTeX\,\3 project team, so bug reports can and should be submitted. PSNFSS gives you a set of packages for changing the default roman, sans-serif and typewriter fonts; e.g., times.sty will set up Times Roman, Helvetica and Courier in place of Computer Modern, while avant.sty just changes the sans-serif family to AvantGarde. To go with these packages, you will need the font metric files (watch out for encoding problems! see question 67) and font description (.fd) files for each font family you want to use. These can be obtained from fonts/pfsfonts, arranged by vendor (e.g., Adobe, Monotype, etc.). For convenience, metrics for the common '35' PostScript fonts found in most printers are provided with PSNFSS, packaged as macros/latex/packages/psnfss/lw35nfss.zip

For older versions of \LaTeX\ there are various schemes, of which the simplest to use is probably the PS\TeX\ macros distributed with dvips.

For \plain\ \TeX, you load whatever fonts you like; if the encoding of the fonts is not the same as Computer Modern it will be up to you to redefine various macros and accents, or you can use the font re-encoding mechanisms available in many drivers and in ps2pk and afm2tfm.

Victor Eijkhout’s sophisticated Lollipop package (macros/lollipop) supports declaration of font families and styles in a similar way to \LaTeX\,\3’s NFSS, and so is easy to use with PostScript fonts.

Some common problems encountered are discussed elsewhere (see question 68).

66  Previewing files using PostScript fonts

Most \TeX\ previewers only display bitmap PK fonts. If you want to preview documents using PostScript fonts, you have three choices:

1. Convert the DVI file to PostScript and use a PostScript previewer. Some modern Unix X implementations have this built in (as does NeXT-step); (X11) Unix, Windows, OS/2, and MS-DOS users can use the free Ghostscript (support/ghostscript), a complete level 2 implementation.

2. Under Windows on a PC, or on a Macintosh, let Adobe Type Manager display the fonts. Textures (Macintosh) works like this, and under Windows you can use Y&Y’s dviwindo for bitmap-free previewing. (See question 46 for details of these suppliers.)

3. If you have the PostScript fonts in Type 1 format, use ps2pk (fonts/utilities/ps2pk) or gsftopk (designed for use with the Ghostscript fonts; fonts/utilities/gsftopk) to
make PK bitmap fonts which your previewer will understand. This can produce excellent results, also suitable for printing with non-PostScript devices. Check the legalities of this if you have purchased the fonts. The very commonest PostScript fonts such as Times and Courier come in Type 1 format on disk with Adobe Type Manager (often bundled with Windows, and part of OS/2).

### 67 TeX font metric files for PostScript fonts

Font vendors such as Adobe supply metric files for each font, in AFM (Adobe Font Metric) form; these can be converted to TFM (TeX Font Metric) form. The CTAN archives have prebuilt metrics which will be more than enough for many people (fonts/psfonts; beware — this directory is at the root of a huge tree), but you may need to do the conversion yourself if you have special needs or acquire a new font. One important question is the encoding of (Latin character) fonts; while we all more or less agree about the position of about 96 characters in fonts (the basic ASCII set), the rest of the (typically) 256 vary. The most obvious problems are with floating accents and special characters such as the ‘pounds sterling’ sign. There are three ways of dealing with this: either you change the TeX macros which reference the characters (not much fun, and error-prone); or you change the encoding of the font (easier than you might think); or you use virtual fonts (question 34) to pretend to TeX that the encoding is the same as it is used to. If you use \LaTeXe, it allows for changing the encoding in TeX read the \LaTeX Companion (see question 20) for more details. In practice, if you do much non-English (but Latin script) typesetting, you are strongly recommended to use the fontenc package with option ‘T1’ to select T1 (‘Cork’: question 37) encoding.

Alan Jeffrey’s fontinst package (fonts/utilities/fontinst) is an AFM to TFM converter written in TeX it is used to generate the files used by \LaTeXe’s PSNFSS package to support use of PostScript fonts. It is a sophisticated package, not for the faint-hearted, but is powerful enough to cope with most needs. Much of its power relies on the use of virtual fonts (question 34).

For slightly simpler problems, Rokicki’s afm2tfm, distributed with dvips (dviware/dvips), is fast and efficient; note that the metrics and styles that come with dvips are not currently \LaTeXe compatible, but Karl Berry plans to distribute metrics directly compatible with PSNFSS in his dvipsk package.

For the Macintosh, there is a program called EdMetrics which does the job (and more). It comes with the Textures distribution, but is in fact free software, available as systems/mac/textures/utilities/EdMetrics.sea.hqx

MS-DOS users can buy (see question 46) Y&Y’s Font Manipulation Tools package which includes a powerful afmtotfm program among many other goodies.

### 68 Problems using PostScript fonts

For the typical \LaTeX user trying to use the PSNFSS (question 65) package, three questions often arise. First, you have to declare to the DVI driver that you are using PostScript fonts; in the case of dvips, this means adding lines to the psfonts.map file. Otherwise, dvips will try to find PK files. If the font isn’t built into the printer, you have to acquire it (in many cases this means buying it from a commercial supplier!). You then have to instruct the driver to download it with each job (the mechanism depends on your driver). So it’s no good just
installing the metrics for Optima and expecting it to work. You have to pay hard cash for the
font itself, which will come (for Unix and MS-DOS users) in pfb (Printer Font Binary) form.

Second, you cannot expect your previewer to suddenly start displaying PostScript fonts; most of
them only know about PK bitmap fonts such as Computer Modern. ps2pk (fonts/utilities/
pk2ps) can create these from the pfb file you have bought; this would also let you use the fonts
with non-PostScript device drivers such as the emTeX ones. You are responsible for making
sure you are not breaking the licence restrictions on font you bought.

Third, the stretch and shrink between words is a function of the font metric; it is not specified
in AFM files, so different converters choose different values. The PostScript metrics that come
with PSNFSS used to produce quite tight setting, but they were revised in mid 1995 to produce
a compromise between American and European practice. Really sophisticated users may not
find even the new the values to their taste, and want to override them. Even the casual user may
find more hyphenation or overfull boxes than CMR produces; but CMR is extremely generous.

69  Choice of scalable outline fonts

If you are interested in text alone, you can use any of over 20,000 fonts(!) in Adobe Type 1
format (called ‘PostScript fonts’ in the TEx world and ‘ATM fonts’ in the DTP world), or any
of several hundred fonts in TrueType format. That is, provided of course, that your previewer
and printer driver support scalable outline fonts.

TEx itself only cares about metrics, not the actual character programs. You just need to
create a TEx metric file TFM using some tool such as afm2tfm, afmtotfm (from Y&Y, see
question 46) or fontinst. For the previewer or printer driver you need the actual outline font
files themselves (pfa for Display PostScript, pfb for ATM on IBM PC, Mac outline font files
on Macintosh).

If you also need mathematics, then you are severely limited by the demands that TEx makes of
maths fonts (for details, see the paper by B.K.P. Horn in TUGboat 14(3)). For maths, then,
there are relatively few choices:

Computer Modern (75 fonts — optical scaling) Donald E. Knuth
Note that CM is available in scalable outline form. There are commercial as well as public
domain versions, and there are both Adobe Type 1 and TrueType versions. Some of these
are ‘commercial grade,’ with full hand-tuned hinting, some render very poorly, while others
are merely incompatible with Adobe Type Manager (ATM).

Lucida Bright with Lucida New Math (25 fonts) Chuck Bigelow and Kris Holmes
Lucida is a family of related fonts including serifed, sans serif, sans serif fixed width, cal-
ligraphic, blackletter, fax, Kris Holmes’ connected handwriting font, etc; they’re not as
‘spindly’ as Computer Modern, with a large x-height, and include a larger set of maths
symbols, operators, relations and delimiters than CM (over 800 instead of 384: among oth-
ers, it also includes the AMS msam and msbm symbol sets). The planned ‘Lucida Bright
Expert’ (14 fonts) adds serifed fixed width, another handwriting font, smallcaps, bold
maths, upright ‘maths italic’, etc., to the set. The distribution includes support for use with
plain TEx and LATEX 2ε. Support under LATEX 2ε is provided in PSNFSS (question 65)
thanks to Sebastian Rahtz.

MathTime 1.1 (3 fonts) TExplorators (Michael Spivak)
The set contains maths italic, symbol, and extension fonts, designed to work well with
Times-Roman. These are typically used with Times, Helvetica and Courier (which are resident on many printers, and which are supplied with some PC versions). In addition you may want to complement this basic set with Adobe’s Times Smallcap, and perhaps the set of Adobe ‘Math Pi’ fonts, which include blackboard bold, blackletter, and script faces. The distribution includes support for use with plain \TeX and \LaTeX 2.09 (including code to link in Adobe Math Pi 2 and Math Pi 6). Support under \LaTeX 2ε is provided in PSNFSS (question 65) thanks to Sebastian Rahtz.

Adobe Lucida, LucidaSans and LucidaMath (12 fonts)
Lucida and LucidaMath are generally considered to be a bit heavy. The three maths fonts contain only the glyphs in the CM maths italic, symbol, and extension fonts. Support for using LucidaMath with \TeX is not very good; you will need to do some work reencoding fonts etc. (In some sense this set is the ancestor of the LucidaBright plus LucidaNewMath font set.)

Concrete, the AMS maths fonts etc. Donald E. Knuth and the AMS.
These are sometimes mentioned as alternatives to CM, but they are really adjuncts, in that you need to use at least the basic CM maths fonts with them.

Proprietary fonts Various sources.
Since having a high quality font set in scalable outline form that works with \TeX can give a publisher a real competitive advantage, there are some publishers that have paid (a lot) to have such font sets made for them. Unfortunately, these sets are not available on the open market, despite the likelihood that they’re more complete than those that are.

Mathptm (4 fonts) Alan Jeffrey.
This set contains maths italic, symbol, extension, and roman virtual fonts, built from Adobe Times, Symbol, Zapf Chancery, and the Computer Modern fonts. The Mathptm fonts are free, and the resulting PostScript files can be freely exchanged. Contains most of the CM math symbols. Support under \LaTeX 2ε in PSNFSS (question 65) thanks to Alan Jeffrey and Sebastian Rahtz.

(A similar development by Walter Schmidt, using the Adobe Palatino fonts, is available from fonts/mathpple)

All of the first three font sets are available in formats suitable for IBM PC/Windows, Macintosh and Unix/NeXT from Y&Y and from Blue Sky Research (see question 46 for details). The MathTime fonts are also available from:

\TeXplorators
1572 West Gray #377
Houston TX 77019
USA

The very limited selection of maths font sets is a direct result of the fact that a maths font has to be explicitly designed for use with \TeX and as a result it is likely to lose some of its appeal in other markets. Furthermore, the \TeX market for commercial fonts is minute (in comparison, for example, to Microsoft TrueType font pack #1, which sold something like 10 million copies in a few weeks after release of Windows 3.1!).

Text fonts in Type 1 format are available from many vendors including Adobe, Monotype, Bitstream. Avoid cheap rip-offs: not only are you rewarding unethical behaviour, destroying the cottage industry of innovative type design, but you are also very likely to get junk. The fonts may not render well (or at all under ATM), may not have the ‘standard’ complement of 228 glyphs, or may not include metric files (needed to make TFM files). Also, avoid TrueType
fonts from all but the major vendors. TrueType fonts are an order of magnitude harder to 'hint' properly than Type 1 fonts and hence TrueType fonts from places other than Microsoft and Apple may be suspect. In any case you may find other problems with TrueType fonts such as service bureaux not accepting jobs calling for them.

70 Including a PostScript figure in (B)\TeX

\LaTeX\2ε (question 131) has a standard package for graphics inclusion, rotation, colour, and other driver-related features. The package is documented in the second edition of Lamport’s \LaTeX\ book, as well as in the \LaTeX\ Graphics Companion (see question 20).

The distribution itself comes with documentation, and a processed copy (grfguide.ps) is available in the distribution so that users can read documentation without first installing the package.

The \texttt{graphics} package comes with a relative, \texttt{graphicx}, which provides more convenient means of scaling and otherwise manipulating graphics. Both packages are configured for the particular processor you’re using (either DVI processor, or PDF\TeX), and the packages cope with any sort of graphics inclusion the processor understands, be it PostScript, bitmap, PDF or anything.

Both \texttt{graphics} and \texttt{graphicx} are to be found on macros/latex/required/graphics

If you don’t use \LaTeX\2ε, a version of the package is available that will work under Plain \TeX (and other formats) — see macros/latex/required/graphics-plain

One point to note about including PostScript figures is that they are not part of the DVI file, but are only included when you use a DVI to PostScript conversion program. The \texttt{special} commands used to do this are potentially different for every DVI processor (which is why the \texttt{graphics} package must be configured for the processor you use). As a result, some DVI previewers will simply show the blank space \TeX\ has reserved for your figure, not the figure itself.

There are two rather good documents on CTAN addressing figure production, with rather different emphasis. Keith Reckdahl’s, info/epslatex.pdf (also available in PostScript format as info/epslatex.ps), covers the standard \LaTeX\ facilities, as well as some of the supporting packages, notably \texttt{subfigure} (macros/latex/contrib/supported/subfigure) and \texttt{psfrag} (macros/latex/contrib/supported/psfrag). Anil K. Goel’s, info/figsinltx.ps, covers the different ways in which you might generate figures, and the old (\LaTeX\2.09) ways of including them into documents.

Part M Special sorts of typesetting

71 Drawing with \TeX

There are many packages to do pictures in \LaTeX\ itself (rather than importing graphics created externally), ranging from simple use of \LaTeX\ \texttt{picture} environment, through enhancements like \texttt{epic}, to sophisticated (but slow) drawing with \texttt{PCTeX}. Depending on your type of drawing, and setup, four systems should be at the top of your list to look at:
1. **graphics/pstricks**: this gives you access to all the power of PostScript from \TeX itself, by sophisticated use of \texttt{\special}s. You need a decent DVI to PostScript driver (like \texttt{dvips}), but the results are worth it. The well-documented package gives you not only low-level drawing commands (and full colour) like lines, circles, shapes at arbitrary coordinates, but also high-level macros for framing text, drawing trees and matrices, 3D effects, and more.

2. MetaPost; you liked \texttt{METAFONT}, but never got to grips with font files? Try MetaPost (question 4) — all the power of \texttt{METAFONT}, but it generates PostScript figures. Knuth uses it for all his work...

3. \texttt{Mfpic}; you liked \texttt{METAFONT}, but can’t understand the language? The package (\texttt{graphics/mfpic}) makes up \texttt{METAFONT} code for you within using familiar-looking \TeX macros. Not quite the full power of \texttt{METAFONT}, but a friendlier interface.

4. You liked \texttt{PICTEX} but don’t have enough memory or time? Look at Eitan Gurari’s \texttt{macro/generic/dratex}, which is as powerful as most other \TeX drawing packages, but is an entirely new implementation, which is not as hard on memory, is much more readable (and is fully documented).

### 72 Double-spaced documents in \LaTeX

Are you producing a thesis, and trying to obey regulations that were drafted in the typewriter era? Or are you producing copy for a journal that insists on double spacing for the submitted articles?

\LaTeX is a typesetting system, so the appropriate design conventions are for “real books”. If your requirement is from thesis regulations, find whoever is responsible for the regulations, and try to get the wording changed to cater for typeset theses (\emph{e.g.}, to say “if using a typesetting system, aim to make your thesis look like a well-designed book”). (If your requirement is from a journal, you’re probably even less likely to be able to get the rules changed, of course.)

If you fail to convince your officials, or want some inter-line space for copy-editing:

- Try changing \texttt{\baselinestretch}:

  \begin{verbatim}
  \renewcommand{\baselinestretch}{1.2}
  \end{verbatim}

  may be enough to give officials the impression you’ve kept to their regulations. Don’t try changing \texttt{\baselineskip}: its value is reset at any size-changing command.

- Alternatively, use a line-spacing package. The best for current is \texttt{macros/latex/contrib/supported/setspace/setspace.sty}.

### 73 Formatting a thesis in \LaTeX

Thesis styles are usually very specific to your University, so it’s usually not profitable to ask around for a package outside your own University. Since many Universities (in their eccentric way) still require double-spacing, you may care to refer to question 72. If you want to write your own, a good place to start is the University of California style (available as \texttt{macros/latex/}
contrib/supported/ucthesis), but it’s not worth going to a lot of trouble. (If officials won’t allow standard typographic conventions, you won’t be able to produce an aesthetically pleasing document anyway!)

74 Flowing text around figures in $\TeX$

There are several $\LaTeX$ packages that purport to do this, but they all have their limitations because the $\TeX$ machine isn’t really designed to solve this sort of problem. Piet van Oostrum has conducted a survey of the available packages; he recommends:

picins $\text{picins.sty}$ is part of a large package (systems/msdos/picins/picins.zip) that allows inclusion of pictures (e.g., with shadow boxes, various MS-DOS formats, etc.). The command is:

\begin{quote}
\verb|\parpic(width,height)(x-off,y-off)[Options][Position]{Picture}|
\end{quote}

Paragraph text

All parameters except the $\text{Picture}$ are optional. The picture can be positioned left or right, boxed with a rectangle, oval, shadowbox, dashed box, and a caption can be given which will be included in the list of figures.

Unfortunately (for those of us whose understanding of German is not good), the documentation is in German. Piet van Oostrum has written an English summary (systems/msdos/picins/picins.zip-summary).

floatflt $\text{macros/latex/contrib/other/floatflt}$ is an improved version (for $\LaTeX$ 2ε) of $\text{floatfig.sty}$, and its syntax is:

\begin{quote}
\verb|\begin{floatingfigure}[options]{width of figure}|
figure contents
\verb|\end{floatingfigure}|
\end{quote}

There is a (more or less similar) $\text{floatingtable}$ environment.

The tables or figures can be set left or right, or alternating on even/odd pages in a double-sided document.

The package works with the $\text{multicol}$ package, but doesn’t work well in the neighbourhood of list environments (unless you change your $\LaTeX$ document).

wrapfig $\text{macros/latex/contrib/other/misc/wrapfig.sty}$ has syntax:

\begin{quote}
\verb|\begin{wrapfigure}[height of figure in lines]{l,r,etc}{overhang}{width}|
figure, caption, etc.
\verb|\end{wrapfigure}|
\end{quote}

The syntax of the $\text{wrappable}$ environment is similar.

Height can be omitted, in which case it will be calculated by the package; the package will use the greater of the specified and the actual width. The {l,r,\textit{etc.}} parameter can also be specified as \texttt{i(nsides)} or \texttt{o(utsides)} for two-sided documents, and uppercase can be used to indicate that the picture should float. The overhang allows the figure to be moved into the margin. The figure or table will entered into the list of figures or tables if you use the \texttt{\caption} command.

The environments do not work within list environments that end before the figure or table has finished, but can be used in a parbox or minipage, and in twocolumn format.
75 Alternative head- and footlines in \LaTeX

The standard \LaTeX document classes define a small set of ‘page styles’ which (in effect) specify head- and footlines for your document. The set defined is very restricted, but \LaTeX is capable of much more; people occasionally set about employing \LaTeX facilities to do the job, but that’s quite unnecessary — Piet van Oostrum has already done the work.

The package is found in directory macros/latex/contrib/supported/fancyhdr and provides simple mechanisms for defining pretty much every head- or footline variation you could want; the directory also contains some (rather good) documentation and one or two smaller packages. Fancyhdr also deals with the tedious behaviour of the standard styles with initial pages (question 111), by enabling you to define different page styles for initial and for body pages.

76 Including a file in verbatim in \LaTeX

A good way is to use Rainer Schöpf’s verbatim.sty, which provides a command \verbatiminput that takes a file name as argument. This package is available as part of macros/latex/required/tools

Another way is to use the alltt environment, which requires alltt.sty (which is now part of \LaTeX). alltt interprets its contents ‘mostly’ verbatim, but executes any \TeX commands it finds: so one can say:

\begin{alltt}
\input{verb.txt}
\end{alltt}

of course, this is little use for inputting (\LaTeX source code...

The moreverb package (macros/latex/contrib/supported/moreverb) extends the facilities of verbatim package), providing a listing environment and a \listinginput command, which line-number the text of the file.

The fancyvrb package (macros/latex/contrib/supported/fancyvrb) offers configurable implementations of everything verbatim and moreverb have, and more besides. It is nowadays the package of choice for the discerning typesetter of verbatim text, but its wealth of facilities makes it a complex beast and study of the documentation is strongly advised.

77 Including line numbers in typeset output

For general numbering of lines, there are two packages for use with \LaTeX, macros/latex/contrib/supported/lineno (which permits labels attached to individual lines of typeset output) and macros/latex/contrib/supported/numline/numline.sty.

Both of these packages play fast and loose with the \LaTeX output routine, which can cause problems: the user should beware...

If the requirement is for numbering verbatim text, the macros/latex/contrib/supported/moreverb or macros/latex/contrib/supported/fancyvrb packages (see question 76) may be used.

One common use of line numbers is in critical editions of texts, and for this the edmac package
(macros/plain/contrib/edmac) offers comprehensive support.

78 Generating an index in \LaTeX

Making an index is not trivial; what to index, and how to index it, is difficult to decide, and uniform implementation is difficult to achieve. You will need to mark all items to be indexed in your text (typically with \index commands).

It is not practical to sort a large index within \TeX, so a post-processing program is used to sort the output of one \TeX run, to be included into the document at the next run.

The following programs are available:

- **makeindex** for \LaTeX under Unix (but runs under other OSs without changes). Available in indexing/makeindex; a version for the Macintosh is available as systems/mac/macmakeindex.sit, and one for MS-DOS is part of the em\TeX distribution (the em\TeX version also runs under OS/2).

  The Makeindex documentation is a good source of information on how to create your own index. Makeindex can be used with some \TeX macro packages other than \LaTeX, such as Eplain (question 12), and \TeXsis, nonfree/macros/texsis (whose macros, nonfree/macros/texsis-index, can be used independently with plain).

- **idxtex** for \LaTeX under VMS. Available (together with a glossary-maker called glotex) in indexing/glo+idxtex

- **texindex** A witty little shell/sed-script-based utility for \LaTeX under Unix. Available from support/texindex

  There are other programs called texindex, notably one that comes with the Texinfo distribution (question 14).

- **xindy** a recent development, designed for wide-ranging flexibility (including support for multilingual indexes), based on Common Lisp. The system is available on CTAN (support/xindy), but is more easily accessed from a web browser via http://www.itl.informatik.th-darmstadt.de/xindy/ since the distribution contains several different implementations.

79 Typesetting URLs

URLs tend to be very long, and contain characters that would naturally prevent them being hyphenated even if they weren’t typically set in \ttfamily, verbatim. Therefore, without special treatment, they often produce wildly overfull \hboxes, and their typeset representation is awful.

There are two approaches to this problem:

- **macros/latex/contrib/other/misc/path.sty**, which defines a \path command. The command defines each potential break character as a \discretionary, and offers the user the opportunity of specifying a personal list of potential break characters. Its chief disadvantage is fragility in the \LaTeX context.
• `macros/latex/contrib/other/misc/url.sty`, which defines an \url command (among others, including its own \path command). The command gives each potential break character a maths-mode ‘personality’, and then sets the URL itself in the user’s choice of font, in maths mode. It can produce (\LaTeX-style) ‘robust’ commands (see question 123) for use within moving arguments. Note that, because the operation is conducted in maths mode, spaces within the URL argument are ignored unless special steps are taken.

The author of this answer prefers the (rather newer) url.sty; both packages work equally well with plain \TeX (though of course, the fancy \LaTeX facilities of url.sty don’t have much place there).

80 Citing URLs with \BibTeX

There is no citation type for URLs, per se, in the standard \BibTeX styles, though Oren Patashnik (the author of \BibTeX) is considering developing one such for use with the long-awaited \BibTeX version 1.0.

The actual information that need be available in a citation of an URL is discussed at some length in the publicly available on-line extracts of ISO 690–2, available via http://www.nlc-bnc.ca/iso/tc46sc9/standard/690-2e.htm; the techniques below do not satisfy all the requirements of ISO 690–2, but they offer a solution that is at least available to users of today’s tools.

Until the new version arrives, the simplest technique is to use the howpublished field of the standard styles’ @misc function. Of course, the strictures about typesetting URLs (question 79) still apply, so the entry will look like:

```
@misc{...
...
  howpublished = "\url{http://...}"
}
```

Another possibility is that some conventionally-published paper, technical report (or even book) is also available on the Web. In such cases, a useful technique is something like:

```
@techreport{...
...
  note = "Also available as \url{http://...}"
}
```

There is good reason to use `macros/latex/contrib/other/misc/url.sty` in this context, since (by default) it ignores spaces in its argument. \BibTeX has a habit of splitting lines it considers excessively long, and if there are no space characters for it to use as ‘natural’ breakpoints, \BibTeX will insert a comment (‘%’) character … which is an acceptable character in an URL, so that \url will typeset it. The way around the problem is to insert odd spaces inside the URL itself in the `.bib` file, to enable \BibTeX to make reasonable decisions about breaking the line.

81 Using \BibTeX with plain \TeX

The file `macros/eplain/btxmac.tex` contains macros and documentation for using \BibTeX with plain \TeX, either directly or with Eplain (question 12). See question 26 for more infor-
mation about Bib\TeX itself.

82 Typesetting music in \TeX

A powerful package which allows the typesetting of polyphonic and other multiple-stave music is Music\TeX, written by Daniel Taupin (\texttt{taupin@rsovax.lps.u-psud.fr}). It is available in \texttt{macros/music\tex}

In the recent past, Daniel (as well as with various other people, notably Ross Mitchell and Andreas Egler) have been working on a development of Music\TeX, known as Musi\TeX. Musi\TeX is a three-pass system (with a processor program that computes values for the element spacing in the music), and achieves finer control than is possible in the unmodified \TeX-based mechanism that Music\TeX uses. Daniel Taupin and Andreas Egler are pursuing distinct versions of Musi\TeX they are available, respectively, from \texttt{macros/musixtex/taupin} and \texttt{macros/musixtex/egler}

Digital music fans can typeset notation for their efforts by using \texttt{midi2\tex}, which translates MIDI data files into Music\TeX \texttt{source code}. It is available from \texttt{support/midi2\tex}

A rather simpler notation than Music\TeX is supported by \texttt{abc2\tex}; this is a package designed to notate tunes stored in an ASCII format (\texttt{abc} notation). It was designed primarily for folk and traditional tunes of Western European origin (such as Irish, English and Scottish) which can be written on one stave in standard classical notation. However, it should be extendable to many other types of music. It is available from \texttt{support/abc2\tex}

There is a mailing list for discussion of typesetting music in \TeX. To subscribe, send mail to \texttt{mutex-request@stolaf.edu} containing the word \texttt{subscribe} in the body.

83 Drawing Feynman diagrams in \LaTeX

Michael Levine’s macro package for drawing Feynman diagrams in \LaTeX is available in \texttt{macros/latex209/contrib/feynman}

Another possibility is Thorsten Ohl’s \texttt{macros/latex/contrib/supported/feynmp}, that works in combination with METAFONT (or MetaPost). The \texttt{feynmp} or \texttt{feynmf} package reads a description of the diagram written in \LaTeX, and writes out code. METAFONT (or MetaPost) can then produce a font (or PostScript file) for use in a subsequent \LaTeX run. For new users, who have access to MetaPost, the PostScript version is probably the better route, for document portability and other reasons.

Part N How do I do $X$ in \TeX or \LaTeX

84 Proof environment

It is not possible to make a \texttt{proof} environment which automatically includes an ‘end-of-proof’ symbol. Some proofs end in displayed maths; others do not. If the input file contains \texttt{\begin{proof} ... \end{proof}} then \LaTeX finishes off the displayed maths and gets ready for a new line before it reads any instructions connected with ending the proof. But traditionally the end-of-proof
sign goes in the display, not on a new line. So you just have to put it in by hand in every proof.

85 Symbols for the number sets

It is a good idea to have commands such as \( \mathbb{R} \) for the real numbers and other standard number sets. Traditionally these were typeset in bold. Because mathematicians usually do not have access to bold chalk, they invented the special symbols that are now often used for \( \mathbb{R}, \mathbb{C}, \text{ etc} \). These symbols are known as “blackboard bold”. Before insisting on using them, consider whether going back to the old system of ordinary bold might not be acceptable (it is certainly simpler).

A set of blackboard bold capitals is available in the AMS fonts “msam” (e.g., “msam10” for 10pt) and “msbm”. The fonts have a large number of mathematical symbols to supplement the ones in the standard \( \mathrm{\LaTeX} \) distribution. The fonts are available in \texttt{fonts/ams/amsfonts-symbols}

Two files which load the fonts and define the symbols are provided, and both work with either \( \mathrm{\LaTeX} \) or \( \mathrm{\LaTeXe} \). Questions or suggestions regarding these fonts should be directed to tech-support@math.ams.org.

Another complete set of blackboard bold fonts, the bbold family, is available in \texttt{METAFONT} (in \texttt{fonts/bbold}). This set has the interesting property of offering blackboard bold forms of lower-case letters, something rather rarely seen on actual blackboards.

The “lazy person’s” blackboard bold macros:

\begin{verbatim}
\newcommand{\R}{\{\sf R\hspace*{-0.9ex}\%
\rule{0.15ex}{1.5ex}\hspace*{0.9ex}}}
\newcommand{\N}{\{\sf N\hspace*{-1.0ex}\%
\rule{0.15ex}{1.3ex}\hspace*{1.0ex}}}
\newcommand{\Q}{\{\sf Q\hspace*{-1.1ex}\%
\rule{0.15ex}{1.5ex}\hspace*{1.1ex}}}
\newcommand{\C}{\{\sf C\hspace*{-0.9ex}\%
\rule{0.15ex}{1.3ex}\hspace*{0.9ex}}}
\end{verbatim}

work well at normal size if the surrounding text is \texttt{cmr10}. However, they are not part of a proper maths font, and so do not work in sub- and superscripts. Moreover, the size and position of the vertical bar is affected by the font of the surrounding text.

86 Roman theorems

If you want to take advantage of the powerful \texttt{\newtheorem} command without the constraint that the contents of the theorem is in a sloped font (for example, to use it to create remarks, examples, proofs, ...) then you can use the style file \texttt{theorem.sty} (part of \texttt{macros/latex/requ\texttt{ired/tools}}). Alternatively, the following sets up an environment \texttt{remark} whose content is in roman.

\begin{verbatim}
\newtheorem{preremark}{Remark}
\newenvironment{remark}\
\{\begin{preremark}\rm\{\end{preremark}}
\end{verbatim}
This will not work if you are using the prototype NFSS outside of \LaTeX\ (question 131), because the command `\rm` behaves differently there.

87 Fancy enumeration lists

Suppose you want your top-level \texttt{enumerate}s to be labelled ‘I/’, ‘II/’, ..., then give these commands:

\begin{verbatim}
\renewcommand{\theenumi}{\Roman{enumi}}
\renewcommand{\labelenumi}{\theenumi/}
\end{verbatim}

The possible styles of numbering are given in Section 6.3 of Lamport’s book (see question 20). Both \texttt{theenumi} and \texttt{labelenumi} must be changed, since \texttt{theenumi} is used in cross-references to the list.

For lower level \texttt{enumerate}s, replace \texttt{enumi} by \texttt{enumii}, \texttt{enumiii} or \texttt{enumiv}, according to the level. If your label is much larger than the default, you should also change \texttt{leftmargini}, \texttt{leftmarginii}, \texttt{etc}.

If you’re running \LaTeX\, the package \texttt{enumerate.sty} (part of \texttt{macros/latex/required/tools}) offers similar facilities. With \texttt{enumerate.sty}, the example above would be achieved simply by starting the enumeration `\begin{enumerate}[I/]`.

88 Unnumbered sections in the Table of Contents

The easiest way to get headings of funny ‘sections’ such as prefaces in the table of contents is to use the counter \texttt{secnumdepth} described in Appendix C of the \LaTeX\ manual. For example:

\begin{verbatim}
\setcounter{secnumdepth}{-1}
\chapter{Preface}
\end{verbatim}

Of course, you have to set \texttt{secnumdepth} back to its usual value (which is 2 in the standard styles) before you do any ‘section’ which you want to be numbered.

Similar settings are made automatically in the \LaTeX\ book class by the `\frontmatter` and `\backmatter` commands.

This is why it works. \texttt{\chapter} without the star does

1. put something in the `.toc` file;
2. if \texttt{secnumdepth} $\geq 0$, increase the counter for the chapter and write it out.
3. write the chapter title.

Other sectioning commands are similar, but with other values used in the test.

89 Footnotes in tables

The standard \LaTeX\ `\footnote` command doesn’t work in tables; the table traps the footnotes and they can’t escape to the bottom of the page.
If your table is floating, your best bet is (unfortunately) to put the table in a \texttt{minipage}

environment and to put the notes underneath the table, or to use Donald Arseneau’s package \texttt{macros/latex/contrib/other/misc/threeparttable.sty}

Otherwise, if your table is not floating (it’s just a ‘\texttt{tabular}’ in the middle of some text), there are several things you can do to fix the problem.

1. Use \texttt{\footnotemark} to position the little marker appropriately, and then put in \texttt{\footnotetext} commands to fill in the text once you’ve closed the tabular environment.
   This is described in Lamport’s book, but it gets messy if there’s more than one footnote.

2. Stick the table in a \texttt{minipage} anyway. This provides all the ugliness of footnotes in a minipage with no extra effort.

3. Use \texttt{macros/latex/contrib/other/misc/threeparttable.sty} anyway; the package is intended for floating tables, and the result might look odd if the table is not floating, but it will be reasonable.

4. Use \texttt{tabularx} or \texttt{longtable} from the \LaTeX{} tools distribution (\texttt{macros/latex/required/tools}); they’re noticeably less efficient than the standard \texttt{tabular} environment, but they do allow footnotes.

5. Grab hold of \texttt{footnote.sty} from CTAN, lurking in \texttt{macros/latex/contrib/supported/mdwtools}
   Then put your tabular environment inside a \texttt{savenotes} environment. Alternatively, say \texttt{\makesavenoteenv{tabular}} in the preamble of your document, and tables will all handle footnotes correctly.

6. Use \texttt{mdwtab.sty} from the same directory (\texttt{macros/latex/contrib/supported/mdwtools}).
   This will handle footnotes properly, and has other facilities to increase the beauty of your tables. It may also cause other table-related packages (not the standard ‘tools’ ones, though) to become very unhappy and stop working.

**90 Style of section headings**

Suppose that the editor of your favourite journal has specified that section headings must be centred, in small capitals, and subsection headings ragged right in italic, but that you don’t want to get involved in the sort of programming described in \textit{The \LaTeX{} Companion} (question 20; the programming itself is discussed in question 117). The following hack will probably satisfy your editor. Define yourself new commands

\begin{verbatim}
\newcommand{\ssection}[1]{\% \section[#1]{\centering\sc #1}}
\newcommand{\ssubsection}[1]{\% \subsection[#1]{\raggedright\it #1}}
\end{verbatim}

and then use \texttt{\section} and \texttt{\subsection} in place of \texttt{\section} and \texttt{\subsection}. This isn’t perfect: section numbers remain in bold, and starred forms need a separate redefinition. Also, this will not work if you are using the prototype NFSS with \texttt{\LaTeX{} 2.09}, because the font-changing commands behave differently there.

56
The `macros/latex/contrib/supported/sectsty` package provides an easy-to-use set of tools to do this job, while the `macros/latex/contrib/supported/titlesec` package permits more advanced usage as well. (`titlesec` comes with a second package, `titletoc`, which is used to adjust the format of table of contents entries.)

The headings of `\chapter` and `\part` commands are produced by a different mechanism; `sectsty` deals with both, but `titlesec` deals only with `\chapter`. The `macros/latex/contrib/supported/fncychap` package provides a nice collection of customised chapter heading designs.

### 91 Indent after section headings

\LaTeX{} implements a style that doesn’t indent the first paragraph after a section heading. There are coherent reasons for this, but not everyone likes it. The package `indentfirst.sty` (part of `macros/latex/required/tools`) suppresses the mechanism, so that the first paragraph is indented.

### 92 Footnotes in \LaTeX{} section headings

The `\footnote` command is fragile, so that simply placing the command in `\section`’s arguments isn’t satisfactory. Using `\protect\footnote` isn’t a good idea either: the arguments of a section command are used in the table of contents and (more dangerously) potentially also in page headings. Unfortunately, there’s no mechanism to suppress the footnote in the heading while allowing it in the table of contents, though having footnotes in the table of contents is probably unsatisfactory anyway.

To suppress the footnote in headings and table of contents:

- Take advantage of the fact that the mandatory argument doesn’t ‘move’ if the optional argument is present: `\section[title]{title}\footnote{title ftnt}`
- Use the `macros/latex/contrib/supported/footmisc` package, with package option `stable` —this modifies footnotes so that they softly and silently vanish away if used in a moving argument.

### 93 Changing the margins in \LaTeX{}

Don’t do it. Learn some \LaTeX{}, produce some documents, and then ask again.

You can never change the margins of a document by software, because they depend on the actual size of the paper. What you can change are the distances from the apparent top and left edges of the paper, and the width and height of the text. Changing the last two requires more skill than you might expect. The height should bear a certain relationship to `\baselineskip`. And the width should not be more than 75 characters. Lamport’s warning in his section on ‘Customizing the Style’ really must be taken seriously. One-inch margins on A4 paper are fine for 10- or 12-pitch typewriters, but not for 10pt type (or even 11pt or 12pt) because so many characters per line will irritate the reader. However...
Perhaps the easiest way to get more out of a page in \LaTeX is to get \texttt{macros/latex209/contrib/misc/fullpage.sty} which sets the margins of the page identical to those of plain \TeX, \textit{i.e.}, 1-inch margins at all four sides of the paper. It also contains an adjustment for A4 paper.

Somewhat more flexible is \texttt{macros/latex/contrib/other/misc/vmargin.sty}, which has a canned set of paper sizes (a superset of that provided in \LaTeX 2\epsilon), provision for custom paper, margin adjustments and provision for two-sided printing.

For details of \LaTeX\!'s page parameters, see section C.5.3 of the \LaTeX manual (pp. 181–182). The origin in DVI coordinates is one inch from the top of the paper and one inch from the left side; positive horizontal measurements extend right across the page, and positive vertical measurements extend down the page. Thus, for margins closer to the left and top edges of the page than 1 inch, the corresponding parameters, \textit{i.e.}, \texttt{\textbackslash evenpagemargin, \textbackslash oddpagemargin, \textbackslash topmargin}, can be set to negative values.

You cannot simply change the margins of part of a document within the document by modifying the parameters shown in Lamport’s figure C.3. They should only be changed in the preamble of the document, \textit{i.e.}, before the \texttt{\textbackslash begin\{document\}} statement. To adjust the margins within a document we define an environment:

\begin{verbatim}
\newenvironment{changemargin}[2]{}{
\begin{list}{}{
\setlength{\topsep}{0pt}
\setlength{\leftmargin}{#1}
\setlength{\rightmargin}{#2}
\setlength{\listparindent}{\parindent}
\setlength{\itemindent}{\parindent}
\setlength{\parsep}{\parskip}
}\item[]\end{list}}
\end{verbatim}

This environment takes two arguments, and will indent the left and right margins by their values, respectively. Negative values will cause the margins to be narrowed, so \texttt{\begin{changemargin}{-1cm}{-1cm}} narrows the left and right margins by 1cm.

\section*{94 Finding the width of a letter, word, or phrase}

Put the word in a box, and measure the width of the box. For example,

\begin{verbatim}
\newdimen\stringwidth
\letbox0=\hbox{hi}
\stringwidth=\wd0
\end{verbatim}

Note that if the quantity in the \texttt{\hbox} is a phrase, the actual measurement only approximates the width that the phrase will occupy in running text, since the inter-word glue can be adjusted in paragraph mode.

The same sort of thing is expressed in \LaTeX by:

\begin{verbatim}
\newlength{\gnat}
\settowidth{\gnat}{\textbf{small}}
\end{verbatim}

This sets the value of the length command \texttt{\gnat} to the width of “small” in bold-face text.

58
95 Changing the space between letters

A common technique in advertising copy (and other text whose actual content need not actually be read) is to alter the space between the letters (otherwise known as the tracking). As a general rule, this is a very bad idea: it detracts from legibility, which is contrary to the principles of typesetting (any respectable font you might be using should already have optimum tracking built into it).

The great type designer, Eric Gill, is (possibly apocryphally) credited with saying “he who would letterspace lower-case text, would steal sheep” (stealing sheep was, before Gill’s time, a capital offence in Britain). As the remark suggests, though, letterspacing of upper-case text is less awful a crime; the technique used also to be used for emphasis of text set in gothic (or similar) fonts.

Straightforward macros (usable, in principle, with any \TeX{} macro package) may be found in \texttt{macros/generic/letterspacing.tex}

A more comprehensive package is \texttt{macros/latex/contrib/supported/soul} (which is optimised for use with \LaTeX{}, but also works with Plain \TeX{}. Soul also permits hyphenation of letterspaced text; Gill’s view of such an activity is not (even apocryphally) recorded.

96 Excluding blocks of text from the DVI file

Rainer Schöpf’s \texttt{verbatim.sty} provides a comment environment which excludes everything between \texttt{\begin{comment}} and \texttt{\end{comment}}. This package is available as part of \texttt{macros/latex/required/tools}

A more general environment for doing the job is Victor Eijkhout’s \texttt{comment.sty}, which lets you define environments for inclusion or exclusion in a document, thus offering a primitive configuration structure. It is available from the CTAN sites in \texttt{macros/latex209/contrib/misc/comment.sty}

97 Setting bold Greek letters in \LaTeX{}

The simple solution (\texttt{\mathbf}) doesn’t always work, because lower-case Greek letters behave differently from upper-case Greek letters (due to Knuth’s esoteric font encoding decisions). However, \texttt{\mathbf} can be used for upper-case Greek letters in ordinary circumstances, but the \texttt{\AMS-L\TeX} package \texttt{amsmath.sty} disables this font-switching and you must use one of the techniques outlined below.

The plain \TeX{} solution does work, in a limited way:

\begin{verbatim}
{\boldmath$\theta$}
\end{verbatim}

but \texttt{\boldmath} may not be used in maths mode, so this ‘solution’ requires arcana such as:

\begin{verbatim}
$\ldots \mbox{\boldmath$\theta$} \ldots$
\end{verbatim}

which then causes problems in superscripts, etc.

These problems may be addressed by using a bold mathematics package.

- The package \texttt{bm.sty}, which is part of the \LaTeX{} tools distribution (\texttt{macros/latex/}}
required/tools), defines a command \bm which may be used anywhere in maths mode.

- The package amsbsy.sty, which is part of \texttt{AMS-\LaTeX} (macros/latex/required/amslatex) defines a command \texttt{\boldsymbol}, which (though slightly less comprehensive than \texttt{\bm}) covers almost all common cases.

All these solutions cover all mathematical symbols, not merely Greek letters.

98 Defining a new log-like function in \LaTeX

Use the \texttt{\mathop} command, as in:

\begin{verbatim}
\newcommand{\diag}{\mathop{\mathrm{diag}}}
\end{verbatim}

Subscripts and superscripts on \texttt{\diag} will be placed exactly as they are on \texttt{\lim}. If you want your subscripts and superscripts always placed to the right, do:

\begin{verbatim}
\newcommand{\diag}{\mathop{\mathrm{diag}}\nolimits}
\end{verbatim}

This works in \LaTeX\ 2.09 and in \LaTeX\ 2ε, but not under NFSS alone (see question 113). However, the canonical method for doing this in \LaTeX\ 2ε is to use the \texttt{\DeclareMathOperator} command of \texttt{amsopn.sty} (which is part of the \texttt{AMS-L\LaTeX} package: macros/latex/required/amslatex).

(It should be noted that “log-like” was reportedly a joke on Lamport’s part; it is of course clear what was meant.)

99 Typesetting all those \TeX-related logos

Knuth was making a particular point about the capabilities of \TeX when he defined the logo. Unfortunately, many believe, he thereby opened floodgates to give the world a whole range of rather silly ‘bumpy road’ logos such as \texttt{AMS-\TeX}, \texttt{Pl\TeX}, \texttt{Bib\TeX}, and so on, produced in a flurry of different fonts, sizes, and baselines — indeed, everything one might hope to cause them to obstruct the reading process. In particular, Lamport invented \LaTeX\ (silly enough in itself) and marketing input from Addison-Wesley led to the even stranger current logo \LaTeX\ 2ε.

Sensible users don’t have to follow this stuff wherever it goes, but, for those who insist, a large collection of logos is defined in \texttt{macros/eplain/texnames.sty} (but note that this set of macros isn’t entirely reliable in \LaTeX\ 2ε). The \texttt{META\FONT} and MetaPost logos can be set in fonts that \LaTeX\ 2ε knows about (so that they scale with the surrounding text) using the package \texttt{macros/latex/contrib/supported/mflogo}; but be aware that booby-traps surround the use of the Knuthian font for MetaPost (you might get "META O T" in the font used to set \texttt{META\FONT}). You needn’t despair, however — the author himself uses just ‘MetaPost’. For those who don’t wish to acquire the ‘proper’ logos, the canonical thing to do is to say \texttt{AMS-\TeX} (\texttt{AMS-\TeX}) for \texttt{AMS-\TeX}, \texttt{Pic\TeX} (\texttt{Pic\TeX}) for \texttt{Pic\TeX}, \texttt{Bib\TeX} (\texttt{Bib\TeX}) for \texttt{Bib\TeX}, and so on.

While the author of this FAQ list can’t quite bring himself to do away with the bumpy-road logos herein, he regularly advises everyone else to…

\footnote{Re-worded for the GELLMU edition}
100 1-column abstract in 2-column document

One often requires that the abstract of a paper should appear across the entire page, even in a two-column paper. The required trick is:

\documentclass[twocolumn]{article}
...
\begin{document}
... % \author, etc
\twocolumn[
  \begin{abstract}
  ...
  \end{abstract}
  \end{twocolumnfalse}
\]

Unfortunately, with the above \thanks won’t work in the \author list. If you need such specially-numbered footnotes, you can make them like this:

\title{Demonstration}
\author{Me, You\thanks{}}
\twocolumn[
  ... as above ... 
]
{
  \renewcommand{\thefootnote}{\fnsymbol{footnote}}
  \footnotetext[1]{Thanks for nothing}
}

and so on.

101 Changing babel’s ideas of words to use

\LaTeX uses symbolic names for many of the automatically-generated text it produces (special-purpose section headers, captions, etc.). For example, the special section produced by the \tableofcontents command is always called \contentsname (or rather, what \contentsname is set to).

When babel is selecting a new language, it ensures that these symbolic names are translated appropriately for the language in question. In particular, babel selects the document’s main language at the point that \begin{document} is executed, which means that any changes to these symbolic names made in the prologue of a document that uses babel will be lost.

Therefore, babel defines a command to enable users to change the definitions of the symbolic names, on a per-language basis: \addto\captions{language} is the thing.

For example:

\addto\captionsdanish{\%
102 Code listings in $\LaTeX$

‘Pretty’ code listings are sometimes considered worthwhile by the neurotically aesthetic programmer, but they have a serious place in the typesetting of dissertations by computer science and other students who are expected to write programs. Simple verbatim listings are commonly useful, as well.

Verbatim listings are dealt with elsewhere (see question 76). ‘Pretty’ listings are generally provided by means of a pre-compiler, but the listings package (macros/latex/contrib/supported/listings) manages to do the job within $\LaTeX$.

The $\text{lgrind}$ system (CTAN support/lgrind) is a well-established pre-compiler, with all the facilities one might need and a wide repertoire of languages.

The $\text{tiny\_c2l}$ system (CTAN support/tiny\_c2l) is more recent: users are encouraged to generate their own driver files for languages it doesn’t already deal with.

Part O Macros for Particular Types of Documents

103 Setting papers for journals

Publishers of journals have a wide range of requirements for the presentation of papers, and while many publishers do accept electronic submissions in $\LaTeX$, they don’t often submit recommended macros to public archives.

Nevertheless, there are considerable numbers of macros of one sort or another available on CTAN; searching for your journal name in the CTAN catalogue

http://www.tex.ac.uk/tex-archive/help/Catalogue/catalogue.html

may well turn up what you're seeking.

Failing that, you may be well advised to contact the prospective publisher of your paper; many publishers have macros on their own web sites, or otherwise available only upon application.

Check that the publisher is offering you macros suitable to an environment you can use: a few still have no macros for current $\LaTeX$, for example, claiming that $\LaTeX$ 2.09 is good enough…

Some publishers rekey anything sent them anyway, so that it doesn’t really matter what macros you use. Others merely encourage you to use as few extensions of a standard package as possible, so that they will find it easy to transform your paper to their own internal form.

104 A ‘report’ from lots of ‘article’s

This is a requirement, for example, if one is preparing the proceedings of a conference whose papers were submitted in $\LaTeX$. 


There's no canned solution, but Matt Swift's **includex** and **moredefs** packages (both part of the macros/latex/contrib/supported/frankenstein bundle) offer a possible way forward.

**includex** enables you to `\includedoc` complete articles (in the way that you `\include` chapter files in an ordinary report). It doesn't do the whole job for you, though. You need to analyse the package use of the individual papers, and ensure that a consistent set is loaded in the preamble of the main report.

Another tedious problem to address is that **\LaTeX** suppresses all title-related commands after `\maketitle` has been used once in a document: thus you need to define replacements for all of them, for use in the papers.

More work is plainly needed in this area, but at least a start has been made.

### 105 Curriculum Vitae (Resumé)

A framework class for *Curricula Vitae*, provided by Andrej Brodnik, is available from macros/latex/contrib/other/vita

The class can be customised both for subject (example class option files are offered for both computer scientists and singers), and for language (both the options provided are available for both English and Slovene). Extensions may be written by creating new class option files, or by using macros defined in the class to define new entry types, etc.

### Part P Things are Going Wrong...

#### 106 Weird hyphenation of words

You may have a version mismatch problem. **\TeX**’s hyphenation system changed between version 2.9 and 3.0. If you are using (**plain**) **\TeX** version 3.0 or later, make sure your plain.tex file has a version number which is at least 3.0. If you are using **\LaTeX** 2.09 you should consider upgrading to **\LaTeX** 2ε; if for some reason you can’t, the last version of **\LaTeX** 2.09, released on 25 March 1992, is available (for the time being at least) from obsolete/macros/latex209/distribs/latex209.tar and ought to solve this problem.

If you’re using **\LaTeX** 2ε, the problem probably arises from your hyphen.cfg file, which has to be created if you’re using a multi-lingual version.

For the curious, here’s what happened: before **\TeX** 3.0 the hyphenation algorithm would not break a word if the part before the break was not at least two characters long, and the part after the break at least three characters long. Starting with version 3.0 the parameters `\lefthyphenmin` and `\righthyphenmin` control the length of these fragments. These are set to 2 and 3, respectively, in the new plain and lplain formats. They can be set to any value, of course, but if `\lefthyphenmin+\righthyphenmin` is greater than 62, all hyphenation is suppressed.

A further source of oddity can derive from the 1995 release of Cork-encoded fonts (question 37), which introduced an alternative hyphen character. The **\LaTeX** 2ε configuration files in the font release specified use of the alternative hyphen, and this could produce odd effects with words containing an explicit hyphen. The font configuration files in the December 1995 release of
\TeX\,2e do not use the alternative hyphen character, thus removing this source of problems.

107  (Merely) peculiar hyphenation

You may have found that \TeX's famed automatic word-division does not produce the break-points recommended by your dictionary. This may be because \TeX is set up for American English, whose rules for word division (as specified, for example, in Webster's Dictionary) are completely different from the British ones (as specified, for example, in the Oxford Dictionaries). This problem is being addressed by the UK \TeX User community (see Baskerville, issue 4.4) but an entirely satisfactory solution will take time. An interim hyphenation file is available in language/hyphenation/ukhyphen.tex

108  Accented words aren’t hyphenated

\TeX’s algorithm for hyphenation gives up when it encounters an \texttt{\accent} command; there are good reasons for this, but it means that quality typesetting in non-English languages can be difficult.

For \TeX itself, avoiding this effect means using Cork-encoded fonts (question 37) which contain accented letters as single glyphs. In the future, perhaps, Omega (question 133) will provide a rather different solution.

109  Enlarging \TeX

The \TeX error message ‘capacity exceeded’ covers a multitude of problems. What has been exhausted is listed in brackets after the error message itself, as in:

\begin{verbatim}
! \TeX \ capacity \ exceeded, \ sorry
   .. \[ \text{main memory size=263001}. \]
\end{verbatim}

Most of the time this error can be fixed without enlarging \TeX. The most common causes are unmatched braces, extra-long lines, and poorly-written macros. Extra-long lines are often introduced when files are transferred incorrectly between operating systems, and line-endings are not preserved properly (the tell-tale sign of an extra-long line error is the complaint that the ‘buf_size’ has overflowed).

If you really need to extend your \TeX’s capacity, the proper method depends on your installation. There is no need (with modern \TeX implementations) to change the defaults in Knuth’s WEB source; but if you do need to do so, use a change file to modify the values set in module 11, recompile your \TeX and regenerate all format files.

Modern implementations allow the sizes of the various bits of \TeX’s memory to be changed semi-dynamically. Some (such as em\TeX) allow the memory parameters to be changed in command-line switches when \TeX is started; most frequently, a configuration file is read which specifies the size of the memory. On web2c-based systems, this file is called \texttt{texmf.cnf}: see the documentation that comes with the distribution for other implementations. Again, in many cases, the format files need to be regenerated after changing the memory parameters.
110 Moving tables and figures in \LaTeX

Tables and figures have a tendency to surprise, by floating away from where they were specified to appear. This is in fact perfectly ordinary document design: any professional typesetting package will float figures and tables to where they’ll fit without violating the certain typographic rules. Even if you use the placement specifier h for ‘here’, the figure or table will not be printed ‘here’ if doing so would break the rules: the rules themselves are pretty simple, and are given on page 198, section C.9 of the \LaTeX manual. In the worst case, \LaTeX’s rules can cause the floating items to pile up to the extent that you get an error message saying “Too many unprocessed floats”; this means that the limited set of registers in which \LaTeX stores floating items is full. What follows is a simple checklist of things to do to solve these problems (the checklist talks throughout about figures, but applies equally well to tables).

• Are the placement parameters on your figures right? The default (tbp) is reasonable; you should never simply say ‘h’, for example, since that says “if it can’t go here, it can’t go anywhere”, and as a result all subsequent floats pile up behind it.

• Can you perhaps prevent your figures from floating by adjusting \LaTeX’s placement parameters? Again, the defaults are reasonable, but can be overridden in case of problems. The parameters are described on pages 199–200, section C.9 of the \LaTeX manual.

• Are there places in your document where you could ‘naturally’ put a \clearpage command? If so, do: the backlog of floats is cleared after a \clearpage. (Note that the \chapter command implicitly executes \clearpage, so you can’t float past the end of a chapter.)

• Have a look at the \LaTeX 2ε afterpage package (part of macros/latex/required/tools). Its documentation gives as an example the idea of putting \clearpage after the current page (where it will clear the backlog, but not cause an ugly gap in your text), but also admits that the package is somewhat fragile (though it’s improving).

• As a last resort, try the package macros/latex209/contrib/misc/morefloats.sty; this ‘simply’ increases the number of floating inserts that \LaTeX can handle at one time (from 18 to 36), but that may suit your needs.

• If you actually wanted all your figures to float to the end (e.g., for submitting a draft copy of a paper), don’t rely on \LaTeX’s mechanism: get the package macros/latex/contrib/supported/endfloat to do the job for you.

111 \pagestyle{empty} on first page in \LaTeX

If you use \pagestyle{empty}, but the first page is numbered anyway, you are probably using the \maketitle command too. This is not a bug but a feature! The standard \LaTeX styles are written so that initial pages (pages containing a \maketitle, \part, or \chapter) have a different page style from the rest of the document; Hence, the commands internally issue \thispagestyle{plain}. This is usually not acceptable behaviour if the surrounding page style is ‘empty’.

Possible workarounds include:
• Put \thispagestyle{empty} immediately after the \maketitle command, with no blank line between them.

• Use fancyhdr.sty, which allows you to customise the style for initial pages independently of that for body pages. It is available in macros/latex/contrib/support/fancyhdr.

• Use nopageno.sty, which suppresses this behaviour. It is available in macros/latex/contrib/support/carlis/nopageno.sty.

112 Underlined text won’t break

Knuth made no provision for underlining text: he took the view that underlining is not a typesetting operation, but rather one that provides emphasis on typewriters, which typically offer but one typeface. The corresponding technique in typeset text is to switch from upright to italic text (or vice-versa): the \LaTeX command \texttt{\emph} does just that to its argument.

Nevertheless, typographically illiterate people (such as those that specify double-spaced thesis styles — question 72) continue to require underlining of us, so \LaTeX as distributed defines an \texttt{\underline} command that applies the mathematical ‘underbar’ operation to text. This technique is not entirely satisfactory, however: the text gets stuck into a box, and won’t break at line end.

The solution is the package macros/latex/contrib/other/misc/ulem.sty, which redefines the \texttt{\emph} command to underline its argument: the underlined text thus produced behaves as ordinary emphasised text, and will break over the end of a line. (The package is capable of other peculiar effects, too: read its documentation, contained within the file itself.)

113 Odd behaviour of \texttt{\rm}, \texttt{\bf}, etc.

If commands such as \texttt{\rm} and \texttt{\bf} have suddenly stopped working in \LaTeX in the way that you expect, it is likely that your system administrator has installed a version of \LaTeX 2.09 with a prototype version of the NFSS (question 131). Complain loudly; ask your system administrator to replace this version with \LaTeX 2ε (question 131), in which commands such as \texttt{\rm} and \texttt{\bf} work just as before if you are using one of the standard classes — article, report and book (among others). In the meantime, use the option oldlfont.sty, which should have been installed at the same time as NFSS.

114 Old \LaTeX font references such as \texttt{\tenrm}

\LaTeX 2.09 defined a large set of commands for access to the fonts that it had built in to itself. For example, various flavours of \texttt{cmr} could be found as \texttt{fivrm}, \texttt{sixrm}, \texttt{sevrm}, \texttt{egtrm}, \texttt{ninrm}, \texttt{tenrm}, \texttt{elvrm}, \texttt{twlrm}, \texttt{frtnrm}, \texttt{svtnrm}, \texttt{twtyrm} and \texttt{twfvrn}. These commands were never documented, but certain packages nevertheless used them to achieve effects they needed.

Since the commands weren’t public, they weren’t included in \LaTeX 2ε; to use the unconverted \LaTeX 2.09 packages under \LaTeX 2ε, you need also to include the package rawfonts.sty (which is part of the \LaTeX 2ε distribution).
115 Missing symbols

If some symbols, such as \Box and \lhd, no longer appear to exist, then your system administrator has probably upgraded your version of \LaTeX to either use the prototype NFSS or \LaTeX 2ε itself (question 131). In the former case, use oldfont.sty, as in the question 113. In the latter, use the package latexsym, which is part of the \LaTeX 2ε distribution, or the package amsfonts, if it is available.

116 \LaTeX gets cross-references wrong

Sometimes, however many times you run \LaTeX, the cross-references are just wrong. Remember that the \label command must come after the \caption command, or be part of it. For example,
\begin{verbatim}
\begin{figure} \begin{figure}
\caption{A Figure} \caption{A Figure% \label{fig} \label{fig}}
\end{figure} \end{figure}
\end{verbatim}

117 \@ and \@ in macro names

A common source of problems in a \LaTeX document is the diagnostic about the appearance of the command \@, or about other commands containing the character \@. The most common complaint is “You can’t use ‘\texttt{spacefactor}’ in vertical mode”, but others occur.

Such problems are usually caused by including a \LaTeX 2ε class or package file into a \LaTeX document by some means other than \documentclass or \usepackage. \LaTeX defines internal commands whose names contain the character \@; this enables it to avoid clashes between its internal names and names that we would normally use in our documents. In order that these commands may work at all, \documentclass and \usepackage play around with the meaning of \@. Solve this problem by using the correct command to include the file.

But, you will say, “The \LaTeX Companion tells me to use commands containing \@!”

Indeed; for example, there’s a lengthy section about \texttt{@startsection} and how to use it to control the appearance of section titles. Page 15 of The Companion explains this; and suggests that you make such changes in the document preamble, between \texttt{makeatletter} and \texttt{makeatother}.

So the definition of \texttt{subsection} on page 26 could be:
\begin{verbatim}
\makeatletter
\renewcommand{\subsection}{@startsection
{subsection}{\normalfont\normalsize\itshape}}% style
\makeatother
\end{verbatim}

118 Where are the msx and msy fonts?

The msx and msy fonts were designed by the American Mathematical Society in the very early days of \TeX, for use in typesetting papers for mathematical journals. They were designed using the ‘old’ \texttt{METAfont}, which wasn’t portable and is no longer available; for a long time they were
only available in 300dpi versions which only imperfectly matched modern printers. The AMS has now redesigned the fonts, using the current version of METAFONT, and the new versions are called the msa and msb families; they are available from fonts/ams/amsfonts-symbols

Nevertheless, msx and msy continue to turn up to plague us. There are, of course, still sites that haven’t got around to upgrading; but, even if everyone upgraded, there would still be the problem of old documents that specify them.

If you have a .tex source that requests msx and msy, the best technique is to edit it so that it requests msa and msb (you only need to change the single letter in the font names).

If you have a DVI file that requests the fonts, there is a package of virtual fonts (question 34) to map the old to the new series; it’s available in fonts/vf-files/msx2msa

119 Where are the am fonts?

One still occasionally comes across a request for the am series of fonts. The initials stood for ‘Almost [Computer] Modern’, and they were the predecessors of the Computer Modern fonts that we all know and love (or hate)\(^{10}\). There’s not a lot one can do with these fonts; they are (as their name implies) almost (but not quite) the same as the cm series; if you’re faced with a document that requests them, all you can reasonably do is to edit the document. The appearance of DVI files that request them is sufficiently rare that no-one has undertaken the mammoth task of creating a translation of them by means of virtual fonts; however, most drivers let you have a configuration file in which you can specify font substitutions. If you specify that every am font should be replaced by its corresponding cm font, the output should be almost correct.

120 ‘String too long’ in Bib\TeX\X

The Bib\TeX\X diagnostic “Warning–you’ve exceeded 1000, the global-string-size, for entry foo” usually arises from a very large abstract or annotation included in the database. The diagnostic usually arises because of an infelicity in the coding of abstract.bst, or styles derived from it. (One doesn’t ordinarily output annotations in other styles.)

The solution is to make a copy of the style file (or get a clean copy from CTAN — biblio/bibtex/contrib/abstract.bst), and rename it (e.g., on a long file-name system to abstract-long.bst). Now edit it: find function outputnonnull and

- change its first line (line 60 in the version on CTAN) from "{ ’s :=" to "{ swap$", and
- delete its last line, which just says "s" (line 84 in the version on CTAN).

Finally, change your \bibliographystyle command to refer to the name of the new file.

This technique applies equally to any bibliography style: the same change can be made to any similar outputnonnull function.

If you’re reluctant to make this sort of change, the only way forward is to take the entry out of the database, so that you don’t encounter Bib\TeX\X’s limit, but you may need to retain the

\(^{10}\) The fonts acquired their label ‘Almost’ following the realisation that their first implementation in METAFONT79 still wasn’t quite right; Knuth’s original intention had been that they were the final answer
entry because it will be included in the typeset document. In such cases, put the body of the entry in a separate file:

```latex
@article{long.boring,
  author = "Fred Verbose",
  ...
  abstract = "\{\input{abstracts/long.tex}}"
}
```

In this way, you arrange that all \TeX{} has to deal with is the file name, though it will tell \TeX{} (when appropriate) to include all the long text.

**Part Q  Why does it do that?**

121 **What’s going on in my `\include` commands?**

The original \LaTeX{} provided the `\include` command to address the problem of long documents: with the relatively slow computers of the time, the companion `\includeonly` facility was a boon. With the vast increase in computer speed, `\includeonly` is less valuable (though it still has its place in some very large projects). Nevertheless, the facility is retained in current \LaTeX{}, and causes some confusion to those who misunderstand it.

In order for `\includeonly` to work, `\include` makes a separate `.aux` file for each included file, and makes a ‘checkpoint’ of important parameters (such as page, figure, table and footnote numbers); as a direct result, it must clear the current page both before and after the `\include` command. What’s more, this mechanism doesn’t work if a `\include` command appears in a file that was `\include`d itself: \LaTeX{} diagnoses this as an error.

So, we can now answer the two commonest questions about `\include`:

- Why does \LaTeX{} throw a page before and after `\include` commands?
  Answer: because it has to. If you don’t like it, replace the `\include` command with `\input` — you won’t be able to use `\includeonly` any more, but you probably don’t need it anyway, so don’t worry.

- Why can’t I nest `\include`d files? — I always used to be able to under \LaTeX{} 2.09.
  Answer: in fact, you couldn’t, even under \LaTeX{} 2.09, but the failure wasn’t diagnosed. However, since you were happy with the behaviour under \LaTeX{} 2.09, replace the `\include` commands with `\input` commands (with `\clearpage` as appropriate).

122 **Why does it ignore paragraph parameters?**

When \TeX{} is laying out text, it doesn’t work from word to word, or from line to line; the smallest complete unit it formats is the paragraph. The paragraph is laid down in a buffer, as it appears, and isn’t touched further until the end-paragraph marker is processed. It’s at this point that the paragraph parameters have effect; and it’s because of this sequence that one often makes mistakes that lead to the paragraph parameters not doing what one would have hoped (or expected).
Consider the following sequence of \LaTeX:

\begin{verbatim}
{\raggedright % declaration for ragged text
Here's text to be ranged left in our output,
but it's the only such paragraph, so we now
end the group.}

Here's more that needn't be ragged...
\end{verbatim}

\TeX{} will open a group, and set the ragged-setting parameters within that group; it will then
save a couple of sentences of text and close the group (thus restoring the previous value of the
ragged-setting parameters). Then it encounters a blank line, which it knows to treat as a \texttt{\par}
token, so it typesets the two sentences; but because the enclosing group has now been closed,
the parameter settings have been lost, and the paragraph will be typeset normally.

The solution is simple: close the paragraph inside the group, so that the setting parameters
remain in place. An appropriate way of doing that is to replace the last three lines above
with:

\begin{verbatim}
end the group.\par
Here's more that needn't be ragged...
\end{verbatim}

In this way, the paragraph is completed while the setting parameters are still in force within
the enclosing group.

Another alternative is to define an environment that does the appropriate job for you. For the
above example, \LaTeX{} already defines an appropriate one:

\begin{verbatim}
\begin{flushleft}
Here's text to be ranged left...
\end{flushleft}
\end{verbatim}

123 What’s the reason for ‘protection’?

Sometimes \LaTeX{} saves data it will reread later. These data are often the argument of some
command; they are the so-called moving arguments. (‘Moving’ because data are moved around.)
Places to look for are all arguments that may go into table of contents, list of figures, \textit{etc.}; namely,
data that are written to an auxiliary file and read in later. Other places are those data that
might appear in head- or footlines. Section headers and figure captions are the most prominent
examples; there’s a complete list in Lamport’s book (see question 20).

What’s going on really, behind the scenes? The commands in the moving arguments are already
expanded to their internal structure during the process of saving. Sometimes this expansion
results in invalid \TeX{} code when processed again. “\protect\cmd” tells \LaTeX{} to save \cmd{} as
\texttt{\cmd}, without expansion.

What is a ‘fragile command’? It’s a command that expands into illegal \TeX{} code during the
save process.

What is a ‘robust command’? It’s a command that expands into legal \TeX{} code during the
save process.

No-one (of course) likes this situation; the \LaTeX{}3 team have removed the need for protection of
some things in the production of \LaTeX{}2ε, but the techniques available to them within current
\LaTeX mean that this is an expensive exercise. It remains a long-term aim of the team to remove all need for these things.

124 Why doesn’t \verb work within...?

The \LaTeX verbatim commands work by changing category codes. Knuth says of this sort of thing “Some care is needed to get the timing right...”, since once the category code has been assigned to a character, it doesn’t change. So \verb has to assume that it is getting the first look at its parameter text; if it isn’t, \TeX has already assigned category codes so that \verb doesn’t have a chance. For example:

\verb+\error+

will work (typesetting ‘\error’), but

\verb+
newcommand{\unbrace}{[1]{#1}
unbrace{\verb+\error+}
\verb+

will not (it will attempt to execute \error). Other errors one may encounter are ‘\verb ended by end of line’, or even ‘\verb illegal in command argument’.

This is why the \LaTeX book insists that verbatim commands must not appear in the argument of any other command; they aren’t just fragile, they’re quite unusable in any command parameter, regardless of \protection (question 123).

125 Case-changing oddities

\TeX provides two primitive commands \uppercase and \lowercase to change the case of text; they’re not much used, but are capable creating confusion.

The two commands do not expand the text that is their parameter — the result of \uppercase{abc} is ‘ABC’, but \uppercase{\abc} is always ‘\abc’, whatever the meaning of \abc. The commands are simply interpreting a table of equivalences between upper- and lowercase characters. They have (for example) no mathematical sense, and

\begin{verbatim}
\uppercase{About $y=f(x)$}
\end{verbatim}

will produce

ABOUT $Y=F(X)$

which is probably not what is wanted.

In addition, \uppercase and \lowercase do not deal very well with non-American characters, for example \uppercase{\ae} is the same as \ae.

\LaTeX provides commands \MakeUppercase and \MakeLowercase which fixes the latter problem. These commands are used in the standard classes to produce upper case running heads for chapters and sections.

Unfortunately \MakeUppercase and \MakeLowercase do not solve the other problems with \uppercase, so for example a section title containing \begin{tabular}... \end{tabular} will produce a running head containing \begin{TABULAR}. The simplest solution to this problem is using a user-defined command, for example:
Note that \texttt{mytable} has to be protected, otherwise it will be expanded and made upper case.

\section{A section title \protect\mytable{} with a table}

Note that \texttt{mytable} has to be protected, otherwise it will be expanded and made upper case.

\subsection*{126 Why are \# signs doubled in macros?}

The way to think of this is that \texttt{##} gets replaced by \# in just the same way that \texttt{#1} gets replaced by ‘whatever is the first argument’.

So if you define a macro and use it as:

\begin{verbatim}
\def\a#1{+++#1+++#1+++#1+++} \a{b}
\end{verbatim}

the macro expansion produces ‘+++
\texttt{b}+++
\def\x \texttt{b}+++
\def\x \texttt{b}+++’, which people find normal. However, if we now replace part of the macro:

\begin{verbatim}
\def\a#1{+++#1+++\def\x #1{xxx#1}}
\end{verbatim}

\texttt{a\{b\}} will expand to ‘+++\texttt{b}+++\def\x \texttt{b}+++\def\x \texttt{b}+++’. This defines \texttt{x} to be a macro delimited by \texttt{b}, and taking no arguments, which people may find strange, even though it is just a specialisation of the example above. If you want \texttt{a} to define \texttt{x} to be a macro with one argument, you need to write:

\begin{verbatim}
\def\a#1{+++#1+++\def\x #1{xxx#1}}
\end{verbatim}

and \texttt{a\{b\}} will expand to ‘+++\texttt{b}+++\def\x \texttt{b}+++’, because \texttt{#1} gets replaced by ‘\texttt{b}’ and \texttt{##} gets replaced by \#.

To nest a definition inside a definition inside a definition then you need \texttt{####1}, as at each stage \texttt{##} is replaced by \#. At the next level you need 8 \texttt{#s} each time, and so on.

\subsection*{127 Why does \LaTeX{} split footnotes across pages?}

\LaTeX{\kern-.1667em} splits footnotes when it can think of nothing better to do. Typically, when this happens, the footnote mark is at the bottom of the page, and the complete footnote would overfill the page. \LaTeX{\kern-.1667em} could try to salvage this problem by making the page short of both the footnote and the line with the footnote mark, but its priorities told it that splitting the footnote would be preferable.

As always, the best solution is to change your text so that the problem doesn’t occur in the first place. Consider whether the text that bears the footnote could move earlier in the current page, or on to the next page.

If this isn’t possible, you might want to change \LaTeX{\kern-.1667em}’s perception of its priorities: they’re controlled by \texttt{\interfootnotelinepenalty} — the larger it is, the less willing \LaTeX{\kern-.1667em} is to split footnotes.

Setting

\begin{verbatim}
\interfootnotelinepenalty=10000
\end{verbatim}

inhibits split footnotes altogether, which will cause ‘\texttt{Underfull \vbox}’ messages unless you also
specify \raggedbottom. The default value of the penalty is 100, which is rather mild.

An alternative technique is to juggle with the actual size of the pages. \enlargethispage changes the size of the current page by its argument (for example, you might say \enlargethispage{\baselineskip} to add a single line to the page, but you can use any ordinary \TeX length such as 15mm or -20pt as argument). Reducing the size of the current page could force the offending text to the next page; increasing the size of the page may allow the footnote to be included in its entirety. It may be necessary to change the size of more than one page.

128 Getting \marginpar on the right side

In \texttt{twoside} documents, \LaTeX makes stirling attempts to put \marginpars in the correct margin (the outer or the gutter margin, according to the user’s command). However, a booby-trap arises because \TeX runs its page maker asynchronously. If a \marginpar is processed while page $n$ is being being built, but doesn’t get used until page $n+1$, then the \marginpar will turn up on the wrong side of the page.

The solution to the problem is for \LaTeX to ‘remember’ which side of the page each \marginpar should be on. The package \texttt{macros/latex/contrib/supported/mparhack} does this, using marks stored in the \texttt{.aux} file.

129 Why does \texttt{PCTeX} run out of ‘\texttt{dimen’s}?

\texttt{PCTeX} uses a lot of \TeX’s counter and length registers. Since \TeX only has 256 of each, it’s therefore often impossible to load \texttt{PCTeX}.

The \texttt{ConTeXt} modules \texttt{m-pictex.tex} (for plain \TeX and variants) or \texttt{m-pictex.sty} (for \LaTeX both files are available from \texttt{macros/context-tmf}). \texttt{m-pictex.tex} provides an ingenious solution to the problem based on hacking the code of \texttt{newdimen} itself.

An alternative is Andreas Schrell’s \texttt{pictexwd} packages (from \texttt{graphics/pictex-addon}, which replaces \texttt{PCTeX} itself and simply uses less registers.

130 Why won’t \texttt{LaTeX} ‘fixed’ names stay changed?

\LaTeX has many commands that define the names that are used for common functions in document — for example, \texttt{\chaptername}, \texttt{\figurename}, and so on. These names allow users to adjust the appearance of their documents, most notably when the document itself is being written in a language other than \LaTeX’s default English.

The \texttt{babel} package, of course, uses this mechanism extensively, setting up a new set of names whenever the user declares that the current language is changing. However, \texttt{babel}’s defaults are not to everyone’s liking, so users often want to change names \texttt{after} they’ve loaded \texttt{babel}.

The simplest approach:

\begin{verbatim}
\usepackage[frenchb]{babel}
\renewcommand{\refname}{Bibliographie}
\end{verbatim}
doesn’t work: babel actually performs its \selectlanguage operation at the start of the document, and so the \renewcommand is lost. For a single-language document, the sequence:

\usepackage[frenchb]{babel}
\AtBeginDocument{\renewcommand{\refname}{Bibliographie}}

will overcome this problem, but this solution is not stable — if the document later becomes a multi-language one, the original version of \refname will be restored every time the language switches back to French. The solution is to use the babel command \addto, as follows:

\usepackage[frenchb]{babel}
\addto{\captionsfrenchb}{\renewcommand{\refname}{Bibliographie}}

This last code will establish the new name for use at every change of to the language, and is the recommended way for defining new names. The \captionsfrenchb shown above is an example: in general the name to use is \captions<your language>, where <your language> is the language option given to babel in the first place.

Part R Current \TeX Projects

131 \LaTeX 2\varepsilon (the ‘new’ standard \LaTeX)

\LaTeX 2\varepsilon is the version of \LaTeX prepared and supported by the \LaTeX3 project team (question 132). With the advent of \LaTeX 2\varepsilon, users gained a formal support structure, and the knowledge that bug fixes would be incorporated into the distribution on a fairly regular basis (new releases appear at approximately 6-month intervals).

\LaTeX 2\varepsilon is so structured that significant enhancements may be added using macro add-ons, rather than by the different \TeX formats that were the bane of the \LaTeX 2.09 world.

\LaTeX 2\varepsilon is upwardly compatible with \LaTeX 2.09, but has new features. In the latest (June 1999) release, these include:

- The font selection scheme is different (based on the ‘New Font Selection Scheme’ (NFSS) described by Frank Mittelbach and Rainer Schöpf in TUGboat 10(2). The NFSS the user may select a font by specifying an array of required properties (such as size, weight and series); commands that provide the \LaTeX 2.09 syntax remain.
- \SL\TeX is now merely a different document class, so that there is no longer a need for a separate format.
- Better control of floating environments, such as figures.
- There is a documented interface for package and class writers (though not yet for designers).
- The box commands have been enhanced, with e.g., options to specify the height of a minipage.
• Several standard commands are no longer fragile (question 123); they can therefore be included in the argument of commands such as \caption without being protected.

• \newcommand can define commands with one optional argument; such commands are automatically robust; there is also a separate command to define robust commands.

• There are standard packages for graphics inclusion and colour typesetting.

• There is standard support for typesetting in cyrillic.

132 The \LaTeX\TeX3 project

The \LaTeX\TeX3 project team (see http://www.latex-project.org/latex3.html) is a small group of volunteers whose aim is to produce a major new document processing system based on the principles pioneered by Leslie Lamport in the current \LaTeX. It will remain freely available and it will be fully documented at all levels.

The \LaTeX\TeX3 team’s first product, \LaTeX\TeX\ 2ε (question 131), was delivered in 1994, and the team undertakes its maintenance.

133 The Omega project

Omega (Ω) is a program built as an extension of the \TeX\ sources which works internally with 16-bit characters (Unicode); this allows it to work with most scripts in the world with much reduced coding scheme complications. Omega also has a powerful concept of input and output filters to allow the user to work with existing transliteration schemes, etc. Omega is an ongoing project by John Plaice (plaice@cse.unsw.edu.au) and Yannis Haralambous (Yannis.Haralambous@univ-lille1.fr). An email discussion list is available: subscribe by sending a message ‘subscribe omega <your name>’ to listserv@ens.fr

The base distribution of Ω was released in November 1996; it is available via systems/omega

Implementations of Omega are available as part of the te\TeX, mik\TeX, fp\TeX and CMac\TeX distributions (QrefTeX-systems).

134 The \N\TeXS project

The \N\TeXS project first saw the light of day at the Hamburg meeting of DANTE during 1992, as a response to an aspiration to produce something even better than \TeX. The project is not simply enhancing \TeX, for two reasons: first, that \TeX itself has been frozen by Knuth (see question 16), and second, even if they were allowed to develop the program, some members of the \N\TeXS team feel that \TeX in its present form is simply unsuited to further development. While all those involved in the project are involved with, and committed to, \TeX, they recognise that the end product may very well have little in common with \TeX other than its philosophy.

Initially, and despite the reservations expressed at the inaugural meeting, the group is concentrating on extending \TeX per se: members are implementing extensions and enhancements to \TeX through the standard medium of a change-file. These extensions and enhancements,
together with \TeX proper, form a system called ’\TeX’, which is 100% compatible with \TeX; furthermore, it is possible during format creation to construct a format that is \TeX; no extensions or enhancements are present.

The final aim of the project will be to produce an entirely new typesetting system, building on the experience gained in the earlier phases. This system is intended to provide a stable basis for typesetting in the future, in the way that \TeX has since it was first offered to the world.

A distribution of ’\TeX’ was made available in November 1996. It is available via systems/e-tex; ’\TeX’ is also distributed on the \TeX Live CD-ROM (see question 43).

135 The PDFTeX project

PDFTeX (formerly known as \TeX2PDF) arose from Han The Thanh’s post-graduate research at Masaryk University, Brno, Czech Republic. The basic idea is very simple: to provide a version of \TeX that can output PDF as an alternative format to DVI. PDFTeX implements a small number of new primitives, to switch to PDF output, and to control various PDF features. Han The Thanh is continuing work on the project; the latest version is available from systems/pdftex, and a version was distributed on the \TeX Live CD-ROM (see question 43).

Part S  Perhaps There isn’t an Answer

136 What to do if you find a bug

For a start, make entirely sure you have found a bug. Double-check with books about \TeX, L\LaTeX, or whatever you’re using; compare what you’re seeing against the other answers above; ask every possible person you know who has any \TeX-related expertise. The reasons for all this caution are various.

If you’ve found a bug in \TeX itself, you’re a rare animal indeed. Don Knuth is so sure of the quality of his code that he offers real money prizes to finders of bugs; the cheques he writes are such rare items that they are seldom cashed. If you think you have found a genuine fault in \TeX itself (or METAFONT, or the CM fonts, or the \TeXbook), don’t immediately write to Knuth, however. He only looks at bugs once or twice a year, and even then only after they are agreed as bugs by a small vetting team. In the first instance, contact Barbara Beeton at the AMS (bnb@math.ams.org), or contact TUG (question 17).

If you’ve found a bug in L\LaTeX 2\epsilon, look in the bugs database to see if it’s already been reported. If not you should submit details of the bug to the L\LaTeX3 team. To do this, you should process the file latexbug.tex with L\LaTeX (the file is part of the L\LaTeX 2\epsilon distribution. The process will give you instructions about what to do with your bug report (it can, for example, be sent to the team by email). Please be sparing of the team’s time; they’re doing work for the good of the whole \LaTeX community, and any time they spend tracking down non-bugs is time not available to write or debug new code. Details of the whole process, and an interface to the database, are available via http://www.latex-project.org/help.html

If you’ve found a bug in L\LaTeX 2.09, or some other such unsupported software, there’s not a lot you can do about it. You may find help or de facto support on a newsgroup such as comp.tex.tex or on a mailing list such as texhax@tex.ac.uk, but posting non-bugs to
any of these forums can lay you open to ridicule! Otherwise you need to go out and find
yourself a willing ТpX-consultant — TUG maintains a register of ТpX consultants (see http://www.tug.org/consultants.html).