

Baskerville

The Annals of the UK T_EX Users' Group

Guest Editor: David Carlisle

Vol. 6 No. 1

ISSN 1354-5930

February 1996

Articles may be submitted via electronic mail to `baskerville@tex.ac.uk`, or on MSDOS-compatible discs, to Sebastian Rahtz, Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, to whom any correspondence concerning *Baskerville* should also be addressed. This *Baskerville* is set in Computer Modern using the dc fonts release 1.2 implementation, and printed on a Xerox Docutech. Production and distribution is undertaken in Cambridge by Robin Fairbairns. Back issues from the previous 12 months may be ordered from UKTUG for £2 each; earlier issues are archived on CTAN in `usergrps/uktug`.

Please send UKTUG subscriptions, and book or software orders, to Peter Abbott, 1 Eymore Close, Selly Oak, Birmingham B29 4LB. Fax/telephone: 0121 476 2159. Email enquiries about UKTUG to `uktug-enquiries@tex.ac.uk`.

Contents

I	Editorial	3
II	Letter to the Editor	4
III	Supplement to The 'Frequently Asked Questions'	5
	0.1 66a Alternative head- and footlines in L^AT_EX	5
IV	Large brackets and accents: the <code>yhmath</code> package	6
	1 Abstract	6
	2 Installation	6
	3 Availability	6
	4 Very big delimiters	6
	5 A new $\mathcal{A}\mathcal{M}\mathcal{S}$ -L ^A T _E X-like matrix-like environment	7
	6 New roots	7
	7 A few things missing from $\mathcal{A}\mathcal{M}\mathcal{S}$ -L ^A T _E X v1.2	7
	8 Very wide accents	8
V	The Custom-Bib Package	9
	1 Introduction	9
	2 Installation and Use	9
	3 Note on Author-Year Citations	10
	4 The Pseudo Language Babel	10
	5 Selected Features of <code>merlin.mbs</code>	10
	6 Package files	11
VI	Russian Paragraph Shapes	13
	1 The problem	13
	2 The first solution	13
	3 The second solution	13
	4 Comments	14
	5 Final versions	14
VII A	L ^A T _E X Tour: Part 1	16
	1 Introduction	16
	2 The Components of L ^A T _E X	16
	3 Documentation in the Base Distribution	17
	3.1 The ASCII text files	17
	3.2 The L ^A T _E X 'guides'	17
	3.3 L ^A T _E X News	17
	3.4 Example Documents	18

3.5 Documented sources	18
3.6 Errata	18
4 The L ^A T _E X Bug Report Database	18
5 Docstrip files	18
6 The Standard L ^A T _E X Classes	19
7 Standard Packages	20
7.1 Encoding Packages	20
7.2 Remaining Packages in the Base Distribution	20
8 Font Definition Files	21
9 Makeindex Styles	21
10 Miscellaneous Utilities and Files	21
VIII An introduction to PSTricks, part I	22
1 Preface	22
2 Introduction	22
2.1 Basic PSTricks concepts	23
3 The graphic objects	25
4 Examples of basic graphic objects	28
5 Mixing text and graphics	30
6 Working with a third dimension	33
IX Walnut Creek T _E X CDROM	35
1 Part 1: A user's view	36
2 Part 2: An archivist's view	37
X Malcolm's Gleanings	38
1 Out of METAFONT comes forth riches	38
2 Indefatigable	38
3 Guesting	38
XI Treasurer's Report	40
1 Membership Matters	40
2 Software distributions	40

I Editorial

David Carlisle
Mathematics Department
Manchester University

Welcome to the first *Baskerville* of 1996. As announced in *Baskerville* 5.4 Sebastian Rahtz has given up the editorship to devote more time to the TUG board. I am sure that all UKTUG members have appreciated Sebastian's efforts in masterminding a regular production of the journal. As Sebastian notes in a letter in this issue it was not a single handed effort, but nevertheless he must take much of the credit.

One of the major T_EX events of last year was the re-release of the 'Cork' encoded Computer Modern fonts. This release 1.2 of the dc fonts fixed many bugs and improved the fonts in many small but important ways. English language users can (and most of them do!) get by with the old computer modern encoding of Knuth, but the future in an increasingly international community must be with the Cork 8-bit encoding, and so as a mark of the new release this issue has departed from its eponymous text font and uses the 10pt Roman font dcr1000 as its main face. (Eagle eyed readers will no doubt already have noticed that the 'alternative' hyphenation character is being used.)

The position of editor is still *vacant!* If you are interested in the possibility of becoming editor (or guest-editing a single issue) please contact the committee at the above address.

Currently the editorship will rotate amongst the committee members. Robin Fairbairns again masterminded the FAQ issue, I have this one, Malcolm Clark will take *Baskerville* 6.2 and Carol Hewlett 6.3 ...

No matter who is the editor, *Baskerville* can not function without a supply of contributed articles. Sebastian has agreed to remain a 'collection point' for contributions, so they should continue to be sent to him at the address above.

Submission dates are as follows:

Issue	Submit material for publication	Submit last- minute notices	Anticipated posting date
6.2	April 3	April 17	April 26
6.3	June 3	June 7	June 21
6.4	August 5	August 16	August 30

We made a mistake in the FAQ issue. Jonathan Fine hopes to complete his macro package for typesetting SGML documents during 1996, and not by May as we stated in Question 50.

II Letter to the Editor

Sebastian Rahtz

In my report on Baskerville to the 1995 AGM (reported in *Baskerville* 5.5), and in my personal lookback on the history of the Annals, I did not really reflect the work of Jonathan Fine for the journal. Not only did he work hard on the production and distribution of most of the issues, but he also contributed a great many column inches in his regular columns. How could I have passed over the contributions of one of UKTUG's most colourful members? *Mea maxima culpa*. I look forward to reading more of Jonathan's work in the future!

III Supplement to The ‘Frequently Asked Questions’

Robin Fairbairns

I promised, in *Baskerville* 5.6, that I would attempt to provide regular updates of the list of questions and answers published in that edition. However, as one might have guessed, there have been no responses to my suggestion that you, our membership, submit questions to be answered . . .

Fortunately, I had made provision for this eventuality. I rather surprised myself to find that that we hadn’t answered the following question. I answered it (yet again) on `comp.text.tex` today, and speculated in my answer that it was possibly the *most* frequently asked question of all (apart from those that can be answered with the terse ‘read the *** manual’ that so often appears on Usenet).

0.1 66a 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 89 in *Baskerville* 5.6), by enabling you to define different page styles for initial and for body pages.

IV Large brackets and accents: the yhmath package

Yannis Haralambous
187, rue Nationale, 59800 Lille, France
haralambous@univ-lille1.fr

1 Abstract

This package¹ provides a set of big delimiters, intermediate to those of the original \TeX , and also much bigger. It also provides very wide accents (including two new ones: parenthesis and triangle). These symbols are included in a font which has Don's `cmex10` as lower ASCII part.

2 Installation

This package consists of (a) a font, written in Metafont, (b) a \LaTeX style file, (c) an `fd` file for the OMX encoding using the new font. To build the font put all the Metafont files somewhere where your Metafont can find them (for example in `texmf/fonts/src/public/yhmath`)

Then launch Metafont at least once on `yhcmex10` so that at least one `tfm` file exists when you'll start typesetting (`dvips` and similar programs will create the `pks`, don't worry).

Then take the `OMXyhex.fd` file and put it together with your other `fd` (Font Definition) files; and `yhmath.sty` together with your other \LaTeX styles. Have fun!

3 Availability

Don Knuth's code is included in Metafont files, so this code is under the usual \TeX ware copyright conditions. My code is postcard-ware. (If you like it and find it is worth a postcard + a stamp + the mental effort of writing a word [optional!] and the physical effort of going to the nearest mailbox, then do it!)

Everything is on CTAN, and if there are upgrades you will be informed in the usual way.

4 Very big delimiters

I never liked those parentheses of matrices which become almost immediately straight. In traditional math typography, parentheses stay curved, even if they are very big. So I decided to play around with \TeX 's `charlist` font property, and make some more of those big delimiters. I also did intermediate sizes (for all "big" delimiters). Here are some examples :

$$\left(\begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \end{array} \right) \left(\begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{array} \right) \left(\begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \\ m & n & o \end{array} \right) \quad (1)$$

$$\left(\begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \end{array} \right) \left(\begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{array} \right) \left(\begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \\ m & n & o \end{array} \right) \quad (2)$$

¹This article describes the currently available version. An extended package with more symbols is planned.

(1) is produced using the standard $\text{T}_{\text{E}}\text{X}$ fonts, (2) shows the result of using this package.

5 A new $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{T}_{\text{E}}\text{X}$ -like matrix-like environment

Since I also did “very big” versions of the “left angle” and “right angle” symbols, why not make “matrices” with them as delimiters? I have never seen such a mathematical object, but perhaps was it just because this construction wasn’t available yet? (This is a chicken and egg story).

I called this new $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{T}_{\text{E}}\text{X}$ -like environment `amatrrix` (“a” for “angle”). I hope AMS people will just love it and include it into $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{T}_{\text{E}}\text{X}$!²

Here are the same matrices as above, with angles instead of parentheses:

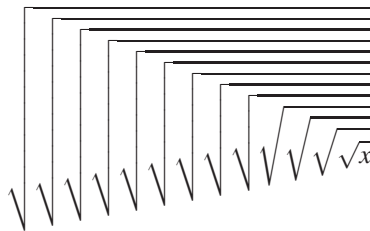
$$\left\langle \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \end{array} \right\rangle \left\langle \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{array} \right\rangle \left\langle \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \\ m & n & o \end{array} \right\rangle \quad (3)$$

$$\left\langle \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \end{array} \right\rangle \left\langle \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{array} \right\rangle \left\langle \begin{array}{ccc} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \\ m & n & o \end{array} \right\rangle \quad (4)$$

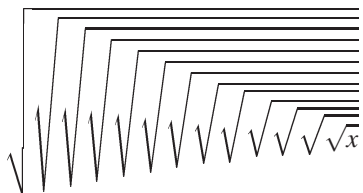
(3) is produced using the standard $\text{T}_{\text{E}}\text{X}$ fonts, (4) shows the result of using this package.

6 New roots

Roots got bigger as well, so that now the “vertical root” comes much later. Example :



$$\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{x}}}}}}}}}}}} \quad (5)$$



$$\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{x}}}}}}}}}}}} \quad (6)$$

(5) is produced using the standard $\text{T}_{\text{E}}\text{X}$ fonts, (6) shows the result of using this package.

7 A few things missing from $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{T}_{\text{E}}\text{X}$ v1.2

In $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{T}_{\text{E}}\text{X}$ there is a `\ddots` command for diagonal dots. How about antidiagonal ones? There are matrices called anti-symmetric, and for them we need the notation “dots going up”. I define a `\adots` macro, with a code symmetric to `\ddots`, here is the result: \cdot^{\cdot} .

²Talking of $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{T}_{\text{E}}\text{X}$ there are a few more macros I would like to see included, see next section.

Another thing missing in all T_EX & Co. packages: the ring accent, used in topology for the interior of a space. I define a macro `\ring` to be used in math mode. Here is the result: if $X = [0, 1]$ then $\overset{\circ}{X} =]0, 1[$.

8 Very wide accents

I added some more hats and tildes (accessed by the standard `\widehat` and `\widetilde` commands). so that you can get really wide accents now; see the examples below:

$$\widehat{A}, \widehat{ABC}, \widehat{ABCDE}, \widehat{ABCDEFG} \quad (7)$$

$$\widehat{A}, \widehat{ABC}, \widehat{ABCDE}, \widehat{ABCDEFG} \quad (8)$$

$$\widehat{A}, \widehat{ABC}, \widehat{ABCDE}, \widehat{ABCDEFG} \quad (9)$$

$$\widetilde{A}, \widetilde{ABC}, \widetilde{ABCDE}, \widetilde{ABCDEFG} \quad (10)$$

$$\widetilde{A}, \widetilde{ABC}, \widetilde{ABCDE}, \widetilde{ABCDEFG} \quad (11)$$

$$\widetilde{A}, \widetilde{ABC}, \widetilde{ABCDE}, \widetilde{ABCDEFG} \quad (12)$$

(7) and (10) show the standard T_EX font. (8) and (11) show the larger accents possible using the AMS fonts, as defined in the *A_MS-L_AT_EX* package `amsfonts`. (9) and (12) show the larger accents produced by the new `yhcmex10` font.

I also designed two new accents: the triangle accent `\widetriangle` and the parenthesis accent `\wideparen`:

$$\widehat{A}, \widehat{ABC}, \widehat{ABCDE}, \widehat{ABCDEFG}$$

$$\widehat{A}, \widehat{ABC}, \widehat{ABCDE}, \widehat{ABCDEFG}$$

The former is used (in France only??) to show that the notation ABC , where A, B, C are three points, means a triangle and not an angle. See what I mean? \widehat{ABC} is a triangle, \widehat{ABC} is an angle.

The latter is used when we want a non-expandable accent to be applied to more than one letters at once. Of course *A_MS-L_AT_EX* has given a solution to this (place the symbols between parentheses and the accent as an exponent of the right parenthesis), by I happen not to like that solution. For example if I want to write “the interior of $[0, 1]$ ”

I prefer to see $\overset{\circ}{[0, 1]}$ than $([0, 1])^\circ$ don't you?

Of course this notation is not my invention, I saw it in many French math books (ever heard of Nick Bourbaki?).

I call this macro `\widering`, because it plays the rôle of a wide symbol (and since the ring can't be widened, a parenthesis is used). Here are some more examples (the first one coded as `\ring{A}`):

$$\overset{\circ}{A}, \overset{\circ}{ABC}, \overset{\circ}{ABCDE}, \overset{\circ}{ABCDEFG}$$

V The Custom-Bib Package

Patrick W. Daly
Max-Planck-Institut für Aeronomie
D-37189 Katlenburg-Lindau, Germany
daly@linmpi.mpa.gwdg.de

1 Introduction

This article³ describes the `custom-bib` package for generating customized $\text{BIB}\text{T}_{\text{E}}\text{X}$ bibliography styles from a generic file by means of Frank Mittelbach's `docstrip` program. Many authors are frustrated at the wide range of bibliographic styles demanded by journals and publishers, and at the limited number available with standard $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ and $\text{BIB}\text{T}_{\text{E}}\text{X}$. This is not $\text{BIB}\text{T}_{\text{E}}\text{X}$'s fault, but rather shows the lack of any bibliographic standards in the English language. Often the differences are trivial — comma or colon, date in brackets or parentheses; but the normal user does not want to tackle the task of making up his own `.bst` file (no normal human would!).

For this reason, I set out to produce a generic `.bst` file that could have features and options selected by means of the `docstrip` program, which is now part of the standard $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ installation. I found over 50 different `*.bst` files and compared their outputs just for article: they were all different. They served as part of my input as to which features were needed. I knew of a few extra that were not covered by these 50.

I also discovered `btxbst.doc`, Oren Patashnik's source for the four standard `.bst` files, as well as a file `physics.bst`, for extracting `.bst` files for a number of physics journals. Both of these do precisely what I had set out to accomplish; however, they require the C preprocessor, or something similar. It is really a simple matter to convert the preprocessor commands into `docstrip` equivalents.

Nevertheless, neither of these really met all of my needs, so I continued to develop `genbst.mbs` (GENeric-BibST, MasterBibStyle). It soon became obvious that the sheer number of options necessary made any kind of customizing a difficult chore. Hence, the next step: the program `makebst` takes menu information out of the selected `.mbs` file and presents the user with descriptive choices as menus. From the answers, it writes a `docstrip` batch job (extension `.dbj`) which when $\text{T}_{\text{E}}\text{X}$ 'ed, creates the desired `.bst` file out of the `.mbs` one. The `.dbj` file may even be hand edited if one wants to alter only one or two options.

Since I first released this system in November 1993, I have received many suggestions and requests for additions. I have tried to incorporate as many as possible. The original 50 options have grown to over 100. The rate at which suggestions are sent to me has decreased considerably, so one can hope that the system is becoming stable.

The second version of `genbst.mbs` allowed other languages to be included. However, since the method had considerable overhead per language, I was unsatisfied with it. A parallel version called `babel.mbs` (which was really `genbst.mbs` version 2) has been available for some time, supporting English, French, German, and Esperanto, as well as a generic language called Babel.

I have now modified `makebst` to allow more than one `.mbs` file to be used as input for any given `.bst` output. This means that the language support can be contained in separate files, one per language, and does not need to burden the main file. The new issue of this main file, version 3, is now called `merlin.mbs`, to emphasize its magical powers.

2 Installation and Use

To install the package you need to have `docstrip.tex` and `doc.sty`; if you do not have the former, you cannot use the package anyway. Both are now part of the standard $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ (2 ϵ) distribution from 1994 June 1.

The steps for installing are:

0. (Optional, for connoisseurs.) $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ `makebst.dtx` to get the documentation. This also extracts `makebst.ins` from `makebst.dtx`, if it does not already exist.
1. Run $\text{T}_{\text{E}}\text{X}$ (or $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$) on `makebst.ins` (to generate `makebst.tex`)

³Based on the README file distributed with version 3.7 of the package, dated February 1, 1996

2. Run $\text{T}_{\text{E}}\text{X}$ (or $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$) on `makebst.tex` to start customizing your own `.bst` file. You may select the master file as either `merlin` (default), or one of the older `genbst` and `babel`, if you have them.
3. Answer the questions that then arise; for `merlin`, you will also be asked which language support file you want (default is `merlin` itself, meaning English or the pseudo-language Babel). You will also be asked if you want to add a file defining short-hand designations for various journals; such files are included (`physjour.mbs`, `photjour.mbs`, `geojour.mbs`) for journals in physics, optics, and geophysics, but you could create your own.
4. The menus that appear may not be informative enough for you. More information can be obtained by reading the documentation (on the options) contained in the `.mbs` files themselves. This documentation can be printed out by running $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ on the `.mbs` files.
5. The `makebst` program only produces a `docstrip` batch job to generate the `.bst` file. The last question it asks is whether that job should be run right away. You can always run it again yourself by running $\text{T}_{\text{E}}\text{X}$ or $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ on this `.dbj` file. (The `.dbj` file can also be manually edited if you want to play around with the various options it includes.)

3 Note on Author–Year Citations

Author–year style citations are not supported by standard $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ and $\text{BIB}_{\text{E}}\text{X}$. However, there exist a large number of bibliography styles for this, all of which need some interface package to run properly. I have identified 5 different interfaces, plus a 6th one that I use myself (see the `natbib` package in a parallel directory). I have made up `merlin.mbs` to be able to produce `.bst` files for all 6. If one selects author–year style, then one is asked which interface package is to be used. (Incidentally my `natbib` package is capable of interpreting all 6!)

4 The Pseudo Language Babel

If one selects the language to be Babel, all explicit words in the resulting `.bst` file are replaced by commands. For example, ‘editor’ is replaced by `\bbleditor`. The translations of these commands are to be found in a file named `babelbst.tex`, which can be extracted from `merlin.mbs` with the `docstrip` option `babelbst`, albeit only for English. My original idea was that one could edit this file as one pleases for other languages, or that it could be incorporated into the `babel` multilingual system. However, this needs further development.

5 Selected Features of `merlin.mbs`

- List all options in the `dbj` file.
The `makebst` program writes to the `.dbj` file all the `docstrip` options that were offered in the interactive session, with all but the selected ones commented out. This makes editing afterwards much easier. Often one wants to experiment with some of these options, but only wants to run the `makebst` program once. This feature (suggested by Frank Mittelbach) is available when the `makebst.tex` file is extracted from `makebst.dtx` with the ‘`optlist`’ option; without it, only the selected options are listed in the `.dbj` file. Edit the `makebst.ins` file accordingly; by default, this feature is included.
- Allow multiple input `.mbs` files. The `.dbj` file is now so constructed that `\generateFile` can read in more than one input `.mbs` file for a single output `.bst` file. The main `.mbs` file must have its menu coding arranged to take advantage of this. Older `.mbs` files (like `genbst.mbs` and `babel.mbs`) will still work just as well with this version of `makebst`. Similarly, `merlin.mbs` can be used with older versions of `makebst`, but without any additional input files.
- Name formatting: can also have reversed full names, as Smith, John George. (Previously reversed names could only be initials.) Thus the following forms are allowed:

```
John George Smith
Smith, John George
J. G. Smith
Smith, J. G.
Smith, J. G. and F. M. Jones
Smith, J G
Smith, JG
Smith J G
```

Editors' names (in collections) for surname-first styles, may now be formatted exactly as the authors'. (Previously they would never be reversed.) It is also possible to have

In: B. G. James (editor) Booktitle
In: B. G. James, editor, Booktitle
In: Booktitle, edited by B. G. James
In: Booktitle (edited by B. G. James)

- ISBN numbers can optionally be included, if they're present in the database.

- Volume, number has more possibilities:

34(2) 34 (2) 34, 2 34, no. 2 34, #2 34

The following page number can be separated by colon, colon space, semi-colon and space, comma and space. It is even possible to add to number to the page specification: 34, (2)234–(2)254.

- Date: year coming just after authors may have colon and space following. Date may appear as '1994 Jul', with or without a dot. Date may be part of journal specification, something that is common in medical journals. Date may be bold.
- Journal names: the periods in abbreviations may be removed, so Phys. Rev. becomes Phys Rev (no change to database necessary). Name of journal can be in normal font, not only italic.
- Author names in may be italic, small caps, or bold. The word 'and' can be in the regular text font, not in the author font. The font style may be different in the citations and list of references.
- If the number of authors exceeds a certain limit, then only so many are listed, followed by *et al.* Both these maximum and minimum numbers can be set. Default is that all authors are listed.
- The page numbers in edited works can have 'pages' or 'pp' suppressed.
- It is possible to have the names sorted by ignoring the 'von' part, so that della Robbia comes after Rabin.
- Publisher's address may come before name, as New York: New Press (required by some psychology journals).
- The extra labels added to years (as 1995a) are grouped in braces to avoid some problems with natbib when this extra label is more than one letter. The font of this extra label can also be selected.
- For author-year systems, both full and abbreviated author lists are possible in the citations.
- Journals can have date between volume and pages, as: J. Geophys. Res. **34** (1994) 333–338
- Technical Reports can have titles treated like books (default is like article).
- Can sort by year and then authors.
- Can include more than one file with prestored journal names.
- The `named` format for `\bibitem` now included.
- Blocks can be separated by colons as well as commas or periods.
- Pages in books may be in parentheses.
- Authors in the list may separated by semi-colons instead of by commas.

6 Package files

The package contains the following files:

`merlin.mbs` A master `BIBTEX` style file for producing customized styles (numerical or author-year) with `docstrip`.

It is self-documenting: simply `latex` it to produce its description.

`english.mbs` A sample language support file for English, to act as a model for hacking others.

`esperant.mbs` A language support file for Esperanto.

`finnish.mbs` A language support file for Finnish.

`french.mbs` A language support file for French.

`german.mbs` A language support file for German.

`italian.mbs` A language support file for Italian.

`norsk.mbs` A language support file for Norwegian.

`spanish.mbs` A language support file for Spanish.

(Further contributions and corrections are welcome.)

`physjour.mbs` A support file to add the names of common Physics journals in shorthand form, for example 'pr' for Physical Review, or Phys. Rev., depending on whether abbreviations chosen.

`photjour.mbs` A contributed file containing names of optics journals.

`geojour.mbs` A contributed file containing names of geophysics journals.

(Contributions for other fields are welcome.)

The `.mbs` files can only be used effectively with the `makebst` ‘program’, which is included in documented source form.

`makebst.dtx` the documented source file; \LaTeX ing this file produces the manual and optionally a documentation of the coding. Requires `ltxdoc` class from the \LaTeX 2 ϵ distribution.

Current version is 3.0; the documentation requires \LaTeX 2 ϵ (1994/06/01). The program itself only requires \TeX .
`makebst.ins` `docstrip` batch job to extract from `makebst.dtx` the program file `makebst.tex`. (This file is actually included within `makebst.dtx`; it is extracted when the `.dtx` file is processed under \LaTeX 2 ϵ .)

VI Russian Paragraph Shapes

Based on `comp.text.tex` article of Peter Schmitt
with extra commentary by David Carlisle.

1 The problem

Recently on the `comp.text.tex` newsgroup The following question was asked (as an aside in a thread that was originally about “My Life with T_EX”).

Related problem: *the last line of a paragraph must be **either** longer than `\parindent` and shorter than `(\hsize - \parindent)`, **or** it must reach the right margin (exactly `\hsize` long).*

How can I implement this with the existing continuous `\parfillskip` glue? This is my publisher’s demand, based on traditional Russian typography rules.

In fact it is clear that this requirement cannot be satisfied by any setting of the paragraph parameters alone, however the question produced two deceptively simple solutions from two ‘regulars’ on that newsgroup, Peter Schmitt and Donald Arseneau. In a couple of followup articles these methods were refined. This article is based on Peter’s summary as posted to the newsgroup and typeset with the requested paragraph style.

The ideas here may be of use to anyone wishing to implement non-standard paragraph shapes, not just the particular requirements of this question.

2 The first solution

You can do this by ending each paragraph by `glue + hbox + glue` where the (empty) `hbox` spans `\parindent`, `glue + hbox` together range from `\parindent` to `\hsize - \parindent`, and `hbox + glue` range from `\hsize - \parindent` to `\hsize`, such that a break may occur either before `glue + hbox` (such that glue disappears) or just after `hbox`. In both cases the paragraph will end in an empty line. Therefore you have to back up one line.

```
\parfillskip0pt

\def\russianpar{%
  \hskip2\parindent plus\hsize
  \hskip-2\parindent
  \hbox{\hskip\parindent}%
  \hskip0pt
  \hbox{}%
  \hskip-\parindent
  \hskip\hsize plus\parindent
  \vadjust{\vskip-\baselineskip}%
  \endgraf}
```

3 The second solution

Donald Arseneau’s solution is similar, but rather than always forcing a blank row at the end of the paragraph, a rule of a special (small) depth is inserted together with glue items. If the line breaks leaving this rule on its own on the last line, then this can be detected by inspecting `\prevdepth`, and a suitable negative skip can be added to compensate for the ‘blank’ line.

```
\def\par{%
  \ifhmode
    \unskip
    \strut
```

```

\hskip-\parindent
\vadjust{ }%
\nobreak
\hskip2\parindent
\vrule depth 54321sp
      height \ht \strutbox width 0sp
\endgraf
\ifdim\prevdepth=54321sp
  \nobreak
  \vskip-2\baselineskip
  \hbox{\strut}%
\fi
\fi}

\parfillskip=\hsize minus\hsize
\advance\parfillskip
  -2\parindent minus-2\parindent

```

4 Comments

Donald Arseneau commented on one problem with the first solution: Unfortunately, plus `\hsize` does not set a firm limit on the stretch the same way that `minus\hsize` sets a limit on the shrink. Inserting the settings `\tolerance=100 \pretolerance=100` may fix this, but I doubt that such low tolerances would be practical when building a paragraph under such “Russian” constraints to the line breaking.

To which the reply was: One might, however, wish to take advantage of this effect by setting a higher (than `\parindent`) limit for the last line where the length is chosen such that a tolerable line would still meet the `\parindent` requirement. On average, this should prefer longer last lines and this is certainly aesthetically better.

Some more remarks: calculating the `\parskip` has the advantage (over setting `\parskip` to a calculated value) that one may change `\hsize` and `\parindent` without needing to adapt other values. If one does not mind this, using fixed values (calculated once) is certainly more efficient. In this case one should also think of using prepared boxes (`\setbox` and `\copybox`) instead of producing these boxes each time when a paragraph is set.

The trick of passing information via the depth of a special strut is well known, but relies on the fact that no other macro package is using the same value. Donald Knuth remarks in the `TEX`Book “*A distance of 1000sp is invisible to the naked eye, so a variety of messages can be passed in this way.*” However if there is a chance that several macro packages really are using this trick, it would be wise to define an allocation mechanism (cf. `\newcount`) that allocates a unique ‘special depth’ each time it is called. Similar comments could apply to allocating penalty values over 10000 which are also often used to flag special actions to be taken.

As all these methods require a redefinition of `\par` (which is inserted by `TEX` automatically for each blank line in the input file) they may need some further work to work in conjunction with other macros that redefine `\par`, probably the most common such case are the `LATEX` list environments.

5 Final versions

After considering the points raised in the discussion Peter Schmitt posted the following two variants, which take more care over inserting the glue, and an original use of `\discretionary`. This article is set with `\par` defined by the first of these methods, and with paragraph indent of 1cm.
1)

```

\parfillskip0pt

\def\russianpar{%
  \ifhmode
    \unskip
    \hskip-2\parindent minus-2\parindent
    \hskip\hsize minus\hsize

```

```

\hbox{\hskip\parindent}%
\hskip0pt
\hbox{\strut}%
\hskip-\parindent
\hskip\hsize plus\parindent
\vadjust{\nobreak\vskip-\baselineskip}%
\endgraf
\fi}

```

2)

```

\parfillskip0pt

\def\Russianpar{%
\ifhmode
\unskip
\strut\vadjust{ }%
\nobreak
\discretionary
{ }%
{\hbox{\hskip2\parindent
\vrule depth 123sp
width 0sp
height \ht \strutbox}}
{\hbox{\hskip\parindent}}%
\hskip-2\parindent minus2\parindent
\hskip\hsize minus\hsize
\kern0pt
\endgraf
\ifdim\prevdepth=123sp
\nobreak
\vskip-2\baselineskip
\hbox{\strut}%
\fi
\fi}

```

VII A L^AT_EX Tour: Part 1

David Carlisle

1 Introduction

In this article I hope to give a ‘guided tour’ around the files that make up the basic L^AT_EX distribution. Subsequent articles in this mini-series will cover other packages by the L^AT_EX development team, and also some of the main contributed packages.

The primary source for L^AT_EX is the ‘CTAN⁴’ network of archives, so if I refer to path names of files this relates to the CTAN file structure. Note however that if you obtained L^AT_EX as part of a ‘pre-packaged’ T_EX distribution, then these files may have been moved (typically documentation files may be separated from T_EX source files). Hopefully this will not cause any confusion.

2 The Components of L^AT_EX

The L^AT_EX distribution at the CTAN archives is organised into the following directories.

`base` Contains the core L^AT_EX files. You need at least these files for a minimal L^AT_EX installation.

`unpacked` Includes *all* the files in `base` together with the result of ‘unpacking’ the source files. (More about this later.) Thus when obtaining L^AT_EX you should get either `base` or `unpacked`, but not both. Getting the former saves on time transferring the files, but getting the latter saves time that would be taken unpacking the source files, so which is preferable depends on the relative speed (and cost) of your machine and your connection to the archives.

`packages` Consists of seven independent L^AT_EX ‘extensions’ that are written and supported by the L^AT_EX developers (or the American Mathematical Society).

`amsfonts`, `amslatex`, `babel`, `graphics`, `mfnfss`, `psnfss` and `tools`

These packages will be described in more detail later in the series.

`fonts` The metafont sources and T_EX font metric files of a few fonts that L^AT_EX requires that are not part of the original plain T_EX distribution.

`doc` This directory is not part of the main L^AT_EX distribution, it is generated by the CTAN archives. As a convenience for those people that have not yet installed L^AT_EX, some of the main introductory documentation files which are available as L^AT_EX files in the base distribution are made available in this directory as dvi and POSTSCRIPT files.

`contrib` This directory contains an ever growing number of contributed L^AT_EX packages, and other extensions, that have been contributed by L^AT_EX users. They are not part of the ‘official’ L^AT_EX distribution, but many of them form a vital part of any ‘working’ L^AT_EX installation. The packages are divided into two subdirectories ‘supported’ and ‘other’, however at the current time one should ignore this distinction when looking for packages to fetch from the archives. Contrary to expectations some of the best supported packages are distributed (at their author’s request) from `contrib/other`.

Unfortunately (for mainly historical reasons) people in search of contributed L^AT_EX packages also need to look in more distant CTAN directories. Firstly, the `macros/latex209/contrib` area on CTAN contains packages that were written for the previous version of L^AT_EX that has been obsolete for 19 months now. Any packages that are still distributed from this L^AT_EX2.09 tree are likely to be less than well supported, but you can still find some useful files there. Secondly, there are some packages that work with multiple formats, not just L^AT_EX, and these are to be found under `macros/generic` or in their own top-level directory, such as `macros/musictex`.

⁴`ftp.tex.ac.uk` in the UK

3 Documentation in the Base Distribution

The documentation that comes with \LaTeX is of two forms: plain (ASCII) text files with extension `.txt`, or \LaTeX documents distributed as \LaTeX source with extension `.tex`. Generally speaking the text files are mainly of interest to people installing \LaTeX , who may need information before they have a working system. Information of more general interest to \LaTeX users is normally distributed as a \LaTeX document.

3.1 The ASCII text files

Installation instructions

`00readme` Provides a general introduction to the system, and should be the first file to look at when installing \LaTeX for the first time.

`install` Provides ‘generic’ installation instructions, but for many \TeX versions more specific instructions that have been contributed by the authors or users of those systems, thus `emtex` gives instructions for the popular \emTeX implementation, `web2ctex` gives specific instructions on installing under UNIX, etc.

`tex2` If you have a \TeX that pre-dates version 3.0 (which was released in 1989) by far the best thing to do is to update your \TeX , but if that is really not possible this file details how \LaTeX may be built under \TeX 2.

`autoload` Describes the installation of an ‘autoloading’ version of \LaTeX . This produces a much smaller format by saving less common commands in external files rather than in memory. These files are automatically ‘autoloaded’ as required. This version of \LaTeX is particularly recommended if you are using a small installation (for instance a ‘small’ \emTeX on a sub-386 PC).

Other text files

`legal` Contains the copyright notices and distribution conditions for \LaTeX .

`bugs` Contains instructions on how to compile a bug report (see below).

`patches` Describes the \LaTeX patch mechanism that is used for distributing small updates between the ‘full’ releases. This file also contains a list of all the files that have changed since the last full release.

`changes` A Change Log of all the changes made to the \LaTeX files. This is mainly intended for internal use by the \LaTeX developers, but some people like to read it.

3.2 The \LaTeX ‘guides’

These documents are distributed as \LaTeX source (i.e. `.tex` files) although as noted in the above introduction, the CTAN archives distribute most of them in ready-formatted versions in the directory `latex/doc` so you can read these before installing \LaTeX if you wish. Unlike the ASCII text files described above, most of these documents are primarily intended for *users* of the system rather than system managers and software installers.

`usrguide` *\LaTeX 2 ϵ for Authors*. This document describes all the main new features of the 2 ϵ release of \LaTeX . It was written originally with the user of the old \LaTeX 2.09 in mind, but newcomers to \LaTeX who have never used the old version should still gain something by reading this document. It does not however cover the majority of \LaTeX commands that were not changed, and so it is not a substitute for a full \LaTeX manual.

`clsguide` *\LaTeX 2 ϵ for class and package writers*. A companion to `usrguide`, gives details of the \LaTeX commands for structuring class files and extension packages.

`fntguide` *\LaTeX 2 ϵ font selection*. For font addicts only, but if you want to know the detailed specification of the ‘New Font Selection Scheme’ commands, here is the place to look.

`cfgguide` *Configuration options for \LaTeX 2 ϵ* . Discusses what you can (and can not) do to configure a \LaTeX installation to the requirements of your local site.

`ltx3info` *The \LaTeX 3 Project*. A brief summary of the aims of the \LaTeX 3 project, the group of volunteers that has taken on the maintenance and development of \LaTeX .

`modguide` *Modifying \LaTeX* . This document discusses some of the rationale behind the \LaTeX distribution conditions as expressed in `legal.txt` and `cfgguide.tex`. Unless you are making a distribution of a modified version of \LaTeX , or are particularly interested in software copyright issues, you probably do not want to read this.

3.3 \LaTeX News

As well as these larger documents there are a series of one-page ‘newsletters’. A new one is produced with each full release of \LaTeX . These detail any changes that have occurred in \LaTeX or the main extension packages over the six months since the previous release. (\LaTeX releases occur at regular intervals, in June and December of each year.) Currently the four files `ltnews01–ltnews04` are distributed corresponding to the four releases of \LaTeX since June 1994.

3.4 Example Documents

There are two (very) small example documents, as described in the \LaTeX book by Leslie Lamport.

`small2e` A very small (1 page) \LaTeX document.

`sample2e` A slightly larger document.

3.5 Documented sources

The source for the \LaTeX format, and for all the packages and classes in the core distribution is distributed as ‘`.dtx`’ files. These are \LaTeX documents which may be processed in the usual way to produce typeset documentation. For example a command such as `latex ltpictur.dtx` would produce documented source of the picture mode commands. The files with names of the form ‘`!tdtx`’ make up the source of the \LaTeX format. If you want to produce a combined document incorporating all these files, you may process `source2e.tex`. This document will produce a typeset version of the \LaTeX sources, together with change log and index. It is well over 500 pages long, and so may take a long time to produce. It may produce an index that is too large to be handled by the ‘`makeindex`’ program on smaller machines.

3.6 Errata

The principal documentation for \LaTeX is the two books *\LaTeX : A Document Preparation System*, and *The \LaTeX Companion*. Errata for these (and the German edition of *The Companion*) are available as `manual.err`, `compan.err` and `begleit.err`.

4 The \LaTeX Bug Report Database

As described in the file `bugs.txt` mentioned above, the \LaTeX 3 project maintain a database of bug reports for \LaTeX .

If, after checking with colleagues, reading the manual, etc., you decide that some behaviour of \LaTeX is incorrect then you may send a message to the \LaTeX bug database. Before doing this you should check that your \LaTeX is not more than one year old (the bug may have been fixed in a recent release). If you have access to the World Wide Web, you may access the database and see if the problem is already reported by using the search page accessible from: <http://www.tex.ac.uk/CTAN/latex/bugs.html>.

If you decide to send a report, two files are available to help compose a message in the correct format:

`latexbug.tex` \LaTeX this file and you will be prompted for information such as your name, and the name of a test file that shows the problem. A mail message will be written to the file `latexbug.msg` which should be sent to `latex-bugs@uni-mainz.de`. (You should *always* use `latexbug.tex` to generate messages to be sent to this bug address. It is an interface to a database (The GNU GNATS problem tracking system) and can not handle messages that are not in the special format written by `latexbug.tex`.)

`latexbug.el` For users of the GNU Emacs text editor, a more convenient interface is provided by this file. It runs `latexbug.tex` automatically, and provides online help for filling in the various fields, and finally automatically mails the message to the correct address.

5 Docstrip files

As mentioned above, \LaTeX is distributed as documented sources. The files that are actually used by \TeX are extracted from these files by running `docstrip.tex`. The \LaTeX distribution contains many files with extension `.ins` that control how `docstrip` extracts each file. Most of these are never used individually, as they would just ‘unpack’ one small part of the distribution. The file `unpack.ins` is a ‘master’ installation script that calls the smaller install files in turn and so unpacks the whole distribution. Normally running \TeX on this file is the first step in installing \LaTeX . This step may be omitted however if the unpacked directory is obtained from CTAN rather than `base`. `unpack` is *exactly* the result of obtaining `base` and running \TeX on `unpack.ins`. If you have a slow machine you may prefer this route as it saves unpacking time, but conversely it requires downloading more files, so if you are transferring the files via a slow connection such as a modem then you may prefer to get the smaller ‘base’ distribution.

There are three install files that are *not* included into `unpack.ins` so you may have need to run these if you need the following features.

`autoload` Processing `autoload.ins` will generate the source file for the ‘autoload’ version of \LaTeX , `latexa.ltx`, as described in `autoload.txt`. This should be processed with `iniTeX` to create a format file to be used in place of the standard `latex.fmt`. As well as the modified format, various packages are created containing the code that has been taken out of the format. Normally these do not need to be invoked explicitly as they are loaded on demand when they are needed. Currently the following package files are produced.

`autopict` Source for `picture` mode.
`autotabg` Source for `tabbing` environment.
`autoerr` The texts of most \LaTeX error commands.
`autofss1` Less used font selection commands.
`autoout1` Code related to `\enlargethispage`.

The `autoload` format is still quite experimental, and so the range of such ‘autoloading’ packages may change with future releases.

`cmextra` Processing `cmextra.ins` installs the ‘fd’ files for the ‘concrete’ variants of the Computer Modern fonts, and also the AMS Cyrillic fonts.

`olddc` If using the Computer Modern fonts in the 8-bit ‘T1’ encoding, \LaTeX defaults to using the ‘dc fonts’. During 1995 these fonts were updated and the names of the fonts *changed*. Thus the 10 pt roman font corresponding to `cmr10` is now `dcr1000` rather than `dcr10`. The install file `unpack.ins` includes `newdc.ins` so by default \LaTeX will use the new 1995 names (dc fonts release 1.2 or later) when using T1 font encoding. If you still have the old dc fonts, then you must process `olddc.ins` to produce suitable `fd` files referring to the old names.

6 The Standard \LaTeX Classes

The general appearance of a \LaTeX document, and the specification of the commands available is specified in a *document class*. This may be further modified by loading *packages*, as described in `usrguide`. In this section I give a brief overview of the available classes in the base distribution. They all have extension `.cls` (after being unpacked from the `.dtx` source file during the installation process).

`article` ‘Article Class’. In some sense the canonical reference class against which all others are judged. This class (which is generated from the same `classes.dtx` source as `report` and `book` described below) is a mixed blessing. On one hand it provides quite a rich collection of commands for marking up documents that means that it serves well as as the basic ‘generic’ class to be used when no more suitable specific class is available. On the other hand the visual appearance of documents produced with this class is very distinctive. Many people who say they “don’t like \LaTeX ” and so use some other format such as plain, in fact are misled into believing that \LaTeX *is* this class. In fact by loading `article` and then making small adjustments one can produce very different visual designs. The class file for *Baskerville* is an example of such a non-standard class based on `article`.

However for many purposes, portability is more important than original typographical design, and in these cases the `article` class has the big advantage of being installed at all \LaTeX sites.

`report` ‘Report Class’. Very similar to `article` (and produced from the same source). The main differences being that this class has a higher level of sectioning command (`\chapter`) than is available in `article`, and the front matter is typeset differently.

`book` The `book` class is again very similar to `report` with the addition of a few extra features for controlling the front matter and back matter. It is unlikely that you would want to use this class ‘as is’ as for a book, you would almost certainly want to spend some effort (and perhaps money!) on an original design. However it can be used as a basis or example of the implementation of a \LaTeX class for book production.

`letter` This provides commands for producing one or more letters. Many sites use this as a basis for producing site-specific letter class files, for instance with a modified heading that inserts a departmental logo and address.⁵

`proc` Proceedings class. This is a variant of `article` class (and inputs the `article.cls` file when used). It defaults to two column mode and makes one or two other small adjustments. It may be used as a model for how make a class that builds on another.

`slides` The `slides` class. This class essentially provides the functionality that was formally built into `SL \TeX` . It provides a mechanism for producing pages suitable for projecting on an overhead projector. It is described in the \LaTeX book, and some people like it, however if you are making a lot of such presentations you may prefer to look at the contributed classes `seminar` (T. v. Zandt) or `foiltex` (J. Hafner). these provide alternatives to the standard class that many people find more useful.

As well as these ‘Standard Classes’ the base distribution contains a few other special purpose classes.

`minimal` This is the minimal \LaTeX class. It just sets up a text area, and a font in a single size. None of the normal sectioning or font size commands are available. This class is not intended to be used in documents, but it is often useful when testing macros as it loads very quickly.

⁵One should be able to find details of such local variants in the famous ‘local guide’.

`ltxguide` A special purpose class for the ‘ \LaTeX guides’ mentioned earlier.

`ltnews` The class file used for the ‘ \LaTeX News’ news sheets.

`ltxdoc` This class is used in all the `dtx` documentation files. It is based on the `article` class and the `doc` package, but with additional commands for documenting the \LaTeX sources. It was not conceived as a class for general use, but some people find it convenient to use it when documenting their own package files.

7 Standard Packages

7.1 Encoding Packages

One of the main features of the 2e release of \LaTeX is that it attempts to remove all ‘hard wired’ assumptions about the encodings being used, both for input and also in the fonts used for typesetting.

It maintains a strict distinction between the *Input Encoding* and the *Output Encoding*. The input encoding relates to the text that you type, this may be a standard encoding such as ASCII (The traditional 7-bit encoding) or ISO-latin-1, or a platform specific encoding such as ‘Windows ANSI’ as used on MicroSoft Windows 3.x machines. The output encoding for text fonts is usually either OT1 (The encoding devised by Knuth and implemented in the original Computer Modern \TeX fonts.) or T1 the new \TeX encoding also known as ‘Cork’ after the meeting where it was agreed.

\LaTeX maintains this separation by *always* translating input to an *Internal Encoding*. This is essentially traditional \TeX 7-bit input. This internal encoding is then translated to the encoding used in the font without reference to the original input mechanism used. Thus if you specify an input encoding that includes the character \acute{e} you may type that directly at the keyboard, and see it as a single character, however internally \LaTeX will treat this as `\'e`. If you are using 7-bit OT1 encoded fonts this command will use the `\accent` primitive to add an accute to the `e`, however if you are using T1 fonts, the existing \acute{e} will be accessed directly. Note however that the position of \acute{e} in the output encoding (T1) is typically *different* from the position of the character in the input encoding used.

`inputenc` Specifies that an 8-bit input encoding is being used. A package option should always be used which sets up the default encoding. The currently available options include `latin1`, `latin2`, `ansinew`, `cp437`, `cp437de`, `applemac`. (The two IBM codepage 437 variants differ just in one slot, the former uses β , the latter uses β .)

So typical usage (to specify ISO Latin-1 input conventions) would be:

```
\usepackage[latin1]{inputenc}
```

`fontenc` Specifies the default output encoding for text fonts. Currently the available options are OT1 and T1. So to specify that fonts in the the T1 (Cork) encoding be used in the document one would declare:

```
\usepackage[T1]{fontenc}
```

7.2 Remaining Packages in the Base Distribution

`alltt` Defines the `alltt` environment, similar to `verbatim` except that `\`, `{` and `}` retain their usual \TeX meanings.

`doc` The package defining the commands used for documenting all the \LaTeX code in the distribution.

`shortvrb` This package (really a small part of the `doc` package) defines the `\MakeShortVerb` command that allows shorthands like `|\foo|` instead of `\verb|\foo|`. This is very convenient if you are documenting \TeX or some other situation where you need to make a lot of use of short sections of verbatim text.

`exscale` For mainly historical reasons \LaTeX always uses the math extension font (used for brackets and sum and integral signs etc.) at the same size, whatever the current font size. This package modifies this behaviour so that magnified fonts are used at larger sizes. At the same time it makes the plain \TeX commands `\big`, `\bigg` etc., work as expected in conjunction with \LaTeX size commands.

`flafter` \LaTeX floats such as the `figure` and `table` environment can float *up* to the top of the current page. This means that it is possible that the figure appears before its first reference. Some publisher’s styles do not allow this. `flafter` redefines the float placement algorithm so that a float never appears before its position in the source file, so by using this package, and placing the `figure` environment after the first reference to the figure, one can ensure that figure will appear after the reference.

`graphpap` The `\graphpaper` command produces a grid for use in the `picture` environment.

`ifthen` Provides an ‘if... then... else...’ programming construct for use in \LaTeX packages. Many of the examples in ‘The \LaTeX Companion’ assume this package has been loaded.

`makeidx` Implements support for generating an index.

`pict2e` This package produces an error message to say that it has not been written. Even if it were written one would be advised to instead use the `PSTricks` package, as described in Sebastian Rahtz’ article elsewhere in this issue.

`showidx` This causes the argument of each `\index` command to be printed on the page where it occurs. See also `idx.tex` described below.

- `syntonly` Used to process a document without typesetting it. On some systems this speeds things up considerably, and so may (possibly) be useful while debugging documents.
- `tracefmt` This allows you to control how much information about L^AT_EX's font loading is displayed.
- `latexsym` Loads the special L^AT_EX symbol font and then defines commands such as `\Box` that use this font. These commands were defined by default in L^AT_EX2.09.
- `newlfont` Defines 'old' font commands to act in the 'new' way. For example it makes `\rm` essentially equivalent to `\rmfamily`. This package is not now recommended but is distributed so old documents written using the L^AT_EX2.09 version of this package still work.
- `oldlfont` A companion to `newlfont`. This package is only to be used for old documents that used the L^AT_EX2.09 package of the same name.

8 Font Definition Files

Unpacking the L^AT_EX distribution creates dozens of 'font definition files' with extension `.fd` from their documented sources (with extension `.fdd`). These map the internal L^AT_EX model of fonts on to the external file names as used on your system. Normally you never need to load these explicitly into a L^AT_EX document and they will not be considered in detail here except to say that if you obtain some new fonts from the T_EX archives, make sure to also get the related `fd` files, and install them where L^AT_EX can find them.

9 Makeindex Styles

The distribution includes three styles (with extension `.ist`) for the *makeindex* index generator. They modify the *makeindex* defaults so as to work with the special requirements of the `doc` package.

- `gind` Produces indices of command definition and use.
- `gglo` Produces 'change log' entries (using the L^AT_EX `\glossary` command rather than `\index`).
- `source2e` This style is only produced if the L^AT_EX document `source2e.tex` is processed. It is almost identical to `gind.ist` but defines 'I' to be in the series 'I–J–K' rather than 'I–II–III'. This is needed for the numbering conventions used in that document.

10 Miscellaneous Utilities and Files

- `idx.tex` Print out index entries in your document.
- `lablst.tex` Generate list of labels used in a document. You may prefer instead to have the labels show up in the margins of your drafts, in which case use the `showkeys` package from the 'tools' collection to be described later in this 'tour'.
- `ltxcheck.tex` This 'document' should always be processed after L^AT_EX has been installed. It produces no output but checks various components of the system are configured correctly for your machine type.
- `nfssfont.tex` Test file for testing a font. A more extensive font test is available if you use the `fontsmpl` package from the 'tools' collection.
- `testpage.tex` Test file for checking the accuracy of a printer. This is particularly useful to see if you need to specify any offsets to your printer driver to ensure that the printed text is correctly positioned on the paper.
- `Makefile.unx` A very simplistic template 'Makefile' for installing the L^AT_EX base distribution under UNIX. Many UNIX T_EX distributions come with far more suitable installation procedures. For example the excellent 'teT_EX' distribution allows you to install T_EX, L^AT_EX, `metafont`, `dvips`, `xdvi`, and a host of other utilities and fonts just by typing `sh install.sh`.
- `latex209.def` This file is loaded whenever a document beginning with `\documentstyle` is seen. It forces L^AT_EX into '2.09 compatibility mode' which is exceedingly slow, but a fairly accurate emulation of the old version of L^AT_EX. This enables old documents to be processed under the current system.

VIII An introduction to PSTricks, part I

Sebastian Rahtz

1 Preface

In all the questions about \TeX and drawing that appear in *fora* like `comp.text.tex`, surprisingly little attention is paid to what seems to me one of the most delightful macro packages available. This is **PSTricks**, which allows the \TeX user almost full access to the power of POSTSCRIPT, using the `\special` mechanism. In this, and the following two issues of, *Baskerville*, I will attempt to offer a survey of what **PSTricks** can do, and perhaps persuade more *Baskerville* readers to experiment.

This material is drawn from a forthcoming book by myself and Michel Goossens; we are grateful to Timothy van Zandt (the author of **PSTricks**) and Denis Girou (its best-known exponent) for the many helpful insights and examples which they have vouchsafed during the several years we have been working on the material.

This first tutorial looks at the principles of **PSTricks**, and the basic building blocks; the next will consider nodes, trees, matrices and the like, and the final part will look at customising **PSTricks** and some programming examples.

2 Introduction

PSTricks consists of a core of picture drawing primitives implemented by `\specials` which pass POSTSCRIPT through to a driver. It also contains a set of higher-level macros for particular applications. With it you can:

1. draw lines, polygons, circles and curves;
2. place and manipulate \TeX text;
3. plot data with complicated labelled axes;
4. draw nodes and connectors (including trees);
5. colour lines and fill objects;
6. define new graphical commands.

This is an extremely powerful package, and its facilities can take some time to understand. It is documented in van Zandt 1993, and its implementation is described in van Zandt & Girou 1994. Girou 1994 provides an excellent demonstration of the abilities of the package, and I am grateful to Denis Girou for permission to reproduce some of his examples below.

The package relies on the ability of a dvi driver to pass through literal POSTSCRIPT code, and know that it will interact with the \TeX text in a controlled way. The *dvips* driver provides the reference implementation, but it works with other drivers (like *Textures*) as well. The **PSTricks** installation guide explains what a driver has to be able to do.

PSTricks is not a tool for drawing just one type of diagram well, like so many of the other packages you can find on CTAN. It is a programming environment for as close a combination of \TeX and POSTSCRIPT as is possible with existing software; its strength is its modularity, extensibility, and ability to access all the power of POSTSCRIPT.

The main package, and the subsidiary ones, need access to various POSTSCRIPT header files; the user does not need to explicitly load them, but the driver must be capable of doing so. The method for loading header files, and other `\special` communication, is defined by a configuration file.

The majority of **PSTricks** is loaded in a single package, but some more complicated facilities need an extra file to be loaded:

<i>File</i>	<i>Function</i>
pst-coil	Coil and zigzag objects
textpath	Typesetting text on a path
charpath	Stroking and filling character paths
pst-plot	Data plotting
pst-3d	Three-dimensional drawing
pst2eps	Export of objects direct to EPS files
pst-node	Placing and joining nodes
pstree	Tree macros
gradient	Gradient colour fills

In the descriptions that follow, we do *not* normally indicate which file needs loading for a particular function, since this may change with new releases of the package.

Most of the commands provided will draw some kind of object at specified coordinates, which are relative to the current point in T_EX. The objects do not usually take up any space, in T_EX terms; they can either be mixed in with normal commands, or used in a picture environment to construct a whole drawing. The L^AT_EX environment corresponding to the command:

```
\pspicture*[baseline](x0,y0)(x1,y1)
```

is normally used for pictures; space is reserved in T_EX for a rectangle with corners at (x_0, y_0) and (x_1, y_1) (as in other cases, (x_0, y_0) defaults to $(0, 0)$). The * form clips graphic objects which appear outside the frame. By default, the baseline is set at the bottom of the box, but the optional argument [baseline] sets the baseline fraction `baseline` from the bottom.

I will not attempt to describe absolutely every PSTricks macro, or give examples of all the possible combinations and tricks, as this would require a large book, so we strongly commend the reader to study the published examples by Denis Girou for ideas on PSTricks programming, as well as the manual itself.

2.1 Basic PSTricks concepts

Luckily, almost all the commands have the same (complex) structure; they need some or all of the following arguments, each of which has its consistent delimiters:

<i>Type</i>	<i>Delimiters</i>	<i>Example</i>
Obligatory parameter	curly brackets	{arg}
Optional settings	square brackets	[par1=val1,...
Coordinates	parentheses	(x,y)

Many macros can have a lot of arguments, so it is useful to know that you can leave a space or new line between arguments, except those enclosed in curly braces.

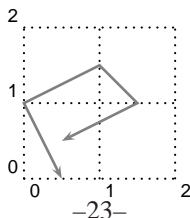
The layout of a command is

```
\command*[settings]{arrows/parameters}
                    (coordinates)
```

- The * form of the command generally means that the object being drawn is to be solid, rather than an outline;
- The *settings* are optional, and consists of a set of *key=value* pairs which over-ride, for the current object, PSTricks's drawing defaults; this is discussed below;
- Many commands which draw lines have an argument which specifies whether, and how, arrow heads are to be drawn at either end; or they need an argument like a rotation angle;
- Most objects require one or more coordinates; these consist of two numbers separated by a comma; multiple coordinate pairs each have their own set of brackets.

A simple complete example is

```
\psline[linewidth=1pt,linecolor=gray]
  {<->}(0.5,0)(0,1)(1,1.5)(1.5,1)(.5,.5)
```



where a grid has been superimposed to show the coordinate system.

Coordinates

By default the coordinate system is in units of 1cm in the x and y directions, but that can be simply overridden as we shall see in a moment. Coordinate pairs can not only be given in the default units, but also in any \TeX dimensions. This applies to all ‘quantity’ settings, so `linewidth=.5` means .5 of whatever the current unit is, but `linewidth=2pt` is an absolute size.

The command `SpecialCoor` lets you use polar coordinates, in the form $(\langle r \rangle ; \langle a \rangle)$, where r is the radius (a dimension) and a is the angle. You can still use Cartesian coordinates.

The `unit` parameter actually sets the x and y parameters which can also be set individually, so that you can scale the x and y dimensions in Cartesian coordinates unevenly.

Angles, in polar coordinates and other arguments, should be a number giving the angle in degrees, by default. You can also change the units used for angles with the command `\degrees[num]` where `num` is the number of units in a circle. Thus

```
\degrees[100]
```

could be used when making a pie chart where the data values are supplied as percentages. The command `\radians` is short for `\degrees[6.28319]`

Colouring objects

`PSTricks` comes with its own collection of colour macros, which provide a basic set of colours, and allow one to define new colour names in terms of RGB, CMYK or HSB models; however, we recommend that \LaTeX users should stick with the `color` package. The standard package and `PSTricks` can be used together by loading the package `pstcol`, which comes with the colour and graphics bundle. The `gradient` package adds facilities for gradations of colour in filled objects.

The following colours are predefined:

black, darkgray, gray, lightgray, white, red, green, blue, cyan, magenta, and yellow.

Setting graphics parameters

`PSTricks` uses a notation similar to that introduced for the `graphicx` package, *i.e.* ‘key=value’ pairs. This is used for setting a large number of graphical parameters which apply to almost all objects. These can be set in two ways:

1. On a per-object basis, with the optional parameter in square brackets; in this case the effect is local to the object with no further grouping needed;
2. Globally, using the `\psset` command. The syntax is:

```
\psset {par1=value1,par2=value2, ... }
```

Extra spaces are only allowed following the comma that separates `par=value` pairs (which is therefore a good place to start a new line if you are giving a long list). A selection of the commoner graphics parameters that can be set for objects are listed in Table 1; the first group can be applied to more or less anything, but the others only apply to a particular group.

Table 1: `PSTricks` Graphical parameters

<i>Parameter</i>	<i>Default</i>	<i>Explanation</i>
General		
<code>unit=dim</code>	1cm	
<code>xunit=dim</code>	1cm	
<code>yunit=dim</code>	1cm	
<code>linewidth=dim</code>	.8pt	
<code>linecolor=colour</code>	black	
<code>fillcolor=colour</code>	white	
<code>fillstyle=style</code>	none	Other possibilities are <code>solid</code> , <code>vlines</code> , <code>vlines*</code> , <code>hlines</code> , <code>hlines*</code> , <code>crosshatch</code> and <code>crosshatch*</code> . The <code>*</code> versions also fill the background. The <code>gradient</code> package adds the extra <code>gradient</code> style for a graded fill, and the following keys.

PSTricks Graphical parameters *cont.*

<i>Parameter</i>	<i>Default</i>	<i>Explanation</i>
<code>gradbegin=colour</code>		The starting and ending colour of a graded fill
<code>gradend=colour</code>		The colour at the midpoint.
<code>gradlines=int</code>	500	The number of lines in the graded fill. More lines means finer gradiation, but slower printing.
<code>gradmidpoint=num</code>	.9	The position of the midpoint, as a fraction of the distance from top to bottom (the <i>num</i> will be between 0 and 1).
<code>gradangle=angle</code>	0	The gradation is rotated by <i>angle</i> .
<code>hatchwidth=dim</code>	.8pt	Width of fill lines
<code>hatchsep=dim</code>	4pt	The gap between fill lines
<code>hatchcolor=colour</code>	black	
<code>hatchangle=angle</code>	45	
<code>arrows=style</code>	none	The possibilities are listed in Table 3; any symbol, or none, can be put at either end of a line
<i>Lines, curves and boxes</i>		
<code>linestyle=style</code>	solid	Other possibilities are <i>dashed</i> , <i>dotted</i> and <i>none</i>
<code>dash=dim1 dim2</code>	5pt 3pt	The black/white dash pattern for dashed lines.
<code>dotsep=dim</code>	3pt	
<code>doubleline=true/false</code>	false	Draw lines as double line, separated by <i>bordersep</i> and with colour <i>bordercolor</i> between the lines.
<code>doublesep=dim</code>	1.25 of linewidth	
<code>shadow=true/false</code>	false	A shadow is drawn at angle <i>shadowangle</i> , of depth <i>shadowsize</i> and colour <i>shadowcolor</i> .
<code>shadowsize=dim</code>	3pt	
<code>shadowangle=angle</code>	-45	
<code>shadowcolor=colour</code>	darkgray	
<code>linearc=dim</code>	0pt	The radius of arcs drawn at the corners of a box or set of line segments.
<code>framearc=dim</code>	0pt	If <i>cornersize</i> is ‘relative’, then the radius of rounded corners of framing boxes is set to <i>num</i> times the width of height of the frame, whichever is less. <i>num</i> cannot be greater than 1. If <i>cornersize</i> is ‘absolute’, <i>num</i> sets the radius of arcs of rounded corners.
<code>cornersize=relative/absolute</code>	relative	
<i>Text frames</i>		
<code>framesep=dim</code>	3pt	The gap between a frame and the enclosed text
<code>boxsep=true/false</code>	true	Whether the T _E X box that is produced includes the size of the frame itself or not
<i>Dots</i>		
<code>dotstyle=style</code>	*	The possible styles are listed in Table 4
<code>dotsize=dim num</code>	2pt 2	The diameter of a circle or disc is <i>dim</i> plus <i>num</i> times the current linewidth

3 The graphic objects

We list in Table 2 the most common basic objects which PSTricks can draw; they all, with the exception of the text-framing commands, take up no T_EX space, and so should be used inside a `pspicture` environment when creating a free-standing picture. In almost every case, an initial first coordinate can be omitted, and defaults to 0,0. Lines and open curves can optionally be terminated with various symbols, and it is this that the `arrows ... styles` set.

PSTricks has two powerful commands for positioning (and rotating, if necessary) something, including normal L^AT_EX material; they are analagous to L^AT_EX's basic `\put` command. The more common command is

```
\rput *[/refpoint]{angle}(x0,y0) {stuff}
```

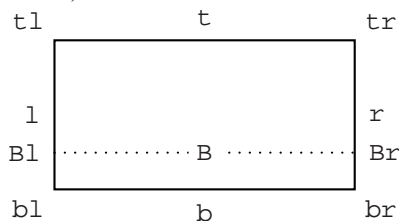
The `*` form puts a `\psframebox` around *stuff*, which effectively blocks out anything underneath. *stuff* is rotated by *angle* if the argument is present; the *angle* it can be preceded by a `*`, which has the effect of undoing all rotations in outer calls to `rput`. This is needed when placing text labels, to make it easy to place them consistently in the right orientation. Since many rotations are in steps of 90, you can use the following letter abbreviations

Letter	degrees	Letter	degrees		
U	<i>Up</i>	0	N	<i>North</i>	*0
L	<i>Left</i>	90	W	<i>West</i>	*90
D	<i>Down</i>	180	S	<i>South</i>	*180
R	<i>Right</i>	270	E	<i>East</i>	*270

refpoint describes the reference point of *stuff*, and this reference point is what is placed at (x,y) . By default, it is the center of the box. This can be changed setting *refpoint* to one or two of the following

Horizontal	Vertical
l Left	t Top
r Right	b Bottom
	B Baseline

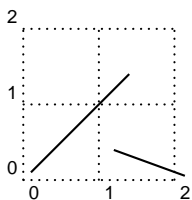
The following diagram shows the reference point represented by the various combinations (the dotted line is the baseline):



```
\rput *[/refpoint]{rotation}(x, y) {stuff}
```

It is important to realize that `\rput` resets the $(0,0)$ point to the chosen coordinate, as shown in the following example:

```
\rput (.1, .1){\psline(0,0)(1.3,1.3)}
\rput {-20}(1.2, .4){\psline(0,0)(1,0)}
```



Note that rotation was applied last of all. Because it is very often a requirement to *put* a label next to some object, a variant of `\rput` is provided:

```
\uput *{labsep}[refangle]{rotation}(x, y) {stuff}
```

which places *stuff* in the direction *angle*, at a distance of *labsep* from (x,y) . *labsep* defaults to 5pt. As before, since angles are often in steps of 45, letter abbreviations are provided for common cases:

Letter	degrees	Letter	degrees		
r	<i>right</i>	0	ur	<i>up-right</i>	45
u	<i>up</i>	90	ul	<i>up-left</i>	135
l	<i>left</i>	180	dl	<i>down-left</i>	225
d	<i>down</i>	270	dr	<i>down-right</i>	315

Table 2: PSTricks basic drawing commands

<code>\parabola*[settings]{arrows}(x_0, y_0) (x_1, y_1)</code>	Draw a parabola that starts at (x_0, y_0) , passes through (x_0, y_0) , and whose maximum or minimum is (x_1, y_1)
<code>\psarc*[settings]{arrows}(x, y){radius} { angleA } { angleB }</code>	Draw a circle segment between <i>angle1</i> and <i>angle2</i> (counter-clockwise);
<code>\psarcn*[settings]{arrows}(x, y){radius} { angleA } { angleB }</code>	As <code>\psarc</code> , but the arc is drawn <i>clockwise</i>
<code>\psbezier*[settings]{arrows}(x_0, y_0)(x_1, y_1) (x_2, y_2) (x_3, y_3)</code>	Draw a Bezier curve with four control points
<code>\psccurve*[settings]{arrows}(x_1, y_1) ... (x_n, y_n)</code>	Draw a closed curve between the points
<code>\pscharclip*[settings]{text} ... \endpscharclip</code>	Set the clipping path to the character shapes
<code>\pscharpath*[settings]{text}</code>	The <i>text</i> obeys the PSTricks linestyle and fillstyle commands; this is only effective if the font used is a POSTSCRIPT Type1 font
<code>\pscircle*[settings](x_0, y_0){radius}</code>	Draw a circle with the center at (x_0, y_0)
<code>\pscirclebox*[settings]{text}</code>	Draw a circle around the text
<code>\pscoil*[settings]{arrows}(x_0, y_0)(x_1, y_1)</code>	Draw a 3D coil from (x_0, y_0) to (x_1, y_1)
<code>\psCoil*[settings]{angle1} { angle2 }</code>	Draw a coil horizontally from <i>angle1</i> to <i>angle2</i>
<code>\pscurve*[settings]{arrows}(x_1, y_1) ... (x_n, y_n)</code>	Draw an open curve through the points
<code>\psdblframebox*[settings]{text}</code>	Draw a double box around the text
<code>\psdiabox*[settings]{text}</code>	Draw a diamond around the text
<code>\psdiamond(x_0, y_0)(x_1, y_1)</code>	Draw diamond centred at (x_0, y_0) with the half width x_1 and height y_1
<code>\psdots*[settings](x_1, y_1) (x_2, y_2) ... (x_n, y_n)</code>	Draw dot at each coordinate
<code>\psecurve*[settings]{arrows}(x_1, y_1) ... (x_n, y_n)</code>	Draw an open curve, but omitting the last and first points
<code>\psellipse*[settings](x_0, y_0)(x_1, y_1)</code>	Draw an ellipse with centre at (x_0, y_0) , and horizontal and vertical radii of x_1 and y_1
<code>\psframe*[settings](x_0, y_0)(x_1, y_1)</code>	Draw a rectangular frame with corners at (x_0, y_0) and (x_1, y_1)
<code>\psframebox*[settings]{text}</code>	Draw a box around the text
<code>\psgrid(x_0, y_0)(x_1, y_1)(x_2, y_2)</code>	Superimpose a grid with corners at (x_1, y_1) and (x_2, y_2) , labelled on the axes starting from (x_0, y_0)
<code>\psline*[settings]{arrows}(x_0, y_0)(x_1, y_1) ... (x_n, y_n)</code>	Draw a line through a series of coordinates
<code>\psovalbox*[settings]{text}</code>	Draw an oval around the text
<code>\pspolygon*[settings](x_0, y_0)(x_1, y_1) (x_2, y_2) ... (x_n, y_n)</code>	Draw a line through the coordinates, and then close the path to make an object that can be filled
<code>\psshadowbox*[settings]{text}</code>	Draw a box around the text, with a shadow
<code>\pstextpath[pos](x, y){graphics object} {text}</code>	The <i>text</i> is drawn along the line defined by the <i>graphics object</i> . <i>pos</i> determines how the text relates to the path; by default (l), it starts at the beginning of the path; c will center the text along the path and r will make it finish at the end of the path. (x, y) provides an offset for the text in relation to the path. By default it is offset above the line by <i>.7ex</i> . <i>This macro, and \pscharclip, are not guaranteed to work with every dvi to POSTSCRIPT driver!</i>
<code>\pstriangle(x_0, y_0)(x_1, y_1)</code>	Draw isocetes triangle with base centred at (x_0, y_0) , width x_1 , and height y_1
<code>\pstribox*[settings]{text}</code>	Draw a triangle around the text
<code>\pswedge*[settings](x_0, y_0){radius} { angle1 } { angle2 }</code>	Draw a wedge segment between <i>angle1</i> and <i>angle2</i> (counter-clockwise)
<code>\pszigzag*[settings]{arrows}(x_0, y_0)(x_1, y_1)</code>	Draw a zigzag from (x_0, y_0) to (x_1, y_1)

Value	Example	
-		None
<->		Arrowheads.
>-<		Reverse arrowheads.
<<->>		Double arrowheads.
>>-<<		Double reverse arrowheads.
-		T-bars, flush to endpoints.
* - *		T-bars, centered on endpoints.
[-]		Square brackets.
(-)		Rounded brackets.
o - o		Circles, centered on endpoints.
* - *		Disks, centered on endpoints.
oo - oo		Circles, flush to endpoints.
** - **		Disks, flush to endpoints.
c - c		Extended, rounded ends.
cc - cc		Flush round ends.
C - C		Extended, square ends.
<->		T-bars and arrowheads.
<*-> *		T-bars and arrowheads, flush.

Table 3. PSTricks line terminators

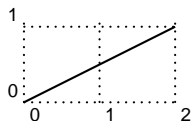
Style	Example
*	
o	
+	
x	
asterisk	
oplus	
otimes	
triangle	
triangle*	
square	
square*	
diamond	
diamond*	
pentagon	
pentagon*	

Table 4. PSTricks dot styles

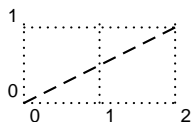
4 Examples of basic graphic objects

The following examples demonstrate some of the PSTricks building blocks, and the use of the graphics parameters. We must remember that all these simple objects take up no space; the surrounding `pspicture` defines the space $\text{T}_\text{E}\text{X}$ is to leave, but within that, we are drawing entirely by coordinates.

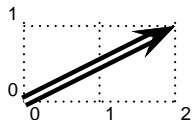
```
\psline(2,1)
```



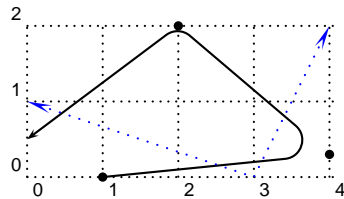
```
\psline[linestyle=dashed](2,1)
```



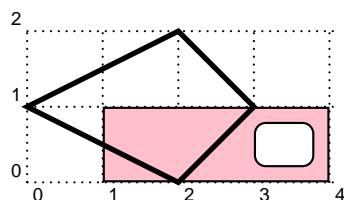
```
\psline[linewidth=0.6mm,doubleline=true,
doublesep=0.5mm]{->}(2,1)
```



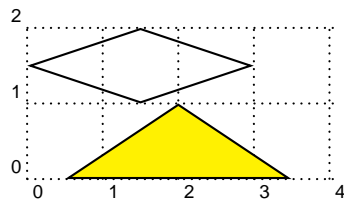
```
\psline[linearc=0.25,showpoints=true]{->}%
(1,0)(4,0.3)(2,2)(0,0.5)
\psline[linestyle=dotted,linecolor=blue,
arrowlength=3]{<->}(0,1)(3,0)(4,2)
```



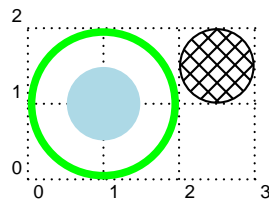
```
\psframe[fillstyle=solid,fillcolor=pink]
(1,0)(4,1)
\psframe[fillstyle=solid,fillcolor=white,
framearc=0.5](3,0.2)(3.8,0.8)
\pspolygon[linewidth=0.7mm,dimen=inner]
(0,1)(2,2)(3,1)(2,0)
```



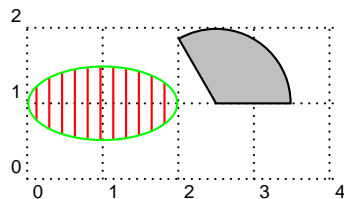
```
\psdiamond(1.5,1.5)(1.5,0.5)
\pstriangle[fillstyle=solid,
fillcolor=yellow](2,0)(3,1)
```

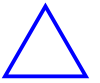


```
\pscicle[linewidth=1mm,linecolor=green]
(1,1){1}
\pscicle[linestyle=none,fillstyle=solid,
fillcolor=lightblue](1,1){0.5}
\pscicle[fillstyle=crosshatch](2.5,1.5)
{0.5}
```

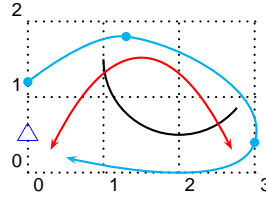


```
\psellipse[linecolor=green,
fillstyle=vlines,hatchangle=0,
hatchcolor=red](1,1)(1,0.5)
\pswedge[fillstyle=solid,
fillcolor=lightgray]
(2.5,1){1}{0}{120}
```

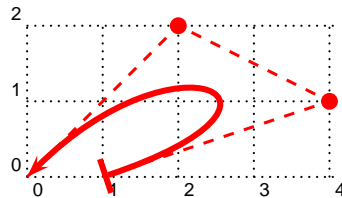




```
\psdots[linecolor=blue,dotstyle=triangle,
        dotscale=2](0,0.5)(1,2)(2.8,1.5)
\pscurve[linecolor=cyan,showpoints=true]
  {->%
    (0,1.2)(1.3,1.8)(3,0.4)(0.5,0.2)
\psarc(2,1.5){1}{180}{320}
\parabola[linecolor=red]{<->%
  (0.3,0.3)(1.5,1.5)
```



```
\psbezier[linewidth=0.8mm,linecolor=red,
          showpoints=true]{|->%
  (1,0)(4,1)(2,2)(0,0)
```



5 Mixing text and graphics

When we come to consider text, the situation is rather different; now the size of objects is determined by the size of the enclosed text. Here $\text{T}_\text{E}\text{X}$ is aware of the space used, so successive objects are placed as if they were letters, and coordinates are not used.

```
\psframebox{The Buck Stops Here}
```

The Buck Stops Here

```
\psframebox[fillstyle=solid,
            fillcolor=black]
  {\bfseries\color{white}\LARGE
  Beware of The Dog}
```

Beware of The Dog

```
\psframebox{The dragon}
\psframebox
  {\psframebox[linecolor=green]{ate}
  \psframebox[linecolor=blue]
    {\psframebox[linecolor=red]{the women}
    and
    \psframebox[linecolor=red]{children}}}
```

The dragon

ate

the women

and children

```
\psshadowbox[fillstyle=solid,
             fillcolor=yellow]
  {\color{red}\begin{tabular}{c}
  Chapter 1\We go to sea
  \end{tabular}}
```

Chapter 1
We go to sea

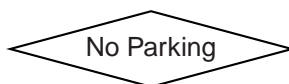
```
\pscirclebox[doubleline=true]
  {\bfseries STOP!}
```



```
\psdblframebox[linecolor=green]
  {\color{red}All Hail Caesar!}
```

All Hail Caesar!

```
\psdiabox{\sffamily No Parking}
```



```
\pstribox[shadow=true,fillstyle=gradient,
  gradbegin=green,gradend=red]
  {\color{white}\Large$\Omega$ }
```

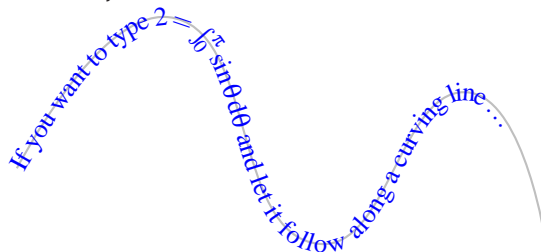


```
\psovalbox[linecolor=red]
  {\color{blue}Today's Menu}
```

Today's Menu

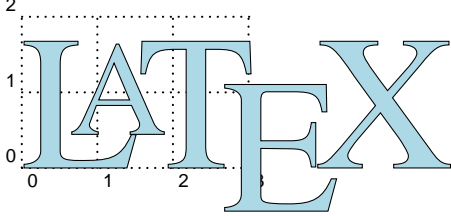
POSTSCRIPT aficionados will be aware that text in POSTSCRIPT is treated just like any other graphical object (this was one of the great revolutions introduced by the language), and can be manipulated. The following examples demonstrate the `\pstextpath` macro, which has two parameters: a graphic object, and some text. The text is made to fit the graphical object; only simple text is allowed, but that includes maths.

```
\psset{linecolor=lightgray}
\pstextpath
  {\pscurve(-4,-2)(-2,0)(0,-3)(2,-1)(3,-3)}
  {\color{blue} If you
  want to type  $z = \int_0^{\pi} \sin \theta$ ,
 $\mathrm{d}\theta$  and let it follow
  along a curving line \ldots}
```



It is also possible to set text as if it were a graphic, with the fill, colour and line properties. This is done with the `\pstextpath` macro; it has limitations (for instance, it cannot be mixed with `\pstextpath`), but it is still a useful tool for special occasions:

```
\pscharpath[fillstyle=solid,
  fillcolor=lightblue,
  linewidth=.4pt]
  {\fontsize{72}{72}\selectfont \LaTeX}
  2
  1
  0
  0 1 2
```



Admirers of the distinctive style of Dorling Kindersley publications may like to try the following effect:

```
\begin{pspicture}(-4,-3)(4,1)
\psset{fillstyle=solid,shadow=true,shadowangle=0}
\DeclareFixedFont{\babyfont}{T1}{ptm}{m}{n}{2cm}
\DeclareFixedFont{\wordfont}{T1}{ptm}{m}{n}{1.5cm}
\def\Cc#1#2{\pscharpath[fillcolor=#1]{#2}}
\bfseries
\rput(0,0){\babyfont
\Cc{red}B\Cc{green}A\Cc{yellow}B%
\Cc{red}{Y'}\Cc{blue}S}
\rput(0,-2){\wordfont\Cc{blue}{WORLD}}
\end{pspicture}
```

B**A****B****Y**'**S**
W**O****R****L****D**

The third of these tools which treat text like a graphic is `\pscharclip`; this takes a parameter of some text, and its effect is terminated by `\endpscharclip`. Any objects drawn inside this group are clipped to the shape of the letters.

```
\newcounter{myN}

\DeclareFixedFont{\bigsf}{T1}{phv}{b}{n}{1.3cm}
\DeclareFixedFont{\tinyrm}{T1}{ptm}{m}{n}{2mm}
\setcounter{myN}{110}
\begin{pspicture}(0,0)(8.2,1)
\pscharclip[linecolor=yellow,fillstyle=solid,
  fillcolor=red]
  {\rput[bl](0,0){\bigsf CHOCOLATE}}
\rput[t]{90}(0,0)
  {\vbox
  {\hsize=2cm \offinterlineskip
  \tinyrm\color{black}
  \loop
  \addtocounter{myN}{-1}
  \ifnum\value{myN}>0
  nuts and raisins
  \repeat}}
\endpscharclip
\end{pspicture}
```


<code>\psshadow*[settings]{text}</code>	Draw a shadow on the text
<code>\pstilt*[settings]{degrees}{text}</code>	Place <code>text</code> tilted
<code>\ThreeDPut(x₀,y₀,z₀){object}</code>	Place <i>object</i> at coordinate x_0, y_0, z_0 , displayed according to the current viewpoint

Table 5. PSTricks 3D commands

Parameter	Default	Explanation
<code>Tshadowsize=size</code>	1	Length of shadow
<code>Tshadowcolor=colour</code>	lightgray	Colour of shadow
<code>Tshadowangle=angle</code>	60	Angle of shadow
<code>viewpoint=x y z</code>	1 -1 1	position of the observer looking at the object origin
<code>normal=x y z</code>	0 0 1	A vector orthogonal to the plane of the 2D object, which specifies its position in 3D space
<code>embedangle=angle</code>	0	The rotation around the axis through the reference point of the object in the direction of the positioning vector

Table 6. PSTricks 3D graphical parameters

0-5-25

6 Working with a third dimension

Later versions of PSTricks offer some experimental facilities for viewing objects in three dimensions. Two-dimensional objects can be projected in a 3D coordinate system, and arbitrary viewpoints established. At the present time, PSTricks does not support true 3D solid objects, perspective projection, hidden-line removal, or lighting of objects, so the usefulness of this part of the package is limited. However, with some patience, pleasing effects can be obtained (well demonstrated, as usual, by Girou 1994). Table 5 lists the new commands, and Table 6 describes the extra graphics parameters which apply to them.

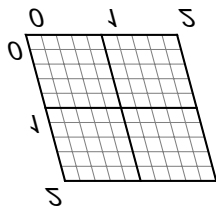
There are two high-level ‘3D’ commands, and one general tool. The high-level commands are `\psshadow`, which attaches a shadow to some text, and `\pstilt`, which angles an object into the third dimension.

```
\psshadow[Tshadowangle=45,
Tshadowsize=2.5]{%
\LARGE\bfseries Words with a shadow}
```



```
\rput(0,1){\pstilt{45}{I Feel Ill!}}
\rput(0,0){\pstilt{-75}{\psgrid(2,2)}}
```

I Feel Ill!



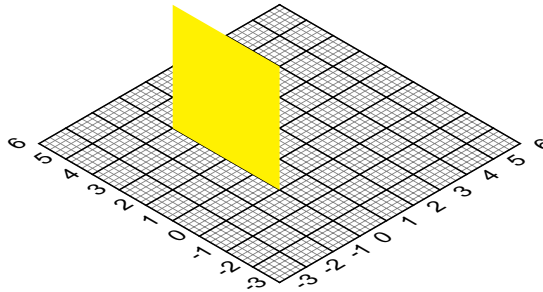
The general macro is `\ThreeDPut`, which places any PSTricks object at a point in 3D space. This will almost always set the `normal` graphics parameter, which sets the vector which will be orthogonal to the plane of the object. To take a simple example, suppose we want to draw a house, with a reference origin at the left-hand end of the front wall; the left side wall would be drawn as follows:

0-6-28**Figure 1.** Different views of a dice cube

```

\psset{unit=.5cm}
\psset{viewpoint=-1 -1 1}
\ThreeDput[normal=0 0 1]{\psgrid(-3,-3)(6,6)}
\ThreeDput[normal=-1 0 0](0,4,0){
  \psframe*[linecolor=yellow](4,4)}

```



The viewpoint is from ‘up, back and to the left’. When we draw a whole cube, it is important to realize that PSTricks does not check which side hides which; the last side drawn will wipe out others drawn earlier, regardless of the fact that it is ‘behind’ them in 3D space. When drawing the different views, we have to give the sides in different orders. Fig. 1 shows this problem, since when we show the ‘underneath’ well, we have to be sure to draw all the six faces in the right order for the viewpoint. The attempt to provide ‘lighting’ on the die is done simply by colouring three faces in a lighter colour. To attempt formal ray-tracing from a light source would be considerably beyond the scope of this package!

Following another example by Denis Girou, we can use the `\ThreeDput` macro to generate the appearance of raised text:

```

\begin{pspicture}(-3,0)(3,5.5)
\psset{unit=.5cm}
\font\bigfont=phvb8t at .8cm
\psset{viewpoint=-1 -1 1.5,normal=0 0 1}
\rput(0,0){\multido{\n=0+0.07}{12}
{\definecolor{AColor}{rgb}{1, \n, \n}
\ThreeDput(\n,\n,0){\pscircle[%
linestyle=none,fillstyle=solid,
fillcolor=AColor](5,5){6}}}}
\psset{linestyle=none}
\rput(0,0){\multido{\n=0+0.07}{12}
{\definecolor{AColor}{rgb}{1, \n, \n}
\ThreeDput(\n,\n,0){\bigfont
\pstextpath[c]{%
\psarcn(5.5,5.5){4}{180}{0}}{%
\color{AColor}Happy Christmas}
\pstextpath[c]{%
\psarc(5.5,5.5){4}{180}{0}}{%
\color{AColor}\TeX\ Lovers}}}}
\end{pspicture}

```

0-6-30**References**

- Girou, D. 1994. Présentation de PSTricks. *Cahiers GUTenberg*, **16**, 21–70.
van Zandt, T. 1993. PSTricks User’s Guide. Unpublished documentation with the software, version 0.93a.
van Zandt, T. & Girou, D 1994. Inside PSTricks. *TUGboat*, **15** (3) September, 239–246.

IX Walnut Creek T_EX CDROM

1 Part 1: A user's view

J. M. Bowsher

I first heard of the *Walnut Creek* T_EX CDROM (actually a pair of disks) when my faithful hardware and software suppliers *Systems Solutions* sent me a list of their latest CDROMs. I asked Peter Abbott about it when I renewed my subscription to UKTUG, and he surprised me by sending me a copy to review and asking me to write down my opinions as one who works alone and is not connected to a network.

The packaging is the usual for a pair of CDs, but the first thing I noticed was the depressing little message on the first disk “type GO to start”; this warned me that the suppliers had IBM PC users only in mind. Fortunately, the disks are in ISO 9660 format, so my Atari ST was able to read them with no problems — I expect that other platforms would have no difficulty either. However, I found when I accessed the first ‘readme’ file that *Walnut Creek* have used a format for informative text files which required me to use my word processor to read them; the ROM based ASCII text reader I normally use placed all the text, apart from the first couple of lines, off the right hand edge of the screen. *Walnut Creek* provide in the root partition of the first disk what they claim is a powerful and versatile viewer, but they have failed to help workers on platforms other than the IBM PC — there are no viewing programs for other systems.

The root partition ‘readme’ file told me that the CDROM contains, *inter alia*, a snapshot of the CTAN archive taken between February and March 1995; it would therefore seem to be of considerable utility to the T_EX community; I look forward to reading the second part of this review by a T_EX expert. The first thing I did after reading that first ‘readme’ file was rush to the ‘systems’ folder and see what was in it. The following systems are included: Acorn Archimedes, Amiga, Atari, Common T_EX, Knuth (Knuth’s original sources), Mac, MSDOS, NT, OS2, Unix, VM-cms, VMS, web2c. I next discovered — and this, I feel, is a very serious drawback — that at least four different compression algorithms are used on this pair of disks. After only a few minutes searching around, I found ARC, LZH, ZIP and ZOO extensions, but no decompression facilities were provided in any of the systems sub-folders I looked through. I feel that the suppliers should have placed in each of the ‘systems’ sub-folders, a set for *that* system of executable decompression programs for every compression method used. Those on systems using 8+3 character file names should be aware that some of the files on this CDROM do not use this naming convention.

Obviously, I opened the Atari sub-folder first, but had to search through my floppy storage box to unearth a ZOO decompressor (I have never before had occasion to use ZOO) before I could discover that it contained version 7.00 of the Christoph Strunk T_EX shell I use, and also a late version of the alternative Lindner shell. I installed version 7.00 on my machine as it is very pleasant to use English commands, but was sad to see that much of the supporting documentation is still in German. There are also several folders of useful utilities including, for example, many `dvi` → printer drivers. Who knows, one of the 600 dpi laser printer drivers may even work!

A quick survey of other systems revealed that they seem to be provided for in a similar manner. I couldn’t check that things worked, of course, but there were folders full of what looked like comparable material to that I had gone through in my native folder. I leave the discussion of the other, more basic, T_EX material to my fellow reviewer. I noticed lots of useful stuff; for example, hundreds of fonts (including a way of using printer resident Hewlett Packard laser fonts) in addition to the expected `cm`, `dc` etc.

Thus my opinion is that this CDROM would be a versatile addition to almost anyone’s collection provided they have the patience (if not using an IBM PC) to sort out decompression and file location problems. My file finding accessory deserved a much needed rest after working on this CDROM for a few hours!

There have been four CTAN CDROM collections that we know of. There were sections on CD even before the formal ‘opening’ of the archive setup by George Greenwade in 1993 at Aston, thanks to Prime Time Freeware’s offering, but the first full set was that issued by PTF in mid 1994. They planned a yearly update, but we are still waiting for a second edition; meanwhile, Walnut Creek issued their two disk set in mid 1995, and recently the German T_EX Users Group, Dante, has produced another one, which has not yet been seen in the UK (it is for ‘members only’, in that delightfully open way Dante has).

When the Prime Time Freeware CD came out, it was criticized for being all compressed using ZIP (which was better than the myriad methods used on the 1992 disk), which meant that all of L^AT_EX, for instance, was in one giant archive. The Walnut Creek CD is better, because everything is uncompressed, as you would find it on the archives (from a year ago — and much has changed since then), but the downside is that because there are two disks, what you want is always on the other one. Apart from that, it is a great convenience having a CTAN snapshot at home, or in an unconnected office, and I congratulate Walnut Creek for producing it.

Is a CTAN dump a good idea, however? The most successful T_EX CD is the 4AllT_EX offering from NTG, which allows DOS users to run straight from a mounted disk if they like, and this can never be the case with a CTAN dump. Do users want an archive, or a usable file system? With the increasing use of compound `docstrip` sources for L^AT_EX packages, we need to *install* material before we use it. The standard directory structure recently described by the TUG TDS working group is a vital stage in making the dream of a complete, usable, T_EX file system on a single CD come true, and I expect to see an offering this year.

We need both CTAN dumps for the unconnected, and TDS standard file systems for ‘plug-n-play’ers. CTAN will continue to provide the raw material, but I hope that CD producers will spend more on adding value, and understanding the content. The Walnut Creek CD is useful, but not very useful for the complete beginner; it has not been done in consultation with the CTAN maintainers, which is a pity.

X Malcolm's Gleanings

Malcolm Clark
m.clark@warwick.ac.uk

1 Out of METAFONT comes forth riches

Dougie Henderson, who some may recall as the author of the first practical implementation of METAFONT for the PC (marketed by Personal T_EX), who worked with Blue Sky Research for several years (the creators of *Textures*, still the finest implementation of T_EX on a personal machine, and a version of METAFONT for the Mac), and who was a member of the TUG board for many years, left the T_EX world a few years ago to brew beer. Microbreweries are a hot item in the US: Dougie's brewery *Hair of the Dog* won a rather coveted award for its *Adambier* – described as 'A very full bodied, "take no prisoners" beer'. The *Malt Advocate Awards Program* selects outstanding products and individuals in the beer and whisky industries. *Hair of the Dog* won 'Domestic Beer of the Year': to give you some idea of their level and appropriateness, their 'Import Whisky of the Year' was a 16-year old *Lagavulin* (by coincidence a distillery Dougie and I visited in 1993), the 'Import beer of the year' the wonderful Belgian *Duvel*, and for the 'Industry Leader' they chose Michael Jackson, who has probably done more than any other individual to spread the word of fine malt whisky and 'craft' beers. I think this gives hope to us all: there *is* life after T_EX!

2 Indefatigable

Readers of the *Times Higher Education Supplement* will have seen that Allan Reese continues his guerilla war of attrition on the detractors of T_EX. In the January edition of the *Multimedia* supplement he comments on an article in the December supplement, correcting some misunderstandings and misapprehensions. Another accolade to that man.

3 Guesting

It was pointed out that there were no gleanings in *Baskerville* 5(5) because I was busy trying to knock an edition of *TUGboat* into shape. As part of Michel Goossens' scheme to revitalise TUG, it was thought imperative to try to get *TUGboat* appearing regularly, if not on time. One ploy was to invite (or instruct) 'guest editors', thus relieving the usual crew from some of the work, and perhaps achieving the throughput needed.

I still believe in the Internet (an act of faith, on a par with religion – like T_EX itself). But my faith was a little shaken by the experience. Since I do not have all the various macros, classes and paraphernalia on my local machine (far less my machine at home), I decided to do most of the editing locally, FTP it to the *TUGboat* machine at SCRI in Florida, use the installation there, ship the *dvi* back by FTP and view or print locally. This strategy at least ensured that I was using the same files that would be run by the editorial team for the finished copy. It also ensured that I minimised network traffic. Text files are not too large, and *dvi* is also fairly compact. I had found that trying to edit over the Internet, though possible, was painful. Even a rather dumb, efficient, editor like *vi* could get badly out of synchronisation. It was far easier to edit locally. Maybe it really took the same amount of lapsed time, when you include the transfers, but the wear and tear on my nerves was minimised. However, despite all this, I still found that the only practical times to do the work was either Saturday morning up to about 1 or 2 in the afternoon, or Sunday mornings up to a similar time. The melt-down or brown out of the Internet seems to hit about lunch time GMT, even at weekends. Sigh.

However, the edition was eventually completed to my satisfaction. There were a few page breaks that could have been improved, but given the complexity of the problem, with lots of floating figures (always the bane of L^AT_EX), I was quite happy. I would have liked to impose my own stamp on it by adopting ragged right throughout, but it is a rather awkward interposition, especially for a periodical which has been 'designed' for justified margins. You just cannot be sure that ragged right will be appropriate in all circumstances. To do that you need to redesign from scratch, the way

the previous guest edited *TUGboat*⁶ was. Now I appreciate just how excellent that edition was, although I was dubious of the design when I first saw it, before my appreciation and understanding of the issues matured.

It is interesting to ask what a guest editor does. I don't know that I had a very clear brief. I decided first to assemble the papers, using some which had been submitted to the annual conference, a couple which had appeared elsewhere (in *Baskerville* and *Cahiers GUTenberg*) and another I invited. Between them I think this gave a reasonable balance, though I was very conscious that there were some areas that needed to be filled out more. My early plan had been to ensure that these articles hung together, referred to one another appropriately, were consistent in tone, used much the same acronyms and logos. In other words, that they blended together in terms of their appearance, if not the message of the individuals' writing. Then I discovered I should worry about line and page breaks – how it would appear in *TUGboat*. This is easily the most time-consuming part. Introducing deliberate breaks has a tendency to alter everything that comes after; and you must also run all the articles together since in standard *TUGboat* form the next article starts when the previous one ends. They don't start at the top of a new page. In the end I was shipping the entire `dvi` file across the 'fat pipe'. The front matter, end matter and page headings were the responsibility of the rest of the editorial team. After all, I didn't know how much front matter there was and couldn't predict page numbers. That appears to have been the source of a problem. Somehow, when the *TUGboat* came to be printed it had the headers at the same position on each page. Normally they would be left- and right-page oriented. I doubt that many people would have noticed this, or, if they had, they would have assumed that it was a design quirk. Unfortunately Barbara Beeton saw fit to announce it loudly to the world as a flaw. The edition took several months from my 'finishing' it to going to print. It seems to have been thoroughly re-edited by what was once termed the GNAW.⁷ The irony is that it was just this process which Michel sought to eliminate. One suspects that there are mightier forces of conservatism and inertia arrayed against him than he knows. Sigh.

It has been interesting to read the several accounts of the expected rescheduling of *TUGboat*. Compare the following:

- 16(1): “You will receive 16(2) and 16(3) before the end of the calendar year; the December issue, 16(4) will be out in early 1996.”
- 16(2): “... the last issue of 1995 ... you will hopefully receive it in the first half of January.”
- 16(3): “... four *TUGboat*'s on our member's desks before Christmas 1995” (i.e. 15(4) to 16(3)). In passing, some attention to the use of the apostrophe is needed here, unless we really do have only one member. I personally would worry more about grammar than running heads.

I received 16(2) in December or so, and 16(3) arrived in the first week of January (so a very close miss): 16(4)? Over to you Michel.

⁶Volume 7(1), guest edited by David Kellerman and Barry Smith, designed by Martha Gannett

⁷Only a small prize for the first correct expansion of this scurrilous and sexist acronym.

XI Treasurer's Report

Peter Abbott
Treasurer and Membership Secretary, UKTUG
Peter.Abbott@tex.ac.uk

1 Membership Matters

This *Baskerville* is the first issue for 1996 and is being sent to all 1996 and 1995 members (who have not yet renewed). Unless you renew your membership this is the last issue that you will receive. If you decide that you do not wish to renew and have not notified me, please do so. I would be interested in the reason for not renewing. The only way that UKTUG can improve its services is by feedback from members.

Everyone who has renewed, at the time of writing (8th February), should have received an acknowledgement of the renewal. I always acknowledge renewals and would like to hear from anyone who has not received an acknowledgement.

For the record the membership statistics at 8th February are

	1995	1996
UKTUG	66	33
Honorary	1	2
UKTUG (Student)	2	2
Full TUG and UKTUG	99	70
Basic TUG and UKTUG	6	4
Full TUG and UKTUG (Student)	4	4
Basic TUG and UKTUG (Student)	1	0
Institutional	5	2
Full TUG	2	0
Basic TUG	0	0
TUG (Student)	0	0
UKTUG (paid via TUG)	3	0
Total	188	116

No doubt by the time that you read this they will have changed.

The two Honorary Members are Don Knuth and Malcolm Clark. I have not yet been notified of any renewals from USA but there is at least one person who has set the process in motion.

2 Software distributions

Turning to other matters for which I have a responsibility, I am pleased to report that the original 50 copies of the 3rd edition of the 4All \TeX CDROM have been sold and further supplies have been ordered.

The December 95 release of L \TeX 2 ϵ has not yet been distributed but will be sent to all who paid the £30/£5 fee in 1995. Andrew Trevorrow has indicated that Oz \TeX 2.0 will be available later this year for distribution to paid up members. I have no further information, at present, on updates for Em \TeX .

The book discounts have been increased to 20% (except for those which include VAT-able elements). A revised price list should be included with this issue. In any case, if you are interested in any book from the lists, please contact me and I can confirm the price etc. All prices quoted include delivery to your address as on the membership form. I would be grateful if you could notify me when books are delivered as it eases the administrative problems with invoices.

My address, phone and fax appear in the masthead.