The New TeX FAQ
Your 111 Questions Answered
version 2.0i, date 1997/01/02

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

April 15, 1997

NOTE

This document is a version of the FAQ article that was published as the December 1994 edition of the UKTUG magazine *Baskerville* (which wasn't formatted like this). The intention is that its content should be available to be fed back into the normal `comp.text.tex` FAQ posting some time soon, but that work isn't yet complete. The article is also to be available as a WWW page.

Contents

A Introduction

This article was prepared by the Committee of the UK TEX Users Group (UK TUG)[1] as a development of a regular posting to the *Usenet* newsgroup `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 needs[2]; a newsgroup is an area within Usenet carrying a particular class of articles. Since a common sort of article asks for help, advice or information, and since certain of these questions are regularly repeated (often with monotonous regularity), some public-spirited souls took to writing articles which listed "Frequently Asked Questions" and answers to them. Many members of UK TUG do not have access to Usenet, but could be expected to value the answers about TEX that have accumulated over the years; so we decided to update the list and publish it in *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, and it can be compiled by anyone who runs a production LATEX 2$_\varepsilon$ (see question 107), and has the required fonts. It is the committee's hope that it will also be possible for the content of this article to

[1]For 1996–97: Peter Abbott, Kaveh Bazargan, David Carlisle, Malcolm Clark, Robin Fairbairns, Hewlett, Alan Jeffrey and Sebastian Rahtz

[2]Usenet, as its name implies, is a means of using some sort of network; in the earliest days the network was made by stringing together a series of telephone lines, but nowadays Usenet is most often carried over the Internet

feed back to the world-wide TEX community via Bobby's regular posting.

In addition, a translation of the article is available on the World-Wide Web, via URL `http://www.cogs.susx.ac.uk/cgi-bin/texfaq2html?introduction=yes`[3]

We have rearranged Bobby's article quite a lot, and have 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. We have also pruned it to take account of the changes that have happened in the world of TEX since Bobby first started.

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

Further, Rosemary Bailey, Jonathan Fine and Chris Rowley were members of the committee during the period 1993–95, during which the entreprise of developing this FAQ was conceived and its first version was published, 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 32 gives details of the CTAN archives, and how to retrieve files from them. If you don't have access to the Internet, question 35 tells you of sources of CD-ROMs that offer snapshots of the archives.

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

To avoid confusion, we've also elided the full stop[4] 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 17) 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.

TEX is a macro processor, and offers its users a powerful 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 META O T?

The META O T system implements a picture-drawing language very much like that of METAFONT except that it outputs PostScript commands instead of run-length-encoded bitmaps. META O T 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.)

[3] This is a temporary URL; a final home for the document is to be provided in due course

[4] 'Full stop' (British English)=='period' (American English)

Much of META O T's source code was copied from METAFONT's sources with Knuth's permission.

5 What is LaTeX?

LaTeX is a TeX macro package, originally written by Leslie Lamport, that provides a document processing system. LaTeX 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 LaTeX "*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: LaTeX *can* meet most user requirements, but finding out *how* is often tricky.

6 How should I pronounce "LaTeX(2$_\varepsilon$)"?

Lamport never recommended how one should pronounce LaTeX, but a lot of people pronounce it 'Lay TeX' or perhaps 'Lah TeX' (with TeX pronounced as the program itself; see question 2).

The 'epsilon' in 'LaTeX 2$_\varepsilon$' is supposed to be suggestive of a small improvement over the old LaTeX 2.09. Nevertheless, most people pronounce the name as 'LaTeX-two-ee'.

7 Should I use `plain` TeX or LaTeX?

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

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

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

8 What are the AMS packages (AMS-TeX, *etc*.)?

AMS-TeX is a TeX macro package, originally written by Michael Spivak for the American Mathematical Society (AMS) during 1983–1985. It is described in "*The Joy of TeX*" by Michael D. Spivak (second edition, AMS, 1990, ISBN 0-821-82997-1). It is based on `plain` TeX, 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 TeX runs), and other things.

As LaTeX increased in popularity, authors asked to submit papers to the AMS in LaTeX, and so the AMS developed AMS-LaTeX, which is a collection of LaTeX packages and classes that offer authors most of the functionality of AMS-TeX.

9 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 11). 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.

10 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 17). 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).

11 What is Texinfo?

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

The Texinfo distribution, including a set of TEX macros for formatting Texinfo files is available as `macros/texinfo/texinfo-3.9.tar.gz` (also available as a `.zip` file `macros/texinfo/texinfo-3.9.zip`).

12 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 38 lists the major vendors.

13 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 108) 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 109 and 110.

14 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 microcomputer software on disk. TUG has a Technical Council to coordinate TEX-related developments (see question 16). Enquiries should be directed to:

TEX Users Group
1850 Union Street, #1637
San Francisco CA 94123
USA

Tel: (+1) 805-963-1338
Fax: (+1) 805-963-8358
Email: `tug@tug.org`
Web: `http://www.tug.org/`
CTAN details: `usergrps/tug`

15 Are there nationally-based user groups, too?

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

DANTE, Deutschsprachige Anwendervereinigung TEX e.V.
Postfach 10 18 40
D-69008 Heidelberg
Germany

Tel: (+49) 06221 2 97 66
Fax: (+49) 06221 16 79 06
Email: `dante@dante.de`
Web: `http://www.dante.de/`
CTAN details: `usergrps/dante`

Association GUTenberg,
BP 10,
93220 Gagny principal,
France

Email: `gut@irisa.fr`
Web: `http://www.ens.fr/gut/`
CTAN details: `usergrps/gut`

NTG
Postbus 394, 1740AJ Schagen,
The Netherlands

Email: `ntg@nic.surfnet.nl`
Web: `http://ei0.ei.ele.tue.nl/ntg/ntg.html`
(note that this is a temporary address)
CTAN details: `usergrps/ntg`

UK TEX 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://www.tex.ac.uk/UKTUG/home.html`
CTAN details: `usergrps/uktug`

A listing of all known groups is available as `usergrps/info/usergrps.tex`

16 TUG Technical Working Groups

TUG (see question 14) has an autonomous Technical Council which oversees a number of working groups on areas of common interest to the TeX community. The Council has three members (current chair is Michael Ferguson, assisted by Yannis Haralambous and Sebastian Rahtz), who liaise with chair people of each working group. Each group establishes its own working methods and membership, and anyone interested in taking part should contact the chair. Suggestions for new groups should be addressed to Michael Ferguson (`mike@inrs-telecom.uquebec.ca`).

A brief list of the active groups follows:

WG-92-00 (IRP-TWG) *Independent Research Project TWG.*
To recognise and report to the TeX Board and the TeX Community on important projects which are independent of TUG but are of concern to the entire TeX Community.

Contact: Alan Hoenig (`ajhjj@cunyvm.cuny.edu`)

WG-92-01 *TeX Extended Mathematics Font Encoding.*
To create font encoding standards for Mathematical fonts used in TeX systems.

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

WG-92-03 *Multiple Language Coordination.*
The primary purpose of this working group is to obtain, for TeX systems, a consistent means for implementing, accessing, and describing, the fonts, ligature rules, hyphenation patterns and other special requirements for a given linguistic group.

Contact: Yannis Haralambous (`Yannis.Haralambous@univ-lille1.fr`)

WG-92-04 *TeX for the Disabled.*
The primary purpose of this working group is as a forum for those people interested in using and/or enhancing TeX to serve the needs of those with visual and other disabilities.

Contact: T.V. Raman (`raman@adobe.com`)

WG-92-05 *TeX Archive Guidelines.*
The purpose of this Technical Working Group is to develop guidelines for the effective management and utilisation of major TeX archives, and to initiate communication among the maintainers of the existing archives for the purpose of coordination and synchronisation.

Contact: Sebastian Rahtz (`s.rahtz@elsevier.co.uk`)

WG-94-07 *TeX Directory Structures.*
The primary purpose of this TWG is to identify a universal directory structure for macros, fonts and other related TeX software so that recommendations can be made to all suppliers of TeX software.

The group's current set of proposals are to be found on CTAN at `tds/draft-standard`

Contact: Karl Berry (`kb@cs.umb.edu`)

WG-94-08 *DVI Driver Implementation and Standardisation Issues.*
The major objective shall be 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-94-09 *TeX and SGML.*
The major objective is to investigate the requirements and difficulties in developing an interface technology for TeX and SGML.

Contact: Ken Dreyhaupt (`kend@springer-ny.com`)

WG-94-10 *TeX and Linguistics.*
The main goal is to study and discuss the requirements for typesetting linguistics in TeX and as a means of identifying, examining, testing, and comparing macros, fonts, style files and other aids for typesetting linguistics.

Contact: Christina Thiele (`cthiele@ccs.carleton.ca`)

C Documentation and Help

17 Books on TeX and its relations

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

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

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

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

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

TeX by Topic by Victor Eijkhout (Addison-Wesley, 1992, ISBN 0-201-56882-9)

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 $\boxed{1}$[5] 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 2ε Helmut Kopka and Patrick W. Daly (second edition, Addison-Wesley, 1995, ISBN 0-201-42777-X)

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

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)

Math into TEX: A Simplified Introduction using AMS-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 AMS-LATEX by George Grätzer (Birkhäuser, 1996, ISBN 0-817-63805-9)

Of the list, Lamport's, Goossens, Mittelbach and Samarin's, Kopka and Daly's, and Grätzer's "*Math into LATEX*" cover LATEX 2ε. 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)

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.

18 Where to find this article

Bodenheimer's article, from which the present one was developed, is posted (nominally monthly) to newsgroup `comp.text.tex` and cross-posted to newsgroups `news.answers` and `comp.answers`. The most recently posted copy of Bodenheimer's article is kept on CTAN in `help/TeX-FAQ`; it is also archived at any site that archives `news.answers`, such as `rtfm.mit.edu` (`18.181.0.24`), and the article is available there via anonymous `ftp` (in the directory `pub/usenet/news.answers/tex-faq`). If you have access to email, but not to `ftp`, use the *ftpmail* server (see 32).

A version of the present article may be browsed via the World-Wide Web, at URL `http://www.cogs.susx.ac.uk/cgi-bin/texfaq2html?introduction=yes`[6]

19 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.

The mailing list `info-tex` offers a mail analogue of the Usenet group `comp.text.tex`; mail to the list us automatically submitted to the newsgroup, and thus answers to questions may be given by people who only read the newsgroup. Subscribe to the list by sending a message '`subscribe info-tex <your name>`' to `listserv@shsu.edu`

The (rather high volume of) postings to `comp.text.tex` may be had in digested form through the mailing list `ctt-digest`. Subscribe to the list by sending a message '`subscribe ctt-Digest <your name>`' to `listserv@shsu.edu`

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-O T) 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.

The literate programming newsgroup (see question 46) `comp.programming.literate` is gatewayed to the `litprog` mailing list; subscribe by sending a message '`subscribe litprog <your name>`' to `listserv@shsu.edu`.

[5] That's 'Starting from Square One'
[6] This is a temporary URL; a final home for the document is to be provided in due course

20 BIBTEX Documentation

BIBTEX, a program originally designed to produce bibliographies in conjunction with LATEX, is explained in Section 4.3 and Appendix B of Leslie Lamport's LATEX manual (see question 17). The document "BIBTEXing", contained in the file `btxdoc.tex`, gives a more complete description. *The LATEX Companion* (see question 17) also has information on BIBTEX and writing BIBTEX style files.

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

There is a Unix BIBTEX *man* page in the *web2c* package (see question 36). Any copy you may find of a *man* page written in 1985 (before "BIBTEXing" and "Designing BIBTEX Styles" appeared) is obsolete, and should be thrown away.

21 The PICTEX manual

PICTEX is a set of macros for drawing diagrams and pictures. The macros are freely available in `graphics/pictex`; however, the PICTEX manual itself is not free. It is available for $30 ($35 with the disk) from the TEX Users Group (see question 14). The proceeds from the sales go to Michael Wichura, the author of PICTEX, and to TUG.

22 Finding (LA)TEX macro packages

Before you ask for a TEX macro or LATEX class or package file to do something, try searching Graham Williams' (`Graham.Williams@dit.csiro.au`) catalogue, available as `help/Catalogue/catalogue.html`; this lists many macro packages together with brief descriptive texts.

Having learnt of a file that seems interesting, search a CTAN archive for it (see question 23). For packages listed in *The LATEX Companion* (see question 17), 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://www.ora.com/homepages/CTAN-Web/`, which permits limited 'keyword' searching for files on the CTAN sites.

23 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 LATEX packages are often distributed sources with extension `dtx` rather than as package files with extension `sty`.)

One should make the regular expresion general enough to find the file you are looking for, but not too general, as the `ftp` interface will only return the first 20 lines that match your request.

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

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

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

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

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

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

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

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

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

```
quote site index dvips
```

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

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

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

A basic introduction to searching with regular expressions is:

- Most charcters 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';
- `"\"` 'quotes' a special character such as `"."` so `"\."` just matches `"."`;
- `"^"` matches the beginning of a line;
- `"$"` matches the end of a line.

For technical reasons in the quote site index command, you need to 'double' any \ hence the string `/caption\\.` in the above example. The quote site command ignores the case of letters. Searching for `caption` or `CAPTION` would produce the same result.

D Bits and pieces of TEX

24 What is a DVI file?

A DVI file (that is, a file with the type or extension `.dvi`) is TEX's main output file, using TEX in its broadest sense to include LATEX, etc. 'DVI' is supposed to be an acronym for DeVice-Independent, meaning that the file can be printed on almost any kind of typographic output device. The DVI file is designed to be read by a driver (see question 25) to produce further output designed specifically for a particular printer (e.g., a LaserJet) or to be used as input to a previewer for display on a computer screen. DVI files use TEX's internal coding; a TEX input file should produce the same DVI file regardless of which implementation of TEX is used to produce it.

A DVI file contains all the information that is needed for printing or previewing except for the actual bitmaps or outlines of fonts, and possibly material to be introduced by means of `\special` commands (see question 29).

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

25 What is a driver?

A driver is a program that takes as input a `dvi` file (see question 24) 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 26, 27, 28 and 57.

26 What are PK files?

PK files (packed raster) contain font bitmaps. The output from METAFONT (see question 54) 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.

27 What are TFM files?

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

28 Virtual fonts

Virtual fonts for TEX were first implemented by David Fuchs in the early days of TEX, but for most people they started when Knuth redefined the format, and wrote some support software, in 1989. Virtual fonts provide a way of telling TEX about something more complicated than just a one-to-one character mapping. The entities you define in a virtual font look like characters to TEX (they appear with their sizes in a font metric file), but the DVI processor may expand them to something quite different. You can use this facility just to remap characters, to make a composite font with glyphs drawn from several sources, or to build up an effect in arbitrarily complicated ways — a virtual font may contain anything which is legal in a DVI file. In practice, the most common use of virtual fonts is to remap PostScript fonts (see question 59) or to build 'fake' maths fonts.

It is important to realise that TEX itself does *not* see virtual fonts; for every virtual font read by the DVI driver there is a corresponding TFM file read by TEX. Virtual fonts are normally created in a single ASCII `vpl` (Virtual Property List) file, which includes both sets of information. The *vptovf* program is then used to the create the binary TFM and VF files. The commonest way (nowadays) of generating `vpl` files is to use the *fontinst* package, which is described in detail in question 59. `fonts/utilities/qdtexvpl` is another utility for creating ad-hoc virtual fonts.

29 `\special` commands

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

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

The allowable arguments of `\special` depend on the device driver you're using. Apart from the examples above, there are `\special` commands in the emTeX drivers (e.g., *dvihplj*, *dviscr*, *etc*.) that will draw lines at arbitrary orientations, and commands in *dvitoln03* that permit the page to be set in landscape orientation.

30 Documented LaTeX sources (`.dtx` files)

LaTeX 2ε, and many support macro packages, are now written in a literate programming style (see question 46), 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 LaTeX. Alternatively you can run LaTeX 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 LaTeX 2ε *docstrip* package (among other things, the installation process strips all the comments that make up the documentation for speed when loading the file into a running LaTeX system). Several packages can be included in one `.dtx` file, with conditional sections, and there facilities for indices of macros etc. Anyone can write `.dtx` files; the format is explained in *The LaTeX Companion* (see question 17). There are no programs yet to assist in composition.

`.dtx` files are not used by LaTeX 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.

31 What are the DC fonts?

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

When TeX version 3 arrived, some at least of the reasons for the eccentricity of Knuth's encodings went away, and at TUG's Cork meeting, an encoding for a set of 256 glyphs, for use in TeX text, was defined. The intention was that these glyphs should cover 'most' European languages, in the sense of including all accented letters needed. (Knuth's CMR fonts missed things necessary for Icelandic, Polish and Sami, for example, but the Cork fonts have them.) LaTeX 2ε (see question 107) 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 89).

The only METAFONT-fonts that conform to the Cork encoding are the DC fonts (available as `fonts/dc`; ensure you have version 1.2, patch level 1, released in December 1995, or later). They look CM-like, and should be regarded as an interim version of a hypothetical set of EC fonts (which, it is hoped, will be available some time in 1996). Their serious disadvantage for the casual user is that they are large — each DC 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 Cork encoding is also implemented by the PSNFSS system (see question 57), for PostScript fonts.

E Acquiring the Software

32 Repositories of TeX material

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

The CTAN sites are currently `ftp.dante.de` (`129.206.100.192`) and `ftp.tex.ac.uk` (`128.232.1.87`). The organisation of TeX files on 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` or `ctan@ftp.dante.de`; 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 23 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 TeX collection of material for VMS, Unix, MS-DOS, and the Macintosh. It is available via anonymous `ftp` from `wuarchive.wustl.edu` (128.252.135.4) in `decus/tex/`. It can also be obtained from the DECUS Library (reference number VS0058) in the US, or through your DECUS office outside of the US. To contact the DECUS Library, send mail or telephone:

DECUS
LIBRARY ORDER PROCESSING
334 South Street, SHR3-1/T25
Shrewsbury, MA 01545-4195

USA
Tel: 800-DECUS55 (within the USA, for information)
Fax: (+1) 508-841-3373 (for inquiries)

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

Finally, of course, the TeX user who has no access to any sort of network may buy a copy of the archive on CD-ROM (see question 35).

33 Contributing a file to the archives

Use anonymous `ftp` to any CTAN archive (see question 32) 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 23).

34 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 32).

35 TeX CD-ROMs

If you don't have access to the Internet, you can get the CTAN collections on a CD-ROM. Even those who do will find it very convenient to have large quantities of TeX-related files to hand.

Prime Time Freeware produced *TeXcetera* 1.1 in July 1994, which is a snapshot of CTAN taken in June 1994. Regular updates are planned. 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@cfcl.com`

or from many CD-ROM resellers, or the TUG office (see question 14). Price will be around $60. Please note that PTF is not a big commercial firm, and is a good friend of the TeX community.

Walnut Creek CDROM also provide a two-disc CD-ROM set, holding 1000Mb of TeX-related information. Information about the CD-ROM is available at `http://www.cdrom.com/titles/tex.html`, which also has a link to an ordering page. Walnut Creek's address, etc., are:

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

If you want a ready-to-run TeX system on CD-ROM, one is available for MS-DOS only (so far). The Dutch TeX Users Group (NTG) publish the whole 4AllTeX workbench on a 2-CD-ROM set packed with all the MS-DOS TeX software, macros and fonts you can want. It is available from NTG direct (see question 15), 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.

F TeX Systems

36 (LA)TeX for different machines

We list here the free or shareware packages; see question 38 for commercial packages.

Unix Instructions for retrieving the Unix TeX distribution via anonymous `ftp` are available in the document `systems/unix/unixtex.ftp`

A useful set of binaries for various common Unix systems is to be found as part of the teTeX distribution (`systems/unix/teTeX/distrib/binaries`); teTeX will compile on most Unix systems, though it was originally developed for use under Linux (see below).

AIX TeX for the IBM RS6000 running AIX is available in `systems/unix/aix3.2`

386/ix Executables for 386/ix are available in `systems/unix/386ix`

Linux There are at least two fairly complete implementations of TeX to run on Linux. The Slackware distribution includes NTeX (available as `systems/unix/linux/ntex`), which probably contains more TeX-related material than you would ever want. The more recent teTeX (available as `systems/unix/teTeX`) is based on Karl Berry's path-searching mechanisms, and is more compact than NTeX while still being pretty comprehensive.

PC The emTeX package for PCs running OS/2, MS-DOS or Windows includes LaTeX, BibTeX, previewers, and drivers, and is available in `systems/msdos/emtex` as a series of zip archives. The package was written by Eberhard

Mattes, and documentation is available in both German and English. Appropriate memory managers for using emTEX with 386 (and better) processors and under Windows, are included in the distribution.

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

TUG (and some of the other user groups) offer all freely-available TEX software for the PC. A catalogue is available free from TUG (see question 14).

PC: Win32 MikTEX, 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 or Windows/NT, and is available from `systems/win32/miktex`

Mac OzTEX is a shareware version of TEX for the Macintosh. A DVI previewer and PostScript driver are also included. It should run on any Macintosh Plus, SE, II, or newer model, but will not work on a 128K or 512K Mac. It was written by Andrew Trevorrow, and is available in `systems/mac/oztex`, or on floppy disks from TUG (see question 14). UK TUG prepays the shareware fee, so that members of UK TUG may acquire the software without further payment. Questions about OzTEX may be directed to `oztex@midway.uchicago.edu`

Another partly shareware program is CMacTEX (available as `systems/mac/cmactex`), put together by Tom Kiffe. This is much closer to the Unix TEX setup (it uses *dvips*, for instance).

VMS TEX for VMS is available as `systems/vms/Alpha/tex_axp_exe.zip` (for Alpha-based machines) or `systems/vms/VAX/tex_vax_exe.zip` (for VAX machines). Standard tape distribution is through DECUS (see question 32).

Atari TEX 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 TEX 3.1 (PasTEX) and METAFONT 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.

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

37 TEX-friendly editors and shells

There are good TEX-writing environments and editors for most operating systems; some are described below, but this is only a personal selection:

Unix Try GNU *emacs*, and the AUCTEX mode (`support/auctex`). This provides menu items and control sequences for common constructs, checks syntax, lays out markup nicely, lets you call TEX and drivers from within the editor, and everything else like this that you can think of. Complex, but very powerful.

VMS An *lsedit* mode for editing TEX source is available from TUG (see question 14) as TEXniques 1, VAX Language-Sensitive Editor, by Kent MacPherson (1985).

MS-DOS There are several choices:

- The (shareware) 4AllTEX workbench (`systems/msdos/4alltex`) provides a very comprehensive environment written in 4DOS which lets you access most TEX-related software in a friendly way. You can choose your own editor; something such as *QEdit* or *Brief* is suitable. This whole package is available in easy-to-use form on CD-ROM from TEX user groups.
- TEXshell (`systems/msdos/texshell`) is a simpler, easily-customisable environment, which can be used with the editor of your choice.
- Eddi4TEX (`systems/msdos/e4t`; also shareware) is a specially-written TEX editor which features intelligent colouring, bracket matching, syntax checking, online help and the ability to call TEX programs from within the editor. It is highly customisable, and features a powerful macro language.

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

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

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

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

Y&Y's commercial (and high-quality) Windows previewer, *dviwindo*, can be used as a good TEX shell, calling programs such as TEX, drivers, and editors (Y&Y supply the public domain PE, and recommend the commercial

Epsilon) from customisable menus (see question 38 for details of Y&Y).

Scientific Word is a WYSIWYG editing program, strong on maths, which uses LaTeX for output (see question 38 for contact address).

OS/2 Eddi4TeX 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 OzTeX.

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

There is another set of shell programs to help you manipulate BibTeX databases.

38 Commercial TeX implementations

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

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

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

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

ArborText Inc
1000 Victors Way
Suite 400
Ann Arbor MI 48108
USA

Tel: (+1) 313-996-3566
Fax: (+1) 313-996-3573

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

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

Northlake Software, Inc.
812 SW Washington, Ste 1100
Portland, OR 97201
USA

Tel: (+1) 503-228-3383
Fax: (+1) 503-228-5662
Email: `rau@nls.com`

Source: Email from Pat Rau, November 1994

PC; TrueTeX Runs on Windows 3.1, Window NT and Windows 95.

The Kinch Computer Co.
6994 Pebble Beach Court
Lake Worth FL 33467
USA

Tel: (+1) 561-966-8400 Fax: (+1) 561-966-0692 Email: `kinch@holonet.net`
Web: `http://www.emi.net/~kinch`

Source: Email from Richard Kinch, December 1995

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

pcTeX Long-established: now has 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: `pti@crl.com`
Web: `http://www.crl.com/~pti/`

Source: *TUGboat* **16**(2) (1995)

PC; VTeX Also "Bitmap-free".

MicroPress Inc
68-30 Harrow Street
Forest Hills, NY 11375
USA

Tel: (+1) 718-575-1816
Fax: (+1) 718-575-8038
Email: `support@micropress-inc.com`
Web: `http://www.micropress-inc.com/`

Source: MicroPress home page, April 1996

PC; microTEX MicroTEX and TEX 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 LATEX documents; they ship with TrueTEX from Kinch (see above). Queries within the UK should be addressed to Scientific Word Ltd., others should be addressed directly to the publisher, TCI.

Dr Christopher Mabb
Scientific Word Ltd.
98 Pont Adam
Ruabon
Wrexham
Clwyd, LL14 6EF
UK

Tel: 0345 660340 (within the UK)
Tel: +44 1978 824684
Fax: 01978 823066 (within the UK)
Email: `christopher@sciword.demon.co.uk`

TCI Software Research Inc.
1190 Foster Road
Las Cruces NM 88001–3739
USA

Tel: (+1) 505-522-4600
Fax: (+1) 505-522-0116
Email: `info@tcisoft.com`
Web: `http://www.tcisoft.com/tcisoft.html`

Source: Mail from Christopher Mabb, November 1995

Macintosh; Textures "A TEX system 'for the rest of us' "; also gives away a METAFONT implementation and some font manipulation tools.

Blue Sky Research
534 SW Third Avenue
Portland, OR 97204
USA

Tel: 800-622-8398 (within the USA)
Tel: (+1) 503-222-9571
Fax: (+1) 503-222-1643
Email: `sales@bluesky.com`
Web: `http://www.bluesky.com/`

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

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

Radical Eye Software
PO Box 2081
Stanford, CA 94309
USA

Source: Mail from Tom Rokicki, November 1994

G DVI Drivers and Previewers

39 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 32), and also from CTAN: `systems/vms/Alpha/dvips555_axp_exe.zip` (for Alpha-based machines), `systems/vms/VAX/dvips555_vax_exe.zip` (for VAX machines); support files are available in `systems/vms/dvips555_support.zip`, and a set of fonts for use with LATEX 2_ε are available in `systems/vms/dvips_fontsupport.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 OzTEX or Textures, or a port of *dvips* in the CMacTEX package.

40 DVI drivers for HP LaserJet

The emTEX package (see question 36) 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 39), is available in `dviware/dviljk`

41 DVI previewers

EmTEX and gTEX for the PC, and OzTEX for the Macintosh, all come with previewers that can be used on those platforms. There is a good OS/2 Presentation Manager previewer in emTEX, and a public domain Windows previewer (`dviware/dviwin`). Commercial PC TEX packages (see question 38) 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 39); 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

42 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`

43 TEXCAD, a drawing package for LATEX

TEXCAD 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 emTEX driver-family `\special`s (see question 29). TEXCAD is part of the emTEX distribution.

A Linux port of the program, `systems/unix/linux/xtexcad-2.1.tar.z`, is reported also to run on other Unix operating systems.

44 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-20-sea.hqx` (there are other dictionaries in the same directory).

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

45 The VORTEX package

VORTEX (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 VORTEX project*" in *TUGboat* **10**(1), pp. 11–14, 1989. It includes several nice previewers and some *emacs* modes for TEX and BIBTEX. The VORTEX distribution is not maintained, and now looks distinctly long in the tooth (it was never upgraded to TEX version 3).

VORTEX needed a separate workstation to run TEX in the background; modern PCs for the home can provide more processor power (than was available to VORTEX) 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 38), and is also the basis of many of the other environments mentioned in question 37.

I Literate programming

46 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`, which is gatewayed to the mailing list `litprog@shsu.edu` (see question 19 for details). The literate programming FAQ is stored as `help/LitProg-FAQ`

47 WEB for C, FORTRAN, and other languages

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

48 Conversion between (LA)TEX 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 32) 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 document to a TEX-compatible disk file. It was written 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 BIBTEX and *refer*/*tib* formats. They are in `biblio/bibtex/utils/refer-tools`

In spite of the directory name, it also contains a shell script to convert BIBTEX 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 (`rhl@astro.princeton.edu`). There is also a convertor to LATEX by Erwin Wechtl, in `support/rtf2latex`

Translation *to* RTF may be done (for a somewhat constrained set of LATEX documents) by TEX2RTF, which can produce ordinary RTF, Windows Help RTF (as well as HTML, see question 51). TEX2RTF 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.

An 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.html` and is archived as `help/wp-conv/texcnven.txt`

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.

49 Conversion from (LA)TEX 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-TEXing program.

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

50 Conversion from SGML or HTML to TEX

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

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

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

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

   - David Megginson's *sgmlspm*, written in Perl 5. Available from `http://www.uottawa.ca/~dmeggins/SGMLSpm/sgmlspm.html`
   - 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 LATEX file from it. The conversion code is subject to NCSA restrictions, but the whole source is available as `support/html2latex`

Jonathan Fine (`J.Fine@pmms.cam.ac.uk`) hopes to release, during 1996, his macro package that directly interprets and typesets an SGML source file.

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).

51 (LA)TEX conversion to HTML

TEX is a typesetting language, not a markup system. With properly-used LATEX, 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.

LATEX2HTML (`support/latex2html`) is a package by Nikos Drakos (mostly of *perl* scripts) that breaks up a LATEX 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 LATEX intra-document references and hyperlinks, and extends the mechanisms to permit reference to other (possibly remote) documents and other Internet resources. It translates LATEX 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.

LATEX2HTML 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 LATEX2HTML, and how to tailor it, in *TUGboat* **16**(2).

There are two alternative strategies:

1. Free-standing LATEX to HTML translations. Hard, but not impossible. Julian Smart's *latex2rtf* (available from `support/latex2rtf`) does a plausible job on a subset of LATEX;

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

52 Making hypertext documents from TeX

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

1. Try direct LaTeX conversion to HTML; see question 51;
2. Rewrite your document using Texinfo (see question 11), and convert that to HTML;
3. Look at Adobe Acrobat, an electronic delivery system guaranteed to preserve your typesetting perfectly. See question 53;
4. Invest in the hyperTeX conventions (standardised `\special` commands); there are supporting macro packages for plain TeX and LaTeX).

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

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

href: `html:<a href = "href_string">`

name: `html:<a name = "name_string">`

end: `html:</a>`

image: `html:<img src = "href_string">`

base_name: `html:<base href = "href_string">`

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

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

53 Making Acrobat documents from LaTeX

In the simplest case, use your DVI to PostScript driver, and run the result through Adobe's Acrobat *Distiller*; even simpler, if you use a Mac or Windows TeX system, is to install Acrobat Exchange, and use PDFwriter like a printer from your application. The latter is a dead end, though fine for simple documents, since you can't insert extra hyperlinks in the PDF output. For that, you need the Distiller route, which supports a special PostScript operator called `pdfmark`, for passing through information to the PDF.

To translate all the LaTeX cross-referencing into Acrobat links, you need a LaTeX package to suitably redefine the internal commands. There are two of these for LaTeX 2$_\varepsilon$, both based on the HyperTeX specification (see question 52): Sebastian Rahtz's *hyperref* (available from `macros/latex/contrib/supported/hyperref`), and Michael Mehlich's *hyper* (available from `macros/latex/contrib/supported/hyper`). You use *dvihps* (a modified *dvips*) to translate the DVI into PostScript acceptable to Distiller. Alternatively, if you know you only want Acrobat, *hyperref* also has a 'native PDF' mode, which works with plain *dvips* (or most other translators) and gives access to all the functionality of `pdfmark`.

Sadly, there are no free implementations of Distiller, nor any signs of them. *GhostScript* (versions 3.51 onwards) can display and print PDF files, however, if you are on a platform with no Acrobat Reader. You may see a DVI to PDF translator soon, but do not hold your breath.

K METAFONT

54 Getting METAFONT to do what you want

METAFONT allows you to create your own fonts, and most TeX users will never need to use it. METAFONT, unlike TeX, requires some customisation: each output device for which you will be generating fonts needs a mode associated with it. Modes are defined using the `mode_def` convention described on page 94 of *The METAFONTbook* (see question 17). You will need a file, which conventionally called `local.mf`, containing all the `mode_def`s 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 14). 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* distribution[7]) 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 `<font file name>.2602gf`) The `<magnification>` is a floating point number or 'magstep' (magsteps are defined in *The METAFONTbook* and *The TEXbook*). 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

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

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 2/27/90
% 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`

55 Which font files should be kept

METAFONT produces from its run three files, a metrics (TFM) file, a generic font (GF) file, and a log file; all of these files have the same base name as does the input (*e.g.*, if the input file was `cmr10.mf`, the outputs will be `cmr10.tfm`, `cmr10.nnngf`[8] 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.

56 Getting bitmaps from the archives

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

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

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

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

So what to do? You can build the fonts you need yourself with METAFONT: this isn't at all hard, and some drivers help you (*dvips*, and the emTEX drivers) construct the METAFONT commands. You might need to look at Karl Berry's collection of METAFONT modes (`fonts/modes/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

[7]The *command_name* is symbolically linked to *virmf*, and *virmf* loads `command_name.base`
[8]Note that the file name may be transmuted by such operating systems as MS-DOS, which don't permit long file names

Y&Y (see question 38 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

57 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 59. 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 61).

If you use LATEX 2$_\varepsilon$, 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/packages/psnfss`); it's supported by the LATEX3 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 59) 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 fonts found in most printers are provided with PSNFSS, packaged as `macros/latex/packages/psnfss/lw35nfss.zip`

For older versions of LATEX there are various schemes, of which the simplest to use is probably the PSLATEX macros distributed with *dvips*.

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

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

Some common problems encountered are discussed elsewhere (see question 60).

58 Previewing files using PostScript fonts

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

1. Convert the DVI file to PostScript and use a PostScript previewer. Some modern Unix X implementations have this built in (as does NeXT-step); (X11) Unix, Windows, OS/2, and MS-DOS users can use the free *GhostScript* (`support/ghostscript`), a complete level 2 implementation.
2. Under Windows on a PC, or on a Macintosh, let Adobe Type Manager display the fonts. Textures (Macintosh) works like this, and under Windows you can use Y&Y's *dviwindo* for bitmap-free previewing. (See question 38 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).

59 TEX font metric files for PostScript fonts

Font vendors such as Adobe supply metric files for each font, in AFM (Adobe Font Metric) form; these can be converted to TFM (TEX Font Metric) form. The CTAN archives have prebuilt metrics which will be more than enough for many people (`fonts/psfonts`; beware — this directory is at the root of a huge tree), but you may need to do the conversion yourself if you have special needs or acquire a new font. One important question is the *encoding* of (Latin character) fonts; while we all more or less agree about the position of about 96 characters in fonts (the basic ASCII set), the rest of the (typically) 256 vary. The most obvious problems are with floating accents and special characters such as the 'pounds sterling' sign. There are three ways of dealing with this: either you change the TEX macros which reference the characters (not much fun, and error-prone); or you change the encoding of the font (easier than you might think); or you use virtual fonts (see question 28) to *pretend* to TEX that the encoding is the same as it is used to. If you use LATEX 2$_\varepsilon$, it allows for changing the encoding in TEX; read the *LATEX Companion* (see question 17) 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 31) encoding.

Alan Jeffrey's *fontinst* package (`fonts/utilities/fontinst`) is an AFM to TFM converter written in TEX; it is used to generate the files used by LATEX 2$_\varepsilon$'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 28).

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 LATEX 2$_\varepsilon$ 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 38) Y&Y's Font Manipulation Tools package which includes a powerful *afmtotfm* program among many other goodies.

60 Problems using PostScript fonts

For the typical LaTeX user trying to use the PSNFSS (see question 57) 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 emTeX ones. You are responsible for making sure you are not breaking the licence restrictions on font you bought.

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

61 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 38) 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 57) thanks to Sebastian Rahtz.

MathTime 1.1 (3 fonts) TeXplorators (Michael Spivak)
The set contains maths italic, symbol, and extension fonts, designed to work well with Times-Roman. These are typically used with Times, Helvetica and Courier (which are resident on many printers, and which are supplied with some PC versions). In addition you may want to complement this basic set with Adobe's Times Smallcap, and perhaps the set of Adobe 'Math Pi' fonts, which include blackboard bold, blackletter, and script faces. The distribution includes support for use with `plain` TeX and LaTeX 2.09 (including code to link in Adobe Math Pi 2 and Math Pi 6). Support under LaTeX 2_ε is provided in PSNFSS (see question 57) thanks to Sebastian Rahtz.

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

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

Proprietary fonts Various sources.
Since having a high quality font set in scalable outline

form that works with TeX can give a publisher a real competitive advantage, there are some publishers that have paid (a lot) to have such font sets made for them. Unfortunately, these sets are not available on the open market, despite the likelihood that they're more complete than those that are.

Mathptm (4 fonts) Alan Jeffrey.
This set contains maths italic, symbol, extension, and roman virtual fonts, built from Adobe Times, Symbol, Zapf Chancery, and the Computer Modern fonts. The Mathptm fonts are free, and the resulting PostScript files can be freely exchanged. Contains most of the CM math symbols. Support under LaTeX 2$_\varepsilon$ in PSNFSS (see question 57) 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 38 for details). The MathTime fonts are also available from:

TeXplorators
1572 West Gray #377
Houston TX 77019
USA

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

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

62 Including a PostScript figure in LaTeX

LaTeX 2$_\varepsilon$ (see question 107) 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 LaTeX book (see question 17), and is available in `macros/latex/packages/graphics`

If you don't use LaTeX 2$_\varepsilon$, 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 `\special`s. The drivers mentioned in question 39 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 TeX has reserved for your figure, not the figure itself.

There are two rather good documents on CTAN addressing of figure production with rather different emphasis. Anil K. Goel's, `info/figsinltx.ps` covers the different ways in which you might generate figures, and one the old (LaTeX 2.09) ways of including them into documents. Keith Reckdahl's, `info/epslatex.ps`, covers the standard LaTeX 2$_\varepsilon$ 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

63 Drawing with TeX

There are many packages to do pictures in (La)TeX itself (rather than importing graphics created externally), ranging from simple use of LaTeX `picture` environment, through enhancements like *epic*, to sophisticated (but slow) drawing with PICTeX. Depending on your type of drawing, and setup, four systems should be at the top of your list to look at:

1. `graphics/pstricks`; this gives you access to all the power of PostScript from TeX itself, by sophisticated use of `\special`s. 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.

2. META O T; you liked METAFONT, but never got to grips with font files? Try META O T (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 TeX macros. Not *quite* the full power of METAFONT, but a friendlier interface.

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

64 Double-spaced documents in LATEX

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

LATEX is a typesetting system, so the appropriate design conventions are for "real books". If your requirement is from thesis regulations, find whoever is responsible for the regulations, and try to get the wording changed to cater for typeset theses (*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. Options available are:

  – for simple double spacing, `macros/latex209/contrib/misc/doublespace.sty`, and

  – for greater flexibility, `macros/latex/contrib/supported/setspace/setspace.sty`, which has been upgraded for LATEX 2_ε.

65 Formatting a thesis in LATEX

Thesis styles are usually very specific to your University, so it's usually not profitable to ask around for a package outside your own University. Since many Universities (in their eccentric way) still require double-spacing, you may care to refer to question 64. If you want to write your own, a good place to start is the University of California style (available as `macros/latex209/contrib/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 æsthetically pleasing document anyway!)

66 Flowing text around figures in LATEX

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_ε) 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*`}[`*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`*(nside)* or `o`*(utside)* 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.

67 Alternative head- and footlines in LaTeX

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

The package is found in directory `macros/latex/contrib/other/fancyheadings` 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. Fancyheadings also deals with the tedious behaviour of the standard styles with initial pages (see question 92), by enabling you to define different page styles for initial and for body pages.

68 Including a file in verbatim in LaTeX

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

Another way is to use the `alltt` environment, which requires `alltt.sty` (which is now part of LaTeX).

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.

69 Generating an index in (La)TeX

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 emTeX and gTeX distributions (the emTeX 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.

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

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

There are other programs called *texindex*, notably one that comes with the Texinfo distribution (see question 11).

70 Using BibTeX with `plain` TeX

The file `macros/eplain/btxmac.tex` contains macros and documentation for using BibTeX with `plain` TeX, either directly or with Eplain (see question 9). See question 20 for more information about BibTeX itself.

71 Typesetting music in TeX

A powerful package which allows the typesetting of polyphonic and other multiple-stave music is MusicTeX, written by Daniel Taupin (`taupin@rsovax.lps.u-psud.fr`). It is available in `macros/musictex`

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 MusicTeX, known as MusiXTeX. MusiXTeX is a three-pass system (with a processor program that computes values for the element spacing in the music), and achieves finer control than is possible in the unmodified TeX-based mechanism that MusicTeX uses. Daniel Taupin and Andreas Egler are pursuing distinct versions of MusiXTeX; they are available, respectively, from `macros/musixtex/taupin` and `macros/musixtex/egler`

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

A rather simpler notation than MusicTeX 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/abc2mtex`

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

72 Drawing Feynman diagrams in LaTeX

Michael Levine's macro package for drawing Feynman diagrams in LaTeX 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 META O T). The *feynmf* or *feynmp* package reads a description of the diagram written in TeX, and writes out code. METAFONT (or META O T) can then produce a font (or PostScript file) for use in a subsequent LaTeX run. For new users, who have access to META O T, the PostScript version is probably the better route, for document portability and other reasons.

N How do I do *X* in TEX or LATEX

73 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 LATEX finishes off the displayed maths and gets ready for a new line before it reads any instructions connected with ending the proof. But traditionally the end-of-proof sign goes in the display, not on a new line. So you just have to put it in by hand in every proof.

74 Symbols for the number sets

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 TEX 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 TEX or LATEX. 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.

75 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/packages/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 106) outside of LATEX 2_ε (see question 107), because the command `\rm` behaves differently there.

76 Labels on lists

If you want your top-level `enumerate`s to be labelled 'I/', 'II/', ..., then give these commands:

```
\renewcommand{\theenumi}{\Roman{enumi}}
\renewcommand{\labelenumi}{\theenumi/}
```

The possible styles of numbering are given in Section 6.3 of Lamport's book (see question 17). If you are lazy and just change `\labelenumi` in one go then your cross-references will be wrong.

For lower level `enumerate`s, 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_ε, the package `enumerate.sty` (part of `macros/latex/packages/tools`) offers similar facilities. With `enumerate.sty`, the example above would be achieved simply by starting the enumeration `\begin{enumerate}[I/]`.

77 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.

78 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/latex209/contrib/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.

2. Stick the table in a `minipage` anyway. This provides all the ugliness of footnotes in a minipage with no extra effort.

3. Use `macros/latex209/contrib/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/packages/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.

79 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 17; the programming itself is discussed in question 97). 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 106) outside of LaTeX 2_ε (see question 107), because the font-changing commands behave differently there.

80 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/packages/tools`) suppresses the mechanism, so that the first paragraph is indented.

81 Footnotes in LaTeX section headings

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

To suppress the footnote in headings and table of contents:

- Take advantage of the fact that the mandatory argument doesn't 'move' if the optional argument is present: `\section[title]{title\footnote{title footnot`

- Use the (small) package `macros/latex/contrib/other/misc/stblftnt.sty`, which makes `\footnote` automagically disappear in a moving argument.

82 Changing the margins in LaTeX

Don't do it. Learn some LaTeX, produce some documents, and then ask again.

You can never change the *margins* of a document by software, because they depend on the actual size of the paper. What you can change are the distances from the apparent top and left edges of the paper, and the width and height of the text. Changing the last two requires more skill than you might expect. The height should bear a certain relationship to `\baselineskip`. And the width should not be more than 75 characters. Lamport's warning in his section on 'Customizing the Style' really must be taken seriously. One-inch margins on A4 paper are fine for 10- or 12-pitch typewriters, but not for 10pt type (or even 11pt or 12pt) because so many characters per line will irritate the reader. However…

Perhaps the easiest way to get more out of a page in LaTeX is to get `macros/latex209/contrib/misc/fullpage.sty`, which sets the margins of the page identical to those of `plain` TeX, *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/latex/contrib/other/misc/vmargin.sty`, which has a canned set of paper sizes (a superset of that provided in LaTeX 2_ε), provision for custom paper, margin adjustments and provision for two-sided printing.

For details of LaTeX's page parameters, see section C.5.3 of the LaTeX manual (pp. 181–182). The origin in DVI coordinates is one inch from the top of the paper and one inch from the left side; positive horizontal measurements extend right across the page, and positive vertical measurements extend down the page. Thus, for margins closer to the left and top edges of the page than 1 inch, the corresponding parameters, *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 Lamport'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:

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

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.

83 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 LaTeX 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.

84 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/latex/packages/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/latex209/contrib/misc/comment.sty`

85 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 93). However, the canonical method for doing this in LaTeX 2_ε is to use the the `\DeclareMathOperator` command of `amsopn.sty` (which is part of the AMS-LaTeX package: `fonts/ams/amslatex`).

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

86 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 AMS-TeX, PICTeX, BIBTeX, 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{}` (AMS-TeX) for AMS-TeX, `Pic\TeX{}` (PicTeX) for PICTeX, `Bib\TeX{}` (BibTeX) for BIBTeX, and so on.

O Things are Going Wrong…

87 Weird hyphenation of words

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

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

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

A further source of oddity can derive from the 1995 release of the DC fonts (see question 31), which introduced an alternative hyphen character. The LaTeX 2ε configuration files in the font release specified use of the alternative hyphen, and this could produce odd effects with words containing an explicit hyphen. The font configuration files in the December 1995 release of LaTeX 2ε do *not* use the alternative hyphen character, thus removing this source of problems.

88 (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/english/ukhyph.tex`

89 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 31) which contain accented letters as single glyphs. In the future, perhaps, Omega (see question 109) will provide a rather different solution.

90 Enlarging TeX

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; emTeX allows you to adjust the memory allocation criteria on the command line. Consult the documentation that came with your implementation.

91 Moving tables and figures in LaTeX

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

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

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

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

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

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

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

92 `\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 `fancyheadings.sty`, which allows you to customise the style for initial pages independently of that for body pages. It is available in `macros/latex/contrib/other/fancyheadings`

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

93 Odd behaviour of `\rm`, `\bf`, *etc.*

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 106). Complain loudly; ask your system administrator to replace this version with LaTeX 2$_\varepsilon$ (see question 107), 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 `oldlfont.sty`, which should have been installed at the same time as NFSS.

94 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`, `\sevrm`, `\egtrm`, `\ninrm`, `\tenrm`, `\elvrm`, `\twlrm`, `\frtnrm`, `\svtnrm`, `\twtyrm` and `\twfvrm`. 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$_\varepsilon$; to use the unconverted LaTeX 2.09 packages under LaTeX 2$_\varepsilon$, you need also to include the package `rawfonts.sty` (which is part of the LaTeX 2$_\varepsilon$ distribution).

95 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 106) or LaTeX 2$_\varepsilon$ (see question 107). In the former case, use `oldlfont.sty`, as in the question 93. In the latter, use the package `latexsym`, which is part of the LaTeX 2$_\varepsilon$ distribution, or the package `amsfonts`, if it is available.

96 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}
```

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

But, you will say, "*The LaTeX Companion* tells me to use commands containing `@`!"

Indeed; for example, there's a lengthy section about `\@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
```

98 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 28) to map the old to the new series; it's available in fonts/vf-files/msx2msa

99 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)[9]. There's not a lot one can do with these fonts; they are (as their name implies) almost (but not quite) the same as the cm series; if you're faced with a document that requests them, all you can reasonably do is to edit the document. The appearance of DVI files that request them is sufficiently rare that no-one has undertaken the mammoth task of creating a translation of them by means of virtual fonts; however, most drivers let you have a configuration file in which you can specify font substitutions. If you specify that every am font should be replaced by its corresponding cm font, the output should be almost correct.

100 'String too long' in BIBTeX

The BIBTeX diagnostic "Warning–you've exceeded 1000, the global-string-size, for entry foo" is not one that you can hope to avoid by altering the BIBTeX style in a simple way — BIBTeX itself needs recompiling to increase its limit on string sizes (which is often not practical, and is never desirable). You must therefore address the problem by changing your bibliography database.

The problem usually arises from a very large abstract or annotation included in the database. The only way forward is to take the entry out of the database, so that you don't encounter BIBTeX'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 BIBTeX has to deal with is the file name, though it will tell TeX (when appropriate) to include all the long text.

P Why does it *do* that?

101 Why does it ignore paragraph parameters?

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

Consider the following sequence of LaTeX:

```
{\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...
```

TeX will open a group, and set the ragged-setting parameters within that group; it will then save a couple of sentences of text and close the group (thus restoring the previous value of the ragged-setting parameters). Then it encounters a blank line, which it knows to treat as a \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...
```

[9] The fonts acquired their label 'Almost' following the realisation that their first implementation in METAFONT79 still wasn't quite right; Knuth's original intention had been that they were the final answer

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

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

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

102 What's the reason for 'protection'?

Sometimes LaTeX saves data it will reread later. These data are often the argument of some command; they are the so-called moving arguments. ('Moving' because data are moved around.) Places to look for are all arguments that may go into table of contents, list of figures, *etc*.; namely, data that are written to an auxiliary file and read in later. Other places are those data that might appear in head- or footlines. Section headers and figure captions are the most prominent examples; there's a complete list in Lamport's book (see question 17).

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 LaTeX3 team have removed the need for protection of some things in the production of LaTeX 2ε, but the techniques available to them within current LaTeX mean that this is an expensive exercise. It remains a long-term aim of the team to remove all need for these things.

103 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`).

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 102).

104 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

```
  \uppercase{About $y=f(x)$}
```

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.

105 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\a#1{...#1...#1...#1...}  \a{b}
```

the macro expansion produces '. . . b. . . b. . . b. . . ', which people find normal. However, if we now fill in the '. . . ':

```
  \def\a#1{---#1---\def\x #1{xxx#1}}
```

`\a{b}` will expand to '---b---`\def\x b{xxxb}`'. 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\a#1{---#1---\def\x ##1{xxx##1}}
```

and `\a{b}` will expand to '---b---`\def\x #1{xxx#1}`', 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.

Q Recent Developments

106 The New Font Selection Scheme (NFSS)

NFSS was an extension to LaTeX 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 non-standard fonts such as the PostScript ones with LaTeX, 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 LaTeX 2.09 as supported software, the label 'NFSS' has become somewhat misleading, as there's no 'old' scheme with which to contrast it — LaTeX has incorporated the NFSS.

107 LaTeX 2ε (the new standard LaTeX)

LaTeX 2ε is a new version of the LaTeX package, prepared and supported by the LaTeX3 project team. It moved out of its test phase in June 1994, and is now the standard LaTeX; LaTeX 2.09 is no longer supported. New versions are released at (approximately) 6-monthly intervals; this does *not* mean the functionality is unstable, merely that the implementation is steadily being refined.

LaTeX 2ε is upwardly compatible with LaTeX 2.09, but has new features. In the latest (December 1995) release, these include:

- NFSS (see question 106) is part of the distribution.
- SLiTeX 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 102); 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.

108 The LaTeX3 project

The LaTeX3 project team is a small group of volunteers whose aim is to produce a major new document processing system based on the principles pioneered by Leslie Lamport in the current LaTeX. It will remain freely available and it will be fully documented at all levels.

The LaTeX3 team has already delivered its first product, LaTeX 2ε (see question 107), a macro package based on Lamport's original code, but modified to be more readily supportable than was Lamport's.

109 The Omega project

Omega (Ω) is a program built on top of TeX which works internally with 16-bit characters (Unicode); this allows it to work with most scripts in the world without any complications of coding schemes. 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@ift.ulaval.ca`) 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`

110 The $\mathcal{NTS}$ project

The $\mathcal{NTS}$ 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 13), and second, even if they *were* allowed to develop the program, some members of the $\mathcal{NTS}$ 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, will form a system called ε-TeX, which will be 100% compatible with TeX; furthermore, it will be possible during format creation to construct a format that *is* TeX: no extensions or enhancements will be 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.

R Perhaps There *isn't* an Answer

111 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 TEXbook), don't immediately write to Knuth, however. He only looks at bugs once or twice a year, and even then only after they are agreed as bugs by a small vetting team. In the first instance, contact Barbara Beeton at the AMS (`bnb@math.ams.org`), or contact TUG.

If you've found a bug in LATEX 2ε, look in the bugs database to see if it's already been reported. If not you should submit details of the bug to the LATEX3 team. To do this, you should process the file `latexbug.tex` with LATEX (the file is part of the LATEX 2ε 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.tex.ac.uk/ctan/latex/bugs.html`

If you've found a bug in LATEX2.09, 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[10].

[10]TUG maintains a register of TEX consultants; UK TUG is developing one