

The New T_EX FAQ

Your 121 Questions Answered

version 2.2o, date 1999/06/09

The UK T_EX Users Group Committee
after the original maintained by
Bobby Bodenheimer

June 9, 1999

NOTE

This document is a version of the FAQ article that was published as the December 1994 and 1995 editions of the UKTUG magazine *Baskerville* (which weren't formatted like this). The article is also available via the World Wide Web.

Contents

<p>A Introduction 2</p> <p>B The Background 3</p> <p>1 What is T_EX? 3</p> <p>2 How should I pronounce "T_EX"? 3</p> <p>3 What is METAFONT? 3</p> <p>4 What is Metapost? 3</p> <p>5 How can I be sure it's really T_EX? 3</p> <p>6 Are T_EX and friends Y2K compliant? 4</p> <p>7 What is L^AT_EX? 4</p> <p>8 How should I pronounce "L^AT_EX(2ϵ)"? 4</p> <p>9 Should I use plain T_EX or L^AT_EX? 4</p> <p>10 What are the AMS packages (<i>A_MS-T_EX</i>, etc.)? 4</p> <p>11 What is Eplain? 5</p> <p>12 What is Lollipop? 5</p> <p>13 What is Texinfo? 5</p> <p>14 If T_EX is so good, how come it's free? 5</p> <p>15 What is the future of T_EX? 5</p> <p>16 What are TUG and <i>TUGboat</i>? 5</p> <p>17 Are there nationally-based user groups, too? 6</p> <p>18 TUG Technical Working Groups 6</p> <p>C Documentation and Help 6</p> <p>19 Books on T_EX and its relations 6</p> <p>20 Where to find this article 7</p> <p>21 Mailing lists about T_EX and its friends 7</p> <p>22 B_IB_T_EX Documentation 8</p> <p>23 The P_IC_T_EX manual 8</p> <p>24 Finding (L^A)T_EX macro packages 8</p> <p>25 Finding files in the CTAN archives 8</p> <p>D Bits and pieces of T_EX 9</p> <p>26 What is a DVI file? 9</p> <p>27 What is a driver? 9</p> <p>28 What are PK files? 9</p> <p>29 What are TFM files? 9</p> <p>30 Virtual fonts 9</p> <p>31 \special commands 9</p>	<p>32 Documented L^AT_EX sources (.dtx files) 10</p> <p>33 What are the EC fonts? 10</p> <p>E Acquiring the Software 10</p> <p>34 Repositories of T_EX material 10</p> <p>35 Contributing a file to the archives 11</p> <p>36 Finding new fonts 11</p> <p>37 T_EX CD-ROMs 11</p> <p>F T_EX Systems 11</p> <p>38 (L^A)T_EX for different machines 11</p> <p>39 T_EX-friendly editors and shells 12</p> <p>40 Commercial T_EX implementations 13</p> <p>G DVI Drivers and Previewers 14</p> <p>41 DVI to PostScript conversion programs 14</p> <p>42 DVI drivers for HP LaserJet 14</p> <p>43 DVI previewers 15</p> <p>H Support Packages for T_EX 15</p> <p>44 Fig, a T_EX-friendly drawing package 15</p> <p>45 T_EXCAD, a drawing package for L^AT_EX 15</p> <p>46 Spelling checkers for work with T_EX 15</p> <p>47 The V_OR_T_EX package 15</p> <p>I Literate programming 15</p> <p>48 What is Literate Programming? 15</p> <p>49 WEB for C, FORTRAN, and other languages 16</p> <p>J Format conversions 16</p> <p>50 Conversion between (L^A)T_EX and others 16</p> <p>51 Conversion from (L^A)T_EX to plain ASCII 16</p> <p>52 Conversion from SGML or HTML to T_EX 17</p> <p>53 (L^A)T_EX conversion to HTML 17</p> <p>54 Making hypertext documents from T_EX 17</p> <p>55 Making Acrobat documents from L^AT_EX 18</p> <p>K METAFONT 18</p> <p>56 Getting METAFONT to do what you want 18</p> <p>57 Which font files should be kept 19</p> <p>58 Getting bitmaps from the archives 19</p>
---	--

59	Using PostScript fonts with T _E X	19
60	Previewing files using PostScript fonts	20
61	T _E X font metric files for PostScript fonts	20
62	Problems using PostScript fonts	20
63	Choice of scalable outline fonts	21
64	Including a PostScript figure in L ^A T _E X	22
M Special sorts of typesetting		22
65	Drawing with T _E X	22
66	Double-spaced documents in L ^A T _E X	22
67	Formatting a thesis in L ^A T _E X	22
68	Flowing text around figures in L ^A T _E X	23
69	Alternative head- and footlines in L ^A T _E X	23
70	Including a file in verbatim in L ^A T _E X	23
71	Including line numbers in typeset output	23
72	Generating an index in (L ^A)T _E X	24
73	Typesetting URLs	24
74	Citing URLs with BIBT _E X	24
75	Using BIBT _E X with plain T _E X	24
76	Typesetting music in T _E X	25
77	Drawing Feynman diagrams in L ^A T _E X	25
N How do I do X in T_EX or L^AT_EX		25
78	Proof environment	25
79	Symbols for the number sets	25
80	Roman theorems	25
81	Fancy enumeration lists	25
82	Unnumbered sections in the Table of Contents	26
83	Footnotes in tables	26
84	Style of section headings	26
85	Indent after section headings	26
86	Footnotes in L ^A T _E X section headings	27
87	Changing the margins in L ^A T _E X	27
88	Finding the width of a letter, word, or phrase	27
89	Excluding blocks of text from the DVI file	27
90	Setting bold Greek letters in L ^A T _E X	27
91	Defining a new log-like function in L ^A T _E X	28

O Things are Going Wrong...		28
93	Weird hyphenation of words	28
94	(Merely) peculiar hyphenation	28
95	Accented words aren't hyphenated	28
96	Enlarging T _E X	29
97	Moving tables and figures in L ^A T _E X	29
98	\pagestyle{empty} on first page in L ^A T _E X	29
99	Underlined text won't break	29
100	Odd behaviour of \rm, \bf, etc.	30
101	Old L ^A T _E X font references such as \tenrm	30
102	Missing symbols	30
103	L ^A T _E X gets cross-references wrong	30
104	\@ and @ in macro names	30
105	Where are the msx and msy fonts?	30
106	Where are the am fonts?	30
107	'String too long' in BIBT _E X	31
P Why does it do that?		31
108	What's going on in my \include commands?	31
109	Why does it ignore paragraph parameters?	31
110	What's the reason for 'protection'?	32
111	Why doesn't \verb work within...?	32
112	Case-changing oddities	32
113	Why are # signs doubled in macros?	32
114	Why does L ^A T _E X split footnotes across pages?	33
Q Recent Developments		33
115	The New Font Selection Scheme (NFSS)	33
116	L ^A T _E X 2 _ε (the 'new' standard L ^A T _E X)	33
117	The L ^A T _E X3 project	33
118	The Omega project	33
119	The N _T S project	34
120	The PDFT _E X project	34
R Perhaps There <i>isn't</i> an Answer		34
121	What to do if you find a bug	34



A Introduction

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

Usenet is a mechanism for exchanging articles between people who share interests or needs²; 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 T_EX that have accumulated over the years; so we decided to update the list and publish it in *Baskerville*; we are grateful to Bobby for his permission to use his article in this way. As a *quid pro quo*, we are making the source of the article freely available (usergrps/uktug/faq), and it can be compiled by anyone who runs reasonably current L^AT_EX 2_ε (see question 116), 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 T_EX 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 <http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes>

A hypertext version is also to be found on the T_EX Live

¹For 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 Wujastyk.

²Usenet, 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

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 Arseneau, Barbara Beeton, Karl Berry, Damian Cugley, Michael Downes, John Hobby, Berthold Horn, Werner Icking, David Kastrup, Ted Nieland, Pat Rau, Piet van Oostrum, Oren Patashnik, Joachim Schrod, Philip Taylor, Ulrik Vieth, Rick Zaccane 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 Kaveh Bazargan, Malcolm Clark, 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.

Finding the Files

Unless otherwise specified, all files mentioned in this article are available from a CTAN archive, or from one of their mirrors. Question 34 gives details of the CTAN archives, and how to retrieve files from them. If you don't have access to the Internet, question 37 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 stop³ 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!

B The Background

1 What is \TeX ?

\TeX is a typesetting system written by Donald E. Knuth, who says in the Preface to his book on \TeX (see question 19) 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 \TeX , and he provides the literate (WEB) source of \TeX 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 \TeX does. Furthermore, the WEB system provides mechanisms to port \TeX 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 \TeX system. \TeX and its documents are therefore highly portable.

programming capability. For this reason, \TeX on its own is a pretty difficult beast to deal with, so Knuth provided a package of macros for use with \TeX called `plain \TeX` ; `plain \TeX` is effectively the minimum set of macros one can usefully employ with \TeX , 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 \TeX ", they usually mean they're programming in `plain \TeX` .

2 How should I pronounce " \TeX "?

The 'X' stands for the Greek letter Chi (χ), 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 \TeX ; whereas \TeX defines the layout of glyphs on a page, METAFONT defines the shapes of the glyphs and the relations between 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 implements a picture-drawing language very much like that of METAFONT except that it outputs PostScript commands instead of run-length-encoded bitmaps. Metapost is a powerful language for producing figures for documents to be printed on PostScript printers. It provides access to all the features of PostScript and it includes facilities for integrating text and graphics. (Knuth tells us that he uses nothing else 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 (see question 48) which is designed to be portable across a wide range of computer systems. How, then, is a new version of \TeX checked?

³'Full stop' (British English)== 'period' (American English)

own suite of tests to check that his software runs: those who port $\text{T}_{\text{E}}\text{X}$ and its friends to other platforms do indeed perform such tests.

Knuth, however, provides a ‘conformance test’ for both $\text{T}_{\text{E}}\text{X}$ (`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 $\text{T}_{\text{E}}\text{X}$. They are, to the casual reader, pretty incomprehensible!

Once an implementation of $\text{T}_{\text{E}}\text{X}$ has passed its `trip`, or and implementation of `METAFONT` has passed its `trap`, test it may reasonably be distributed as a working version.

6 Are $\text{T}_{\text{E}}\text{X}$ and friends Y2K compliant?

Crashing: None of $\text{T}_{\text{E}}\text{X}$, `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 implementators 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 `\year` primitive: Certification of a $\text{T}_{\text{E}}\text{X}$ implementation (see question 5) does not require that `\year` return a meaningful value (which means that $\text{T}_{\text{E}}\text{X}$ can, in principle, be implemented on platforms that don’t make the value of the clock available to user programs). The *$\text{T}_{\text{E}}\text{X}$ book* (see question 19) defines `\year` as “the current year of our Lord”, which is the only correct meaning for `\year` for those implementations which can supply a meaningful value, which is to say nearly all of them.

In short, $\text{T}_{\text{E}}\text{X}$ implementations should provide a value in `\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 $\text{T}_{\text{E}}\text{X}$, then `\year` will of course return an incorrect value. $\text{T}_{\text{E}}\text{X}$ cannot be considered Y2K compliant, in this sense, on a system that is not itself Y2K compliant.

Macros: $\text{T}_{\text{E}}\text{X}$ macros can in principle perform calculations on the basis of the value of `\year`. The $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ suite (see question 7) performs such calculations in a small number of places; the calculations performed in the current (supported) version of $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ are known to be Y2K compliant.

Other macros and macro packages should be individually checked.

be individually checked.

7 What is $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$?

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ is a $\text{T}_{\text{E}}\text{X}$ macro package, originally written by Leslie Lamport, that provides a document processing system. $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 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 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ “*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: $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ can meet most user requirements, but finding out *how* is often tricky.

8 How should I pronounce “ $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}(2_{\epsilon})$ ”?

Lamport never recommended how one should pronounce $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$, but a lot of people pronounce it ‘Lay $\text{T}_{\text{E}}\text{X}$ ’ or perhaps ‘Lah $\text{T}_{\text{E}}\text{X}$ ’ (with $\text{T}_{\text{E}}\text{X}$ pronounced as the program itself; see question 2).

The ‘epsilon’ in ‘ $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ ’ is supposed to be suggestive of a small improvement over the old $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2.09$. Nevertheless, most people pronounce the name as ‘ $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ -two-ee’.

9 Should I use plain $\text{T}_{\text{E}}\text{X}$ or $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$?

There’s no straightforward answer to this question. Many people swear by plain $\text{T}_{\text{E}}\text{X}$, 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 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ styles for journals and books, and insist that authors stick closely to their markup.

10 What are the AMS packages ($\mathcal{A}\mathcal{M}\mathcal{S}\text{-T}_{\text{E}}\text{X}$, etc.)?

$\mathcal{A}\mathcal{M}\mathcal{S}\text{-T}_{\text{E}}\text{X}$ is a $\text{T}_{\text{E}}\text{X}$ macro package, originally written by Michael Spivak for the American Mathematical Society (AMS) during 1983–1985. It is described in “*The Joy of $\text{T}_{\text{E}}\text{X}$* ” by Michael D. Spivak (second edition, AMS, 1990, ISBN 0-821-82997-1). It is based on plain $\text{T}_{\text{E}}\text{X}$, but 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 $\text{T}_{\text{E}}\text{X}$ runs), and other things.

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 $\AMS\TeX$.

11 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 (see question 13). 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 `\section`, which would format section headings in an “appropriate” way, as \LaTeX 's `\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.

12 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 19). 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 *TUGboat* 13(3).

13 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

information, known as an “Info file”, with an information-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/texinfo.tar.gz`

14 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 40 lists the major vendors.

15 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 π (at the time of writing, Knuth's latest release is version 3.14159). Knuth wants \TeX to be frozen at version π 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 117) 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 118 and 119.

16 What are TUG and TUGboat?

TUG is the \TeX Users Group. *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 micro-computer software on disk. TUG has a Technical Council to coordinate \TeX -related developments (see question 18). Enquiries should be directed to:

\TeX Users Group
1466 NW Front Avenue, Suite 3141
Portland, OR 97209
USA
Tel: (+1) 503-223-9994

Email: tug@mail.tug.org
Web: <http://www.tug.org/>
CTAN details: [usergrps/tug](http://www.tug.org/usergrps/tug)

17 Are there nationally-based user groups, too?

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

DANTE, Deutschsprachige Anwendervereinigung
T_EX e.V.
Postfach 10 18 40
D-69008 Heidelberg
Germany
Tel: (+49) 06221 297 66
Fax: (+49) 06221 16 79 06
Email: dante@dante.de
Web: <http://www.dante.de/>
CTAN details: [usergrps/dante](http://www.tug.org/usergrps/dante)

Association GUTenberg,
BP 10,
93220 Gagny principal,
France
Email: gut@irisa.fr
Web: <http://www.ens.fr/gut/>
CTAN details: [usergrps/gut](http://www.tug.org/usergrps/gut)

NTG
Postbus 394, 1740AJ Schagen,
The Netherlands
Email: ntg@nic.surfnet.nl
Web: <http://www.ntg.nl/>
CTAN details: [usergrps/ntg](http://www.tug.org/usergrps/ntg)

UK T_EX Users' Group,
% Peter Abbott,
1 Eymore Close,
Selly Oak,
Birmingham B29 4LB
UK
Tel: (+44) 0121 476 2159
Email: UKTuG-Enquiries@tex.ac.uk
Web: <http://uk.tug.org/uk-tug>
CTAN details: [usergrps/uktug](http://www.tug.org/usergrps/uktug)

A listing of all known groups is available as [usergrps/info/usergrps.tex](http://www.tug.org/info/usergrps.tex)

18 TUG Technical Working Groups

TUG (see question 16) has an autonomous Technical Council which can appoint Technical Working Groups related to the specific areas of interest to the T_EX 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:

To create font encoding standards for Mathematical fonts used in T_EX systems.

Contact: Barbara Beeton (bnb@math.ams.org)

WG-92-05 T_EX Archive Guidelines.

To develop guidelines for the effective management and utilisation of major T_EX 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_EX Directory Structure.

To identify a universal directory structure for macros, fonts and other related T_EX software so that recommendations can be made to all suppliers of T_EX software.

The group's current set of proposals are to be found on CTAN at [tds/draft-standard](http://www.tug.org/tds/draft-standard)

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_EX and Linguistics.

To study and discuss the requirements for typesetting linguistics in T_EX 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)

C Documentation and Help

19 Books on T_EX and its relations

While Knuth's book is the definitive reference for T_EX, there are other books covering T_EX:

The T_EXbook by Donald Knuth (Addison-Wesley, 1984, ISBN 0-201-13447-0, paperback ISBN 0-201-13448-9)

A Beginner's Book of T_EX by Raymond Seroul and Silvio Levy, (Springer Verlag, 1992, ISBN 0-387-97562-4)

Introduction to T_EX by Norbert Schwarz (Addison-Wesley, 1989, ISBN 0-201-51141-X)

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

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

TeX for the Impatient by Paul W. Abrahams, Karl Berry and Kathryn A. Hargreaves (Addison-Wesley, 1990, ISBN 0-201-51375-7)

TeX in Practice by Stephan von Bechtolsheim (Springer Verlag, 1993, 4 volumes, ISBN 3-540-97296-X for the set, or Vol. 1: 0-387-97595-0, Vol. 2: 0-387-97596-9, Vol. 3: 0-387-97597-7, and Vol. 4: 0-387-97598-5)

TeX: Starting from \square ⁴ by Michael Doob (Springer Verlag, 1993, ISBN 3-540-56441-1)

The Advanced TeXbook by David Salomon (Springer Verlag, 1995, ISBN 0-387-94556-3)

For \LaTeX , see:

LaTeX, a Document Preparation System by Leslie Lamport (second edition, Addison Wesley, 1994, ISBN 0-201-15790-X)

A guide to LaTeX 2 ϵ Helmut Kopka and Patrick W. Daly (third edition, Addison-Wesley, 1998, ISBN 0-201-39825-7)

The LaTeX Companion by Michel Goossens, Frank Mittelbach, and Alexander Samarin (Addison-Wesley, 1993, ISBN 0-201-54199-8)

The LaTeX Graphics Companion: Illustrating documents with TeX and PostScript by Michel Goossens, Sebastian Rahtz and Frank Mittelbach (Addison-Wesley, 1997, ISBN 0-201-85469-4)

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

LaTeX Notes: Practical Tips for Preparing Technical Documents by J. Kenneth Shultis (Prentice Hall, 1994, ISBN 0-131-20973-6)

LaTeX Line by Line by Antoni Diller (John Wiley & Sons, 1993, ISBN 0-471-93471-2)

LaTeX for Scientists and Engineers by David J. Buerger (McGraw-Hill, 1990, ISBN 0-070-08845-4)

TeX Unbound: LaTeX and TeX strategies for fonts, graphics, and more by Alan Hoenig (Oxford University Press, 1998, ISBN 0-19-509685-1 hardback, ISBN 0-19-509686-X paperback)

Math into TeX: A Simplified Introduction using $\mathcal{A}\mathcal{M}\mathcal{S}$ -LaTeX by George Grätzer (Birkhäuser, 1993, ISBN 0-817-63637-4, or, in Germany, ISBN 3-764-33637-4)

Math into LaTeX: An Introduction to LaTeX and $\mathcal{A}\mathcal{M}\mathcal{S}$ -LaTeX by George Grätzer (Birkhäuser, 1996, ISBN 0-817-63805-9)

Daly's, and Grätzer's "*Math into LaTeX*" cover $\LaTeX 2\epsilon$. A sample of the last, in Adobe Acrobat format, is also available (info/mil/mil.pdf).

The list for METAFONT is rather short:

The METAFONTbook by Donald Knuth (Addison Wesley, 1986, ISBN 0-201-13445-4, ISBN 0-201-52983-1 paperback)

A book covering a wide range of topics (including installation and maintenance) is:

Making TeX Work by Norman Walsh (O'Reilly and Associates, Inc, 1994, ISBN 1-56592-051-1)

This list only covers books in English: UK TUG cannot hope to maintain a list of books in languages other than our own.

20 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/de-tex-faq`, and Marie-Paule Kluth posts a FAQ in French to `fr.comp.text.tex`, which is archived as `help/LaTeX-FAQ-francaise`

21 Mailing lists about TeX and its friends

There are (still) people who can use networks but can't read Usenet news; for them, not all is lost if they can send and receive email.

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.

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, Meta-post) 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.

⁴That's 'Starting from Square One'

BIB_TE_X, a program originally designed to produce bibliographies in conjunction with L_AT_EX, is explained in Section 4.3 and Appendix B of Leslie Lamport's L_AT_EX manual (see question 19). The document "BIB_TE_Xing", contained in the file `btxdoc.tex`, gives a more complete description. *The L_AT_EX Companion* (see question 19) also has information on BIB_TE_X and writing BIB_TE_X style files.

The document "Designing BIB_TE_X Styles", contained in the file `btXHak.tex`, explains the postfix stack-based language used to write BIB_TE_X styles (`.bst` files). The file `btXbst.doc` is the template for the four standard styles (`plain`, `abbrv`, `alpha`, `unsrc`). It also contains their documentation. The complete BIB_TE_X documentation set (including the files above) is in `biblio/bibtex/distrib/doc`

There is a Unix BIB_TE_X *man* page in the *web2c* package (see question 38). Any copy you may find of a *man* page written in 1985 (before "BIB_TE_Xing" and "Designing BIB_TE_X Styles" appeared) is obsolete, and should be thrown away.

23 The P_CT_EX manual

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

24 Finding (L_A)T_EX macro packages

Before you ask for a T_EX macro or L_AT_EX class or package file to do something, try searching Graham Williams' (Graham.Williams@dit.csiro.au) catalogue, available as `help/Catalogue/catalogue.html`, or for efficient interactive searching via <http://www.tex.ac.uk/tex-archive/help/Catalogue/ctindex.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 25). For packages listed in *The L_AT_EX Companion* (see question 19), the file `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.

An alternative procedure is to use <http://ctan.tug.org/ctan/>, which permits limited 'keyword' searching for files on the CTAN sites.

25 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 <http://www.dante.de/cgi-bin/ctan-index>

To get the best use out of the ftp facility you should remember that `<term>` is a *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

`dtX` rather than as package files with extension `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 L_AT_EX 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-F]" matches any single character from the set {"a","b","c","D","E","F"};
- "*" placed after an expression matches zero or more occurrences so "a*" matches "a" and "aaaa", and "[a-zA-Z]*" matches a 'word';

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.

D Bits and pieces of T_EX

26 What is a DVI file?

A DVI file (that is, a file with the type or extension .dvi) is T_EX’s main output file, using T_EX in its broadest sense to include L^AT_EX, 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 (see question 27) 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 T_EX’s internal coding; a T_EX input file should produce the same DVI file regardless of which implementation of T_EX 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 \special commands (see question 31).

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

27 What is a driver?

A driver is a program that takes as input a dvi file (see question 26) 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 28, 29, 30 and 59.

28 What are PK files?

PK files (packed raster) contain font bitmaps. The output from METAFONT (see question 56) 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 type at a site. As a result, they are often held in an elaborate directory structure, or in ‘font library files’, to regularise access.

TFM stands for T_EX 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 T_EX, 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 T_EX (L^AT_EX, etc.), but are not, generally, needed by the printer driver.

30 Virtual fonts

Virtual fonts for T_EX were first implemented by David Fuchs in the early days of T_EX, 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 T_EX about something more complicated than just a one-to-one character mapping. The entities you define in a virtual font look like characters to T_EX (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 61) or to build ‘fake’ maths fonts.

It is important to realise that T_EX itself does *not* see virtual fonts; for every virtual font read by the DVI driver there is a corresponding TFM file read by T_EX. 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 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 61. *fonts/utilities/qdtevxpl* is another utility for creating ad-hoc virtual fonts.

31 \special commands

T_EX provides the means to express things that device drivers can do, but about which T_EX itself knows nothing. For example, T_EX 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 T_EX 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. L^AT_EX 2_ε 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

divscr, etc.) that will draw lines at arbitrary orientations, and commands in *dvitoln03* that permit the page to be set in landscape orientation.

32 Documented L^AT_EX sources (.dtx files)

L^AT_EX 2_ε, and many support macro packages, are now written in a literate programming style (see question 48), 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 .dtx, and should normally be stripped of documentation before use with L^AT_EX. Alternatively you can run L^AT_EX on a .dtx file to produce a nicely formatted version of the documented code. An installation script (with suffix .ins) is usually provided, which needs the standard L^AT_EX 2_ε *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 L^AT_EX system). Several packages can be included in one .dtx file, with conditional sections, and there facilities for indices of macros etc. Anyone can write .dtx files; the format is explained in *The L^AT_EX Companion* (see question 19). There are no programs yet to assist in composition.

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

33 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 T_EX 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 T_EX 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 T_EX 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.) L^AT_EX 2_ε (see question 116) 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 95).

The only METAFONT-fonts that conform to the Cork encoding are the EC fonts (available as *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 pro-

are encoded according to the L^AT_EX 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 (see question 59), for PostScript fonts.

E Acquiring the Software

34 Repositories of T_EX material

To aid the archiving and retrieval of of T_EX-related files, a TUG working group developed the Comprehensive T_EX 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 `ftp.dante.de` (129.206.100.192), `ftp.tex.ac.uk` (128.232.1.87) and `ctan.tug.org` (158.121.106.16). The organisation of T_EX files on all these sites is identical and starts at `tex-archive/`. 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 *finger* utility on ‘user’ `ctan@ftp.tex.ac.uk`, `ctan@ftp.dante.de` or `ctan@tug.cs.umb.edu`; it is also available as file `CTAN.sites`

To find software at a CTAN site, use anonymous ftp to the host, and then execute the command ‘`quote site index <term>`’ (see question 25 for details).

The email server `ftpmail@ftp.dante.de` provides an ftp-like interface through mail. Send a message containing just the line ‘help’ to your nearest server for details of use.

Users on BITNET may access anonymous ftp for some files indirectly by sending mail to `BITFTP@PUCC.BITNET`. Send a message containing the line ‘help’ to this address for more information.

There is also the DECUS T_EX 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 T_EX Collection Editor, Ted Nieland (`nieland@ted.hcst.com`).

Finally, of course, the T_EX user who has no access to any sort of network may buy a copy of the archive on CD-ROM (see question 37).

Use anonymous ftp to any CTAN archive (see question 34) 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 (see question 25).

36 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

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 34).

37 T_EX CD-ROMS

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

Ready-to-run T_EX systems on CD-ROM are available:

- A consortium of User Groups (notably TUG, UK TUG and GUTenberg) distribute the T_EX 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 T_EX Live are available at <http://www.tug.org/texlive.html>

- The Dutch T_EX Users Group (NTG) publish the whole 4AllT_EX workbench on a 2-CD-ROM set packed with all the MS-DOS T_EX software, macros and fonts you can want. It is available from NTG direct (see question 17), 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 T_EX-related material. Information about the CD-ROM is available at <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

800 786-9907 (within the USA and Canada)

Fax: (+1) 510 674-0821

Email: info@cdrom.com (for questions) and
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 T_EXcetera 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:

Prime Time Freeware
370 Altair Way, Suite 150
Sunnyvale CA 94086
USA
Tel: (+1) 408 433 9662
Fax: (+1) 408 433 0727
Email: ptf@cfc1.com

or from many CD-ROM resellers, or the TUG office (see question 16). Price will be around \$60. Please note that PTF is not a big commercial firm, and is a good friend of the T_EX community.

F T_EX Systems

38 (L)T_EX for different machines

We list here the free or shareware packages; see question 40 for commercial packages.

Unix Instructions for retrieving the Unix T_EX 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` (browse `systems/unix/teTeX-beta`, which is essentially stable despite its beta status, for relevant files), or the T_EX Live CD-ROM (see question 37).

A useful (though somewhat dated) set of binaries for various common Unix systems is to be found as part of the `teTeX` distribution (browse `systems/unix/teTeX/0.4/distrib/binaries` for your architecture); `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 T_EX to run on Linux, N_TE_X (available as `systems/unix/linux/ntex`) and `teTeX` (browse `systems/unix/teTeX/0.4`).

Beware the Slackware '96 CD-ROM distribution of N_TE_X: 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.

or Windows includes L^AT_EX, B^IB_TE_X, 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 em_TE_X with 386 (and better) processors and under Windows, are included in the distribution.

A second package, g_TE_X, runs under MS-DOS or Windows (and its users speak well of it). It is available from systems/msdos/gtex

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 Mik_TE_X, by Christian Schenk, first arrived on CTAN in 1996. It has been welcomed by those that have used it and reported their experiences. It will run under Windows'95, 98 or Windows/NT; the current version is in systems/win32/miktex

Fabrice Popineau's port of Web2C for Win32 is also available, from systems/win32/fptex

Windows NT, other platforms A Power PC port of Mik_TE_X is available from systems/win32/miktexppc, and an AXP port is available from systems/win32/miktex-AXP

Mac Oz_TE_X is a shareware version of _TE_X 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 nonfree/systems/mac/oztex

UK TUG prepays the shareware fee, so that its members may acquire the software without further payment. Questions about Oz_TE_X may be directed to oztex@midway.uchicago.edu

Another partly shareware program is CMac_TE_X (available as systems/mac/cmactex), put together by Tom Kiffe. This is much closer to the Unix _TE_X setup (it uses *dvips*, for instance). CMac_TE_X includes a port of the latest version of Omega (see question 118).

OpenVMS _TE_X for OpenVMS is available as systems/OpenVMS/TEX97_CTAN.ZIP

Standard tape distribution is through DECUS (see question 34).

Atari _TE_X is available for the Atari ST in systems/atari

If anonymous ftp is not available to you, send a message containing the line 'help' to atari@atari.archive.umich.edu

Amiga Full implementations of _TE_X 3.1 (Pas_TE_X) and META-FONT 2.7 are available in systems/amiga

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

WAITS, and so was easily ported to TOPS-20. A distribution that runs on TOPS-20 is available via anonymous ftp from ftp.math.utah.edu (128.110.198.34) in pub/tex/pub/web

39 _TE_X-friendly editors and shells

There are good _TE_X-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 AU_TE_X mode (support/auctex). This provides menu items and control sequences for common constructs, checks syntax, lays out markup nicely, lets you call _TE_X and drivers from within the editor, and everything else like this that you can think of. Complex, but very powerful.

VMS An *lscdit* mode for editing _TE_X source is available from TUG (see question 16) as _TE_Xniques 1, VAX Language-Sensitive Editor, by Kent MacPherson (1985).

MS-DOS There are several choices:

- The (shareware) 4All_TE_X workbench (systems/msdos/4alltex) provides a very comprehensive environment written in 4DOS which lets you access most _TE_X-related software in a friendly way. You can choose your own editor; something such as *QEdit* or *Brief* is suitable. This whole package is available in easy-to-use form on CD-ROM from _TE_X user groups.
- _TE_Xshell (systems/msdos/texshell) is a simpler, easily-customisable environment, which can be used with the editor of your choice.
- Eddi4_TE_X (systems/msdos/e4t; also shareware) is a specially-written _TE_X editor which features intelligent colouring, bracket matching, syntax checking, online help and the ability to call _TE_X programs from within the editor. It is highly customisable, and features a powerful macro language.

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

Windows 3.1 Your best public domain bet is probably to use MicroEmacs as an editor and control centre for _TE_X programs. The g_TE_X package (systems/msdos/gtex) comes with MicroEmacs ready to go, integrated with _TE_X, previewer, *dvips* and *GhostScript*.

_TE_XtelmExtel (systems/msdos/emtex-contrib/TeXtelmExtel) is a Shell for em_TE_X or W_TE_X 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.

GNU *emacs*, that will run under Windows, is available; thus you can also use AUCT \TeX 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 40 for details of Y&Y).

Scientific Word is a WYSIWYG editing program, strong on maths, which uses L \TeX for output (see question 40 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. L \TeX users who like *make* should try `support/latexmk`

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

40 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 \TeX book!) 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

VAX/VMS; Convergent \TeX Complete system for VAX/VMS machines (a version for Alphas is in preparation); includes L \TeX , 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
6994 Pebble Beach Court
Lake Worth FL 33467
USA
Tel: (+1) 561-966-8400 Fax: (+1) 561-966-0962
Email: kinch@holonet.net
Web: <http://idt.net/~truetex>

Source: News posting from Richard Kinch, October 1997

PC; \TeX "Bitmap free \TeX for Windows."

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: <http://www.YandY.com/>

Source: Y&Y announcement, February 1995

pc \TeX Long-established: pc \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: <http://www.pctex.com/>

Source: Mail from Personal \TeX Inc, September 1997

PC; VT \TeX Also "Bitmap-free".

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: <http://www.micropress-inc.com/>

Source: MicroPress home page, April 1996

PC; microT_EX MicroT_EX and T_EX tools.

Micro Programs, Inc.
251 Jackson Ave.
Syosset, NY 11791
USA
Tel: (+1) 516-921-1351
Email: sales@microprograms.com

Source: AMS listing, November 1994

PC; Scientific Word Scientific Word and Scientific Workplace offer a mechanism for near-WYSIWYG input of L^AT_EX documents; they ship with TrueT_EX 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
Aberdeenshire, 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 T_EX 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: (+1) 503-222-9571
Fax: (+1) 503-222-1643
Email: sales@bluesky.com
Web: <http://www.bluesky.com/>

Source: TUGboat 15(1) (1994)

AmigaT_EX 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

G DVI Drivers and Previewers

41 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 34), and also as part of the distribution of T_EX 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 `dviware/dvitops`

Macintosh users can use either the excellent drivers built into OzT_EX or Textures, or a port of *dvips* in the CMacT_EX package.

42 DVI drivers for HP LaserJet

The emT_EX package (see question 38) 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 41), is available in `dviware/dviljk`

Em \TeX and g \TeX for the PC, and Oz \TeX for the Macintosh, all come with previewers that can be used on those platforms. There is a good OS/2 Presentation Manager previewer in em \TeX , and a public domain Windows previewer (dviware/dviwin). Commercial PC \TeX packages (see question 40) have good MS-DOS and Windows previewers.

Previewers available for other operating systems include:

xdvi The most widely used previewer for the X Window System (and hence almost any Unix or modern VMS workstation); available in dviware/xdvi

Karl Berry's version of *xdvi*, called *xdvik*, has features analogous to his *dvipsk* (see question 41); it is available in dviware/xdvik

dvipage For SunView on (old enough) Sun workstations. This was published in volume 15 of comp.sources.unix and is archived in dviware/dvipage

xtex An older previewer for the X Window System; available in dviware/seetex

dviapollo For Apollo Domain workstations; available in dviware/dviapollo

dvidis For (old enough, VMS) VAXstations running VWS; available in dviware/dvidis

dvitovdu for Tektronix 4010-compatible and other terminals under Unix and VMS; available as dviware/dvitovdu

dvi2tty A DVI to ASCII conversion program, for normal terminals; available as dviware/dvi2tty

texsgi For SGI under Irix; both a binary and source are available, but be sure to get the fonts as well. Available as dviware/texsgi

H Support Packages for \TeX

44 Fig, a \TeX -friendly drawing package

(X)*Fig* is a menu driven tool that allows you to draw objects on the screen of an X workstation. *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 graphics/xfig and graphics/transfig

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

45 \TeX CAD, a drawing package for \LaTeX

\TeX 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 \specials (see question 31). \TeX CAD is part of the em \TeX distribution.

46 Spelling checkers for work with \TeX

For Unix, *ispell* is probably the program of choice. It is available in support/ispell; beware of any version 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. *amspell* can be called from within an editor (available as support/amspell). *jspell* is an extended version of *ispell* (available as 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 systems/mac/support/excalibur/Excalibur-2.5-sit.hqx (there are other dictionaries in the same directory).

For VMS, a spell checker can be found in support/vmspell

47 The $\text{VOR}\TeX$ package

$\text{VOR}\TeX$ (available in support/vortex) is a package of programs written at the University of California at Berkeley, and was described by Michael A. Harrison in “*News from the $\text{VOR}\TeX$ project*” in *TUGboat* 10(1), pp. 11–14, 1989. It includes several nice previewers and some *emacs* modes for \TeX and $\text{BIB}\TeX$. The $\text{VOR}\TeX$ distribution is not maintained, and now looks distinctly long in the tooth (it was never upgraded to \TeX version 3).

$\text{VOR}\TeX$ needed a separate workstation to run \TeX in the background; modern PCs for the home can provide more processor power (than was available to $\text{VOR}\TeX$) in a single box. This fact has been recognised by Blue Sky Research in their ‘Lightning Textures’ (which runs on a Macintosh in a somewhat similar way) and by TCI Software Research in ‘Scientific Word’ (see question 40), and is also the basis of many of the other environments mentioned in question 39.

I Literate programming

48 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 \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

\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 \TeX macro files, using *noweb*, has recently been made available in `web/tweb`

J Format conversions

50 Conversion between (\LaTeX) and others

troff *troff-to-latex* (available as `support/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 \TeX distribution (see question 34) also contains a program which converts *troff* to \TeX .

WordPerfect *wp2latex* (available as `support/wp2latex`) is a PC program written in Turbo Pascal by R. C. Houtepen at the Eindhoven University in the Netherlands. It converts *WordPerfect* 5.0 documents to \LaTeX . Pascal source is included. Users find it “helpful” and “decent” in spite of some limitations. It gets high marks for handling font changes, but cannot make indices, tables of contents, margins or graphics, and can’t handle features new in *WordPerfect* version 5.1, in particular the equation formatter. The program is copyrighted but free.

Glenn Geers of the University of Sydney, Australia (`glenn@qed.physics.su.oz.au`) is translating *wp2latex* into C and adding some *WordPerfect* 5.1 features, in particular its equation handling. His work is in the `glenn` subdirectory of `support/wp2latex`

PC-Write *pcwritex.arc*, from `support/pcwritex`, is a print driver for PC-Write that “prints” a PC-Write V2.71

by Peter Flynn at University College, Cork, Republic of Ireland.

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 `support/rnototex`

refer/tib There are a few programs for converting bibliographic data between $\text{BIB}\TeX$ and *refer/tib* formats. They are in `biblio/bibtex/utills/refer-tools`

In spite of the directory name, it also contains a shell script to convert $\text{BIB}\TeX$ to *refer* as well. The collection is not maintained.

RTF A program for converting Microsoft’s Rich Text Format to \TeX is available in `support/rtf2tex`, which was written and is maintained by Robert Lupton (`rh1@astro.princeton.edu`). There is also a converter to \LaTeX by Erwin Wechtl, in `support/rtf2latex`

Translation to RTF may be done (for a somewhat constrained set of \LaTeX documents) by \TeX 2RTF, which can produce ordinary RTF, Windows Help RTF (as well as HTML, see question 53). \TeX 2RTF is supported on various Unix platforms and under Windows 3.1; it is available from `support/tex2rtf`

Microsoft Word A rudimentary program for converting MS-Word to \LaTeX is `wd2latex`, for MS-DOS (`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 `comp.text.tex`, is available via `http://www.kfa-juelich.de/isr/1/texconv/texconv.html` and is archived as `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. *FrameMaker* provides “import filters” to aid translation from alien formats (presumably including \TeX) to *Framemaker*’s own.

51 Conversion from (\LaTeX) to plain ASCII

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

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

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.

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.

52 Conversion from SGML or HTML to T_EX

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. T_EX 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:
 - David Megginson's *sgmlspm*, written in Perl 5. Available from <http://www.perl.com/CPAN/modules/by-module/SGMLS>
 - Joachim Schrod and Christine Detig's *stil*, written in Common Lisp. Available from <ftp://ftp.th-darmstadt.de/pub/text/sgml/stil>

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 L^AT_EX file from it. The conversion code is subject to NCSA restrictions, but the whole source is available as `support/html2latex`

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).

T_EX is a typesetting language, not a markup system. With properly-used L^AT_EX, 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.

L^AT_EX2HTML (`support/latex2html`) is a package by Nikos Drakos (mostly of *perl* scripts) that breaks up a L^AT_EX 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 L^AT_EX intra-document references and hyperlinks, and extends the mechanisms to permit reference to other (possibly remote) documents and other Internet resources. It translates L^AT_EX 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.

L^AT_EX2HTML 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 L^AT_EX2HTML, and how to tailor it, in *TUGboat* 16(2).

There are two alternative strategies:

1. Free-standing L^AT_EX to HTML translations. Hard, but not impossible. Julian Smart's *latex2rtf* (available from `support/latex2rtf`) does a plausible job on a subset of L^AT_EX;
2. Writing an HTML-output backend in L^AT_EX itself. See Sebastian Rahtz' paper in *TUGboat* 16(3) for a discussion of how to go about this for the general case of SGML.

54 Making hypertext documents from T_EX

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

1. Try direct L^AT_EX conversion to HTML; see question 53;
2. Rewrite your document using Texinfo (see question 13), and convert that to HTML;
3. Look at Adobe Acrobat, an electronic delivery system guaranteed to preserve your typesetting perfectly. See question 55;
4. Invest in the hyperT_EX conventions (standardised `\special` commands); there are supporting macro packages for plain T_EX and L^AT_EX).

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

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

name: html:

end: html:

image: html:

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/hypertext/>; 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.

55 Making Acrobat documents from L^AT_EX

There are now two general routes to Acrobat output: Adobe's original 'distillation' route, and the use of PDF_TE_X (see question 120).

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

- distill 'ordinary' PostScript output of your document, or (on a Windows or Macintosh machine with *AcrobatExchange* installed) pass it through the PDFwriter in place of a printer, or
- process with PDF_TE_X, enabling Acrobat output.

(Note that the PDFwriter route is a dead end: it can only be used in this simple mode, as it cannot create hyperlinks.)

To translate all the L^AT_EX cross-referencing into Acrobat links, you need a L^AT_EX package to suitably re-define the internal commands. There are two of these for L^AT_EX 2_ε, both based on the Hyper_TE_X specification (see question 54): Sebastian Rahtz's *hyperref* (available from `macros/latex/contrib/supported/hyperref`), and Michael Mehlich's *hyper* (available from `macros/latex/contrib/supported/hyper`). *Hyperref* can operate using PDF_TE_X primitives rather than the hyper_TE_X conventions. You can use *dvips* or Y&Y's *DVIPSONE* to translate the DVI into PostScript acceptable to Distiller.

Sadly, there is no free implementation of all of Distiller's functionality, but *GhostScript* (version 4.00 onwards) provides some restricted distilling capability, and *Distiller* itself is now remarkably cheap (for academics at least).

Adobe's *AcrobatReader* is available for a very wide range of platforms. For those still omitted, *GhostScript* (versions 3.51 onwards) can display and print PDF files.

Work on a DVI to PDF translator is in progress, but shows no sign of immediate release (Sergey Lesenko spoke about the work at TUG'96 and again at TUG'97).

56 Getting METAFONT to do what you want

METAFONT allows you to create your own fonts, and most T_EX users will never need to use it. METAFONT, unlike T_EX, requires some customisation: each output device for which you will be generating fonts needs a mode associated with it. Modes are defined using the `mode_def` convention described on page 94 of *The METAFONTbook* (see question 19). You will need a file, which conventionally called `local.mf`, containing all the `mode_defs` you will be using. If `local.mf` doesn't already exist, Karl Berry's collection of modes, available as `fonts/modes/modes.mf`, is a good starting point (it can be used as a 'local.mf' without modification in a 'big enough' implementation of METAFONT). Lists of settings for various output devices are also published periodically in *TUGboat* (see question 16). Now create a plain base file using *inimf*, `plain.mf`, and `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 `plain.base` (or something similar; for example, it will be `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 METAFONT used).

Now you need to make sure METAFONT loads this new base when it starts up. If METAFONT loads the `plain` base by default on your system, then you're ready to go. Under Unix (using the default *web2c* distribution⁵) this does indeed happen, but we could for instance define a command *mf* which executes `virmf &plain` loading the `plain` base file.

The usual way to create a font with `plain` METAFONT is to start it with the line

```
\mode=<mode name>;mag=<magnification>;
input <font file name>
```

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

```
mf \mode=epson;mag=magstep 1;input cmr10
```

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

⁵The `command_name` is symbolically linked to `virmf`, and `virmf` loads `command_name.base`

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

```
% 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;
```

(note the absence of the `mode_def` and `enddef` commands), you would type

```
mf \smode="ln03"; input cmr10
```

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 METAFONT, is available as `info/metafont-for-beginners.tex`

57 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.nnnngf`⁶ 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.

58 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

`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 $\text{em}\TeX$ drivers construct the METAFONT commands). You might need to look at Karl Berry's collection of METAFONT modes (`fonts/modes/modes.mf`). 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 (see question 40 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).

L PostScript and \TeX

59 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 61. 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 63).

If you use $\text{\LaTeX} 2_{\epsilon}$, 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 $\text{\LaTeX} 3$ 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 61) and font description (`.fd`) files for each font family you want to use. These can be obtained from `fonts/psfonts`, arranged by vendor (*e.g.*, Adobe, Monotype, *etc.*). For convenience, metrics for the common '35' PostScript

⁶Note that the file name may be transmuted by such operating systems as MS-DOS, which don't permit long file names

aged as macros/latex/packages/psnfss/lw35nfss.zip. For older versions of L^AT_EX there are various schemes, of which the simplest to use is probably the PSL^AT_EX macros distributed with *dvips*.

For plain T_EX, 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 L^AT_EX's NFSS, and so is easy to use with PostScript fonts.

Some common problems encountered are discussed elsewhere (see question 62).

60 Previewing files using PostScript fonts

Most T_EX 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 40 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).

61 T_EX 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 (T_EX Font Metric) form. The CTAN archives have pre-built 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.

the T_EX 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 (see question 30) to *pretend* to T_EX that the encoding is the same as it is used to. If you use L^AT_EX 2_ε, it allows for changing the encoding in T_EX; read the *L^AT_EX Companion* (see question 19) 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': see question 33) encoding.

Alan Jeffrey's *fontinst* package (fonts/utilities/fontinst) is an AFM to TFM converter written in T_EX; it is used to generate the files used by L^AT_EX 2_ε'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 (see question 30).

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 L^AT_EX 2_ε 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 40) Y&Y's Font Manipulation Tools package which includes a powerful *afmtotfm* program among many other goodies.

62 Problems using PostScript fonts

For the typical L^AT_EX user trying to use the PSNFSS (see question 59) 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/ps2pk) 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 emT_EX 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

hyphenation or overfull boxes than CMR produces; but CMR is extremely generous.

63 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 40) 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 seriffed, sans serif, sans serif fixed width, calligraphic, 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 others, it also includes the AMS *msam* and *msbm* symbol sets). The planned 'Lucida Bright Expert' (14 fonts) adds seriffed 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 2.09. Support under \LaTeX 2 ϵ is provided in PSNFSS (see question 59) thanks to Sebastian Rahtz.

MathTime 1.1 (3 fonts) \TeX plorators (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 2 ϵ is

tian 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 2 ϵ in PSNFSS (see question 59) thanks to Alan Jeffrey and Sebastian Rahtz.

(A similar development by Thomas Esser, using the Adobe Palatino fonts, is available from `systems/unix/teTeX/updates/texmf/mathppl.sh`)

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 40 for details). The MathTime fonts are also available from:

\TeX plorators
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

for 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.

64 Including a PostScript figure in L^AT_EX

L^AT_EX 2_ε (see question 116) has a standard package for graphics inclusion, rotation, colour, and other driver-related features. The package is documented in the second edition of the Lamport’s L^AT_EX book (see question 19), and is available in `macros/latex/required/graphics`

If you don’t use L^AT_EX 2_ε, perhaps the best method is to use the `psfig` macros written by Trevor Darrell, available in `graphics/psfig`

You will also need a DVI to PostScript conversion program that supports the `\specials`. The drivers mentioned in question 41 do, and come with a version of `psfig` ready to use with them. The `psfig` macros work best with Encapsulated PostScript Files (EPS). In particular, `psfig` will need the file to have a BoundingBox (see Appendix H of the *PostScript Language Reference Manual*). If you don’t have an EPS file, life can be difficult.

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. As a result, most DVI previewers will simply show the blank space T_EX has reserved for your figure, not the figure itself.

There are two rather good documents on CTAN addressing of figure production with rather different emphases. Anil K. Goel’s, `info/figsinltx.ps` covers the different ways in which you might generate figures, and one the old (L^AT_EX 2.09) ways of including them into documents. Keith Reckdahl’s, `info/epslatex.ps`, covers the standard L^AT_EX 2_ε facilities, as well as some of the supporting packages, notably `subfigure` (`macros/latex/contrib/supported/subfigure`) and `psfrag` (`macros/latex/contrib/supported/psfrag`).

M Special sorts of typesetting

65 Drawing with T_EX

There are many packages to do pictures in (L^A)T_EX itself (rather than importing graphics created externally), ranging from simple use of L^AT_EX `picture` environment, through enhancements like `epic`, to sophisticated (but slow) drawing with P_TC_TE_X. 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 T_EX itself, by sophisticated use of `\specials`. You need a decent DVI to PostScript driver (like `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.

with font files? Try Metapost (see question 4) — all the power of METAFONT, but it generates PostScript figures. Knuth uses it for all his work. . .

3. `Mfpic`; you liked METAFONT, but can’t understand the language? The package (`graphics/mfpic`) makes up METAFONT code for you within using familiar-looking T_EX macros. Not *quite* the full power of METAFONT, but a friendlier interface.
4. You liked P_TC_TE_X but don’t have enough memory or time? Look at Eitan Gurari’s `macros/generic/dratex`, which is as powerful as most other T_EX drawing packages, but is an entirely new implementation, which is not as hard on memory, is much more readable (and is fully documented).

66 Double-spaced documents in L^AT_EX

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?

L^AT_EX 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 (*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 `\baselinestretch: \renewcommand{\baselinestretch}{1.2}` may be enough to give officials the impression you’ve kept to their regulations. Don’t try changing `\baselineskip`: its value is reset at any size-changing command.
- Alternatively, use a line-spacing package. The best for current is `macros/latex/contrib/supported/setspace/setspace.sty`

67 Formatting a thesis in L^AT_EX

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 66. If you want to write your own, a good place to start is the University of California style (available as `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!)

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` `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:

```
\parpic(width,height)(x-off,y-off)[Options][Position]{Picture}
```

Paragraph text

All parameters except the *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 `macros/latex209/contrib/picins/picins.txt`

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

```
\begin{floatingfigure}[options]{width of figure}  
figure contents  
\end{floatingfigure}
```

There is a (more or less similar) `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 `multicol` package, but doesn't work well in the neighbourhood of list environments (unless you change your \LaTeX document).

`wrapfig` `macros/latex/contrib/other/misc/wrapfig.sty` has syntax:

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

The syntax of the `wraptable` 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,etc.}` parameter can also be specified as *i*(*n*side) or *o*(*u*tside) 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 `\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.

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 (see question 98), by enabling you to define different page styles for initial and for body pages.

70 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.

71 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 70) 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.

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 ones for MS-DOS are part of the $\em\TeX$ and $g\TeX$ distributions (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 (see question 11), and \TeX sis, `nonfree/macros/texsis` (whose macros, `nonfree/macros/texsis/index/index.tex`, can be used independently with plain).

idxTeX for \LaTeX under VMS. Available (together with a glossary-maker called `glotex`) in `indexing/glotexidxTeX`

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 (see question 13).

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.iti.informatik.th-darmstadt.de/xindy/` since the distribution contains several different implementations.

73 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

'robust' commands (see question 110) 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).

74 Citing URLs with $\BIB\TeX$

There is no citation type for URLs, *per se*, in the standard $\BIB\TeX$ styles, though Oren Patashnik (the author of $\BIB\TeX$) is considering developing one such for use with the long-awaited $\BIB\TeX$ 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 (see question 73) 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. $\BIB\TeX$ has a habit of splitting lines it considers excessively long, and if there are no space characters for it to use as 'natural' breakpoints, $\BIB\TeX$ 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 $\BIB\TeX$ to make reasonable decisions about breaking the line.

75 Using $\BIB\TeX$ with plain \TeX

The file `macros/eplain/btxmac.tex` contains macros and documentation for using $\BIB\TeX$ with plain \TeX , either directly or with Eplain (see question 11). See question 22 for more information about $\BIB\TeX$ itself.

A powerful package which allows the typesetting of polyphonic and other multiple-stave music is MusicT_EX, written by Daniel Taupin (taupin@rsovax.lps.u-psud.fr). It is available in 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 MusicT_EX, known as MusiXT_EX. MusiXT_EX 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 T_EX-based mechanism that MusicT_EX uses. Daniel Taupin and Andreas Egler are pursuing distinct versions of MusiXT_EX; they are available, respectively, from macros/musix_{tex}/taupin and macros/musix_{tex}/egler

Digital music fans can typeset notation for their efforts by using *midi2tex*, which translates MIDI data files into MusicT_EX source code. It is available from support/midi2_{tex}

A rather simpler notation than MusicT_EX is supported by *abc2mtex*; this is a package designed to notate tunes stored in an ASCII format (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 support/abc2_{mtex}

There is a mailing list for discussion of typesetting music in T_EX. To subscribe, send mail to mutex-request@stolaf.edu containing the word ‘subscribe’ in the body.

77 Drawing Feynman diagrams in L^AT_EX

Michael Levine’s macro package for drawing Feynman diagrams in L^AT_EX is available in macros/latex209/contrib/feynman

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

N How do I do X in T_EX or L^AT_EX

78 Proof environment

It is not possible to make a proof environment which automatically includes an ‘end-of-proof’ symbol. Some proofs end in displayed maths; others do not. If the input file contains `... \] \end{proof}` then L^AT_EX 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.

It is a good idea to have commands such as `\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 `\R`, `\C`, *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 T_EX distribution. The fonts are available in fonts/ams/amsfonts/sources/symbols

Two files which load the fonts and define the symbols are provided, and both work with either T_EX or L^AT_EX. 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 METAFONT (in 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:

```
\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}}
```

work well at normal size if the surrounding text is 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.

80 Roman theorems

If you want to take advantage of the powerful `\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 *theorem.sty* (part of macros/latex/required/tools). Alternatively, the following sets up an environment `remark` whose content is in roman.

```
\newtheorem{preremark}{Remark}
\newenvironment{remark}%
{\begin{preremark}\rm}{\end{preremark}}
```

This will not work if you are using NFSS (see question 115) outside of L^AT_EX 2_ε (see question 116), because the command `\rm` behaves differently there.

81 Fancy enumeration lists

Suppose you want your top-level enumerates to be labelled ‘I’, ‘II’, ..., then give these commands:

The possible styles of numbering are given in Section 6.3 of Lamport's book (see question 19). Both `\theenumi` and `\labelenumi` must be changed, since `\theenumi` is used in cross-references to the list.

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

If you're running $\LaTeX 2_{\epsilon}$, the package `enumerate.sty` (part of `macros/latex/required/tools`) offers similar facilities. With `enumerate.sty`, the example above would be achieved simply by starting the enumeration `\begin{enumerate}[I/]`.

82 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 `secnumdepth` described in Appendix C of the \LaTeX manual. For example:

```
\setcounter{secnumdepth}{-1}
\chapter{Preface}
```

Of course, you have to set `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. `\chapter` without the star does

1. put something in the `.toc` file;
2. if `secnumdepth` ≥ 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.

83 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 `minipage` environment and to put the notes underneath the table, or to use Donald Arseneau's package `macros/latex/contrib/other/misc/threeparttable.sty`

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

1. Use `\footnotemark` to position the little marker appropriately, and then put in `\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.

all the ugliness of footnotes in a minipage with no extra effort.

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

5. Grab hold of `footnote.sty` from CTAN, lurking in `macros/latex/contrib/supported/mdwtools`

Then put your tabular environment inside a `savenotes` environment. Alternatively, say `\makesavenoteenv{tabular}` in the preamble of your document, and tables will all handle footnotes correctly.

6. Use `mdwtab.sty` from the same directory (`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.

84 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 *The \LaTeX Companion* (see question 19; the programming itself is discussed in question 104). The following hack will probably satisfy your editor. Define yourself new commands

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

and then use `\ssection` and `\ssubsection` in place of `\section` and `\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 NFSS (see question 115) outside of $\LaTeX 2_{\epsilon}$ (see question 116), because the font-changing commands behave differently there.

85 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.

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 footnote}}`
- Use the macros `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.

87 Changing the margins in L^AT_EX

Don't do it. Learn some L^AT_EX, 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. L^Ampport'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 L^AT_EX is to get macros `macros/latex209/contrib/misc/fullpage.sty`, which sets the margins of the page identical to those of plain T_EX, *i.e.*, 1-inch margins at all four sides of the paper. It also contains an adjustment for A4 paper.

Somewhat more flexible is macros `macros/latex/contrib/other/misc/vmargin.sty`, which has a canned set of paper sizes (a superset of that provided in L^AT_EX 2_ε), provision for custom paper, margin adjustments and provision for two-sided printing.

For details of L^AT_EX's page parameters, see section C.5.3 of the L^AT_EX 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, *i.e.*, `\evensidemargin`, `\oddsidemargin`, `\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 L^Ampport's figure C.3. They should only be changed in the preamble of the document, *i.e.*, before the `\begin{document}` statement. To adjust the margins within a document we define an environment:

```
\begin{list}{-}{%
\setlength{\topsep}{0pt}%
\setlength{\leftmargin}{#1}%
\setlength{\rightmargin}{#2}%
\setlength{\listparindent}{\parindent}%
\setlength{\itemindent}{\parindent}%
\setlength{\parsep}{\parskip}%
}%
\item[]{\end{list}}
```

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 `\begin{changemargin}{-1cm}{-1cm}` narrows the left and right margins by 1cm.

88 Finding the width of a letter, word, or phrase

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

```
\newdimen\stringwidth
\setbox0=\hbox{hi}
\stringwidth=\wd0
```

Note that if the quantity in the `\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 L^AT_EX by:

```
\newlength{\gnat}
\settowidth{\gnat}{\textbf{small}}
```

This sets the value of the length command `\gnat` to the width of "small" in bold-face text.

89 Excluding blocks of text from the DVI file

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

A more general environment for doing the job is Victor Eijkhout's `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 macros `macros/latex209/contrib/misc/comment.sty`

90 Setting bold Greek letters in L^AT_EX

The simple solution (`\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, `\mathbf` can be used for upper-case Greek letters in ordinary circumstances, but the \mathcal{AMS} -L^AT_EX package `amsmath.sty` disables this font-switching and you must use one of the techniques outlined below.

The plain T_EX solution *does* work, in a limited way:

```
{\boldmath$\theta$}
```

tion' requires arcana such as:

```
$... \mbox{\boldmath$\theta$} ...$
```

which then causes problems in superscripts, etc.

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

- The package `bm.sty`, which is part of the \LaTeX tools distribution (`macros/latex/required/tools`), defines a command `\bm` which may be used anywhere in maths mode.
- The package `amsbsy.sty`, which is part of $\mathcal{A}\mathcal{M}\mathcal{S}$ - \LaTeX (`macros/latex/required/amslatex`) defines a command `\boldsymbol`, which (though slightly less comprehensive than `\bm`) covers almost all common cases.

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

91 Defining a new log-like function in \LaTeX

Use the `\mathop` command, as in:

```
\newcommand{\diag}{\mathop{\rm diag}}
```

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

```
\newcommand{\diag}{\mathop{\rm diag}\nolimits}
```

This works in \LaTeX 2.09 and in \LaTeX 2 ϵ , but not under NFSS alone (see question 100). However, the canonical method for doing this in \LaTeX 2 ϵ is to use the `\DeclareMathOperator` command of `amsopn.sty` (which is part of the $\mathcal{A}\mathcal{M}\mathcal{S}$ - \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.)

92 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 logos such as $\mathcal{A}\mathcal{M}\mathcal{S}$ - \TeX , $\text{P}\mathcal{C}\mathcal{T}\mathcal{E}\mathcal{X}$, $\text{B}\mathcal{I}\mathcal{B}\mathcal{T}\mathcal{E}\mathcal{X}$, and so on. Lamport invented \LaTeX , and marketing input led to the current logo \LaTeX 2 ϵ .

The common people don’t have to follow this stuff wherever it goes, but, for those who insist, a large collection of logos is defined in `macros/eplain/texnames.sty`; the METAFONT logo can be set in fonts that \LaTeX 2 ϵ knows about (so that it scales with the surrounding text) using the package `macros/latex/contrib/supported/mflogo`

For those who don’t wish to acquire the ‘proper’ logos, the canonical thing to do is to say `AMS-\TeX{}` ($\mathcal{A}\mathcal{M}\mathcal{S}$ - \TeX) for `\mathcal{A}\mathcal{M}\mathcal{S}\TeX{}`, `\text{P}\mathcal{C}\mathcal{T}\mathcal{E}\mathcal{X}` for $\text{P}\mathcal{C}\mathcal{T}\mathcal{E}\mathcal{X}$, `\text{B}\mathcal{I}\mathcal{B}\mathcal{T}\mathcal{E}\mathcal{X}` for $\text{B}\mathcal{I}\mathcal{B}\mathcal{T}\mathcal{E}\mathcal{X}$, and so on.

93 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 2 ϵ ; 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/distrib/latex209.tar` and ought to solve this problem.

If you’re using \LaTeX 2 ϵ , 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 (see question 33), which introduced an alternative hyphen character. The \LaTeX 2 ϵ 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 \LaTeX 2 ϵ do *not* use the alternative hyphen character, thus removing this source of problems.

94 (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`

95 Accented words aren’t hyphenated

\TeX ’s algorithm for hyphenation gives up when it encounters an `\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 (see question 33) which contain accented letters as single glyphs. In the future, perhaps, Omega (see question 118) will provide a rather different solution.

People sometimes get messages saying ‘memory capacity exceeded’. 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. In the purest form, you change the parameters in module 11 of the WEB source. In less pure forms, you might need to modify a change file, or perhaps change some environment variables; $\epsilon\TeX$ allows you to adjust the memory allocation criteria on the command line. Consult the documentation that came with your implementation.

97 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 $\LaTeX 2_{\epsilon}$ `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).

`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.

98 `\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/supported/fancyhdr`
- Use `nopageno.sty`, which suppresses this behaviour. It is available in `macros/latex/contrib/supported/carlisle/nopageno.sty`

99 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 `\emph` does just that to its argument.

Nevertheless, typographically illiterate people (such as those that specify double-spaced thesis styles — see question 66) continue to require underlining of us, so \LaTeX as distributed defines an `\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 `\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.)

If commands such as `\rm` and `\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 NFSS (see question 115). Complain loudly; ask your system administrator to replace this version with \LaTeX 2 ϵ (see question 116), in which commands such as `\rm` and `\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 `oldfont.sty`, which should have been installed at the same time as NFSS.

101 Old \LaTeX font references such as `\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 `cmr` could be found as `\fivrm`, `\sixrm`, `\sevrn`, `\egtrn`, `\ninrm`, `\tenrm`, `\elvrn`, `\twlrm`, `\frtrn`, `\svtrn`, `\twtyrm` and `\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 2 ϵ ; to use the unconverted \LaTeX 2.09 packages under \LaTeX 2 ϵ , you need also to include the package `rawfonts.sty` (which is part of the \LaTeX 2 ϵ distribution).

102 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 NFSS (see question 115) or \LaTeX 2 ϵ (see question 116). In the former case, use `oldfont.sty`, as in the question 100. In the latter, use the package `latexsym`, which is part of the \LaTeX 2 ϵ distribution, or the package `amsfonts`, if it is available.

103 \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{figure}           \begin{figure}
\caption{A Figure} or  \caption{A Figure%
\label{fig}           \label{fig}}
\end{figure}           \end{figure}
```

104 `\@` 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 ‘`\spacefactor`’ in vertical mode”, but others occur.

Such problems are usually caused by including a \LaTeX 2 ϵ 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

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 `\@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 `\makeatletter` and `\makeatother`. So the definition of `\subsection` on page 26 could be:

```
\makeatletter
\renewcommand{\subsection}{\@startsection
  {subsection}%                % name
  ...
  {\normalfont\normalsize\itshape}}% style
\makeatother
```

105 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’ 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/sources/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 (see question 30) to map the old to the new series; it's available in `fonts/vf-files/msx2msa`

106 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)⁷. 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

⁷The 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

the output should be almost correct.

107 ‘String too long’ in BIB_TE_X

The BIB_TE_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 `output.nonnull` 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 `output.nonnull` 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_TE_X’s limit, but you may need to retain the entry because it will be included in the typeset document. In such cases, put the body of the entry in a separate file:

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

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

P Why does it *do* that?

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

The original L^AT_EX 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 L^AT_EX, 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

this as an error.

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

- Why does L^AT_EX throw a page before and after `\include` commands?
- Why can’t I nest `\included` files? — I always used to be able to under L^AT_EX 2.09.

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.

Answer: in fact, you couldn’t, even under L^AT_EX 2.09, but the failure wasn’t diagnosed. However, since you were happy with the behaviour under L^AT_EX 2.09, replace the `\include` commands with `\input` commands (with `\clearpage` as appropriate).

109 Why does it ignore paragraph parameters?

When T_EX 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 L^AT_EX:

```
{\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...

T_EX 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 `\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:

```
end the group.\par}
Here’s more that needn’t be ragged...
```

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, L^AT_EX already defines an appropriate one:

```
\begin{flushleft}
Here’s text to be ranged left...
\end{flushleft}
```

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, *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 L^amport's book (see question 19).

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 `\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 2_ε, 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.

111 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

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

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` (see question 110).

112 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

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:

```
\newcommand{\mytable}{\begin{tabular}...
\end{tabular}}
\section{A section title \protect\mytable{
with a table}}
```

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

113 Why are # signs doubled in macros?

The way to think of this is that `##` gets replaced by `#` in just the same way that `#1` gets replaced by 'whatever is the first argument'.

So if you define a macro and use it as:

```
\def\aa#1{+++#1+++#1+++#1+++} \a{b}
```

the macro expansion produces '`+++b+++b+++b+++`', which people find normal. However, if we now replace part of the macro:

```
\def\aa#1{+++#1+++\def\x #1{xxx#1}}
```

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

```
\def\aa#1{+++#1+++\def\x ##1{xxx##1}}
```

and `\a{b}` will expand to '`+++b+++b+++b+++`', because `#1` gets replaced by 'b' and `##` gets replaced by #.

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

pages?

L^AT_EX 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. L^AT_EX 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 L^AT_EX's perception of its priorities: they're controlled by `\interfootnotelinepenalty` — the larger it is, the less willing L^AT_EX is to split footnotes.

Setting

```
\interfootnotelinepenalty=10000
```

inhibits split footnotes altogether, which will cause '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 T_EX 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.

Q Recent Developments

115 The New Font Selection Scheme (NFSS)

NFSS was an extension to L^AT_EX written by Frank Mittelbach and Rainer Schöpf. It is described in *TUGboat* 10(2). In traditional typesetting, fonts are described by four parameters: the *family* (e.g., computer modern), the *series* (i.e., the weight and width of the font, such as light or bold), the *shape* (e.g., italic), and the *size*. NFSS is a mechanism allowing the user to change any of these independently. NFSS makes it relatively easy to use nonstandard fonts such as the PostScript ones with L^AT_EX, and easy to change maths fonts. It also allows dynamic loading of fonts at runtime (i.e., not when the format file is created).

With the demise of L^AT_EX 2.09 as supported software, the label 'NFSS' has become somewhat misleading, as there's no 'old' scheme with which to contrast it — L^AT_EX has incorporated the NFSS.

116 L^AT_EX 2_ε (the 'new' standard L^AT_EX)

L^AT_EX 2_ε is the version of L^AT_EX prepared and supported by the L^AT_EX3 project team (see question 117). With the advent of L^AT_EX 2_ε, users gained a formal support structure, and the knowledge that bug fixes would be incorporated into the

approximately 6-month intervals).

L^AT_EX 2_ε is so structured that significant enhancements may be added using macro add-ons, rather than by the different T_EX formats that were the bane of the L^AT_EX 2.09 world.

L^AT_EX 2_ε is upwardly compatible with L^AT_EX 2.09, but has new features. In the latest (December 1998) release, these include:

- NFSS (see question 115) is part of the distribution.
- S^LT_EX 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 (see question 110); 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 now a standard package for colour and graphics inclusion.
- There is standard support for typesetting in cyrillic.

117 The L^AT_EX3 project

The L^AT_EX3 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 L^AT_EX. It will remain freely available and it will be fully documented at all levels.

The L^AT_EX3 team's first product, L^AT_EX 2_ε (see question 116), was delivered in 1994, and the team undertakes its maintenance.

118 The Omega project

Omega (Ω) is a program built as an extension of the T_EX 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 teT_EX, mikT_EX, fpT_EX and CMacT_EX distributions (QrefT_EX-systems).

The $\mathcal{N}\mathcal{T}\mathcal{S}$ 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 15), and second, even if they *were* allowed to develop the program, some members of the $\mathcal{N}\mathcal{T}\mathcal{S}$ 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 $\varepsilon\text{-}\text{\TeX}$, 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 $\varepsilon\text{-}\text{\TeX}$ was made available in November 1996. It is available via `systems/e-tex`; $\varepsilon\text{-}\text{\TeX}$ is also distributed on the \TeX Live CD-ROM (see question 37).

120 The PDF \TeX project

PDF \TeX (formerly known as $\text{\TeX}2\text{PDF}$) 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. PDF \TeX 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 37).

121 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 , \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 $\text{\TeX}book$), 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 (see question 16).

If you've found a bug in $\text{\LaTeX} 2_{\varepsilon}$, look in the bugs database to see if it's already been reported. If not you should submit details of the bug to the $\text{\LaTeX}3$ team. To do this, you should process the file `latexbug.tex` with \LaTeX (the file is part of the $\text{\LaTeX} 2_{\varepsilon}$ 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 $\text{\LaTeX} 2.09$, 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 \TeX -consultant — TUG maintains a register of \TeX consultants (see <http://www.tug.org/consultants.html>).