Baskerville

The Annals of the UK TEX Users' Group **Editor: Sebastian Rahtz** **Vol. 4 No. 5**

ISSN 1354–5930 **October 1995**

This reprint of *Baskerville* is set in Times Roman, with Computer Modern Typewriter for literal text; the source is archived on CTAN in `usergrps/uktug`.

Please send UKTUG subscriptions 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

1 New CD supplies for TEX junkies

The NTG 4TEX CD ROM has been issued in a second edition, due to popular demand! UKTUG can now supply copies at £25 each, including postage and packing. We remind you (and those who have already purchased the CD) that you must pay the £50 shareware fee for 4DOS if you use the 4TEX shell for more than a few weeks.

We can also obtain the Prime Time Freeware CD 'TEXcetera' if members have problems finding it. Contrary to Malcolm 'Grizzler' Clark's remarks in this issue, its remarkably easy to use *both* these CDs, so don't be put off…

A *Baskerville* articles needed

Now it's time to write *Baskerville* articles for 1995. Get your writing hats on, please, to 'delight fellow TEX users with your words of wisdom'. *Please note the following copy deadlines*:

Issue	*Submit material for publication*	*Submit last-minute notices*	*Anticipated posting date*
4.6	Dec 2	Dec 2	Dec 9
5.1	Feb 13	Feb 17	Mar 3
5.2	Apr 10	Apr 14	May 5
5.3	Jun 5	Jun 9	Jun 23

Each issue of *Baskerville* has a special theme, although articles on any TEX-related subject are always welcome. The last issue of 1994 will be a bumper Christmas special, containing a totally revised and enriched 'TEX Frequently-Asked Questions'. Contributions on the themes for 1995 are eagerly solicited:

Baskerville 5.1 will (as part of a new policy) contain the group's AGM report, list of activities, introduction to TEX, resource directory etc; introductory articles are welcomed;

Baskerville 5.2 will have more on SGML, Acrobat etc

2 Colophon

Baskerville is set in Monotype Baskerville and Gill Sans, with Computer Modern Typewriter for literal text, and printed on a Hewlett Packard LaserJet 4. Production and distribution is undertaken in Cambridge by Robin Fairbairns and Jonathan Fine.

II Letter to the Editor

In her article on 'Table design' in *Baskerville* 4.4, Siep Kroonenberg says that '`\raggedright` simply doesn't work in a tabular environment' and suggests some difficult ways to get round this. I disagree: `\raggedright` works perfectly well in a tabular environment so long as the current entry ends with `&`. If your ragged right entry is not in the final column then you can put `&` after it even if you intend to have no more entries in that row. If you want any raggedright entries in the final column, the easiest way round the problem is to add a dummy column at the end of the `tabular` specification: for example, instead of saying `\begin{tabular}{... p{2in}}` you could put `\begin{tabular}{... p{2in}c}` or even `\begin{tabular}{... p{2in}@{}c@{}}` if you are fussy about unwanted extra space. When you are entering information in the tabular environment, you can forget about the extra dummy column except when the old final column has a ragged right entry, in which case that entry must finish with `&\\` instead of just `\\`.

R. A. Bailey

III A foreigner's impressions of UKTUG

Ulrich Jahnze
ulrich_jahnz@pe.maus.de

I have been interested in LaTeX for the past several years and I usually just *use* it and do not try to write fancy styles or macros. Nevertheless I was very pleased to get an invitation to participate in the 1994 UKTUG AGM at the University of Warwick. Though the meeting was not very well attended, it was still a worthwhile day for me. During the first part of this meeting the members of the group tried to find solutions for several administrative problems, and though most of them have been working together for the last decade, nobody tried to push things through and instead constructive criticism led to solutions everybody could agree with. The whole atmosphere was very friendly. During the afternoon several talks were given on various topics and, again, though many controversial points occurred, the people tried to work *together*. But is all this the main reason, why for me this trip was worthwhile? No! The most important thing for me is, to *see* those people. Almost everytime I buy a book, it has a short biography and a picture of the author in the beginning. If you compare how much time you spend reading one book to the time you work with all those nice programs and styles, I always ask myself, what are these people like as human beings.

Drawing a conclusion for me, the main reason to join in local TUGs and to attend meetings is not in the first instance to hear new things but to see those people, who are concerned about bringing LaTeX and TeX further. For those who cannot attend any meetings it would be a good idea to put at least a little picture of the author into the main articles in UKTUG's brilliant *Baskerville*, which on its own is a product that makes joining the UKTUG worthwhile. Since I had the opportunity to attend this meeting I can only suggest to everybody else to do the same and see "them" live. Before I finish I just want to thank especially Malcolm for his kindness and all his efforts!

IV LaTeX and tables

R. Allan Reese
Computer Centre
Hull University
r.a.reese@ucc.hull.ac.uk

A Tables

Kroonenberg's article on tables ([Kroonenberg 1994]) made some useful suggestions on LaTeX coding but the examples left much to be desired as patterns to follow. I fully agree that the LaTeX User Guide ([Lamport 1986]), [Kopka & Daly 1993] and virtually every book on word processing place far too much stress on rules as boxes. [Chapman 1986] gives better guidance on the presentation of tables; I particularly enjoy commending this book, as it challenges one of the Great Lies of Life. [1] Ehrenberg's short article ([Ehrenberg 1981]) should be studied by anyone putting figures into a table. [Reynolds 1983] is a more general book and discusses presentations of all types.

Chapman gives clear, straightforward guidance with many examples of good and bad practice. Here, for example, is a point almost always overlooked by people who believe 'a picture is worth a thousand words':

> "Since neither charts nor tables ever 'speak for themselves', in order to communicate a message effectively either must be accompanied by a verbal summary."

Tabular material tends to be complex. For a recent workshop on table construction in LaTeX, I selected what seemed to me the simplest tables from journals to hand. The criteria for simplicity were small size and not appearing to need a wide variety of LaTeX commands. In serious use, you would expect to use more commands than when teaching, so this restriction would be relaxed. More complex tables were often 'tables within tables', for which LaTeX's analytical approach of defining logical units is well suited. However, another of Chapman's truths that is easily swept aside by an author's enthusiasm is:

> "Tables should be small. It is better to include three or four compact tables, each illustrating one or two points succinctly, rather than construct a single large table which is then referred to in text covering a large number of paragraphs or pages."

Her next sentence will strike a chord with any LaTeX user:

> "Small tables are easier to position close to their verbal summary, and are easier to include in the main report."

Several problems arose in recreating the examples using LaTeX. Some were due to the desire to reproduce the table exactly as seen, rather than re-present the information in a natural way. On the other hand, the exercise did bring out the flexibility of LaTeX's standard tabulation tools.

The short workshop was successful, in that most of my (Computer Centre) colleagues were able to add further lines into a part-built table despite having no previous experience of LaTeX. They followed the layout by copying the commands that they saw; this contrasts greatly with WYSIWYG word processors where you may see a table but have no idea of how it was constructed.

LaTeX is an excellent program for formatting tables. It is not, however, realistic to expect tables to be laid out optimally first time and automatically. This is one area where the author must be prepared to make judgements and manual adjustments.

B Rules of thumb

Rules for constructing tables can be divided into those dealing with the content and those dealing with the layout. The typesetter will generally have little say in the content or ordering of the information, but an editor might (should) make suggestions. The evidence of most journals suggests their editors are not as critical of tables or graphics as of text.

Chapman distinguishes between demonstration tables and reference tables. The reader uses the former to perceive

[1] I 'm from the Government and I'm here to help you.

a pattern and the latter to look up a value. Tables that try to do both are rarely successful. Ehrenberg's examples would come under the demonstration category, for which he suggests:

give marginal averages to provide a visual focus

1. order the rows or columns by the marginal averages or some relevant measure of size
2. put figures to be compared in columns rather than in rows (i.e., to aid mental arithmetic)
3. round all numbers to two effective digits, unless the exact value is for reference

See [Ehrenberg 1981] and 1978 for further explanation and discussion.

Layout can make a table easier to read—or can destroy its meaning.

Sweep away the black boxes and apply two principles. A table is a discontinuity in reading. The reader is not to scan it linearly, as if reading text. You must therefore guide their eyes:

4. use white space to separate objects
1. use lines (rules) to join or point connections.

As when setting text, beginners tend to add too much space, though Chapman complains that even professional compositors like to widen tables to fill the text width. Physically compact tables are easier to scan. [Tufte 1983] makes this point about graphics; make them smaller so the reader can see the pattern not the dots. This applies also to tables. White space is used to set off the table from the body text, usually by centering. Spaces within the table are used to indent hierarchical headings and to break the table into sections. In another article ([Reese forthcoming]) I discuss LaTeX constructs for interposing space after a set number of lines or when the initial letter changes.

Horizontal rules are standard at top and bottom to further demarcate the table. There is usually a thick rule under the table heading, a thin rule under the banner (column) headings and a thin rule between the table body and any explanatory notes. Incidentally, one of Chapman's warning examples shows how disastrous it is to rely on the reader reading the footnotes to understand the table. They must be strictly to expand or qualify a detail.

Kroonenberg mentions the problem of text coming too close to rules. This can be adjusted with struts or non-aligned `\vspace`; I like to insert a small extra space above the first row and after the last, to unify the body of the table and distinguish it from the headings. Vertical rules are used sparingly, if at all. It is better to put a little space between stubs (row labels) and the 'data' columns, and between the columns and the marginal 'averages', than to add rules crossing the direction of scanning. You can help the reader scan across gaps by adding leaders, or by centering rather than justifying short items.

Use a smaller font inside tables. Kroonenberg implies this by discussing sans serif fonts "to set off the table." It is another way to make the table more compact and visually distinguished.

Telephone directories are large reference tables which demonstrate these principles. They are set in the smallest readable font size, have leaders to bridge the gap between name and number, and are usually multi-column with white space separators.

C Implementation

The layout principles are easily implemented in standard LaTeX. The table itself uses the `tabular` environment. The number of columns will often be greater than at first apparent, with many multicolumn items. For example, this is a natural way to handle hierarchical stubs, where the primary labels span two columns and the secondary labels start in column two (indented). LaTeX will calculate column widths but it is often desirable to force several columns to have the same width. Headings will often require either `\multicolumn` or `\noalign` to position them aesthetically.

LaTeX 2.09 had only `\hline` and `\cline`. The `hhline` package is worth fetching from CTAN. Line spacing for the whole table can be adjusted with `\arraystretch` and made different from `\baselinestretch`. Bear in mind that the table can be compressed as well as extended, and `\arraystretch`= 0.9 may make the pattern more obvious and the table *more* readable.

The table is then embedded in an environment to set it off from the text. You can choose from `quote`, `center`, display maths or `table`. Set the font size smaller within that environment, sans serif for the table and body text style for the captions. (The examples in this article follow their originals as closely as possible and don't do this. I think they would be improved if they did.) The table environment makes the object into a float, hence not to be broken between pages and with a cross-referencing label. The `longtable` package caters for tables that are too large for a single page.

Some mechanism should be used to create a left and right indent, the most obvious being to define the table width explicitly and center it. `minipage` puts the footnotes to the table at the bottom of the table, using different marks from those in the body text. Putting the `minipage` round the `table` rather than just the `tabular` also makes the caption narrower. I prefer the table reference to stand out, and often use the `hangcapt` package.

When there are several tables or a house style, it is better to define a new environment to ensure consistency in their presentation. I also commonly define new length constants for use in tables rather than copy the values.

D A LaTeX gap

One common format for tables has rows of short data values but a final column containing texts. The description parameter may therefore tell LaTeX to calculate the widths of data columns from the values, but the final column will be a `parbox` and should sensibly use the remainder of the `linewidth`. This is one need that LaTeX (2.09) blatantly fails to meet. You have to set the width of a `parbox` or `p`(aragraph) column. Kroonenberg implies this problem when discussing `\raggedright`; as she points out, to choose unjustified text you have to enclose each text in a `parbox`. This is less work if the width is set as a name (to be calculated) and not the numeric length. My pragmatic solution is to set the value for the final column initially to the `\linewidth`, run LaTeX, note (from the log) the overfill, and subtract that from `\linewidth`.

E Kroonenberg's 'after' table

The 'after' table of Economic Forecasts is still poor. This is partly due to the content. It lacks a text explanation (possibly the primary source had one), but I *guess* that the intention is to compare forecasts from two sources. Numbers going *down* the columns are not related (in this sense), so flipping (transposing) the table would make logical sense. This might be the best solution on a wider page, as having the table span two columns is yet another way of distinguishing it from the body. If we insist on keeping it within one (text) column, there is room for another column of values as a 'margin'.

Without a rubric, I cannot decide if the table is trying to show disagreement between the two sets of forecasts, or similarity. Dependent on the message, the best margin might be the difference between each pair of forecasts, a $+/-$ sign for higher or lower, the ratio, or the average. The forecasts themselves should all be rounded to two significant digits.

We seem to have lost between the 'before' and 'after' the detail that the figures are percentages; and *was* government income really a 'percentage change of a percentage'? The labels that include ($\times$ 1000 persons) are misleading; I take that notation to mean that the numbers shown *have been* multiplied by 1000 (like TeX magnifications). The correct notation would be (000s of persons).

Making the rules extend to the linewidth was presumably not thought out. They unbalance the design, and all the white space implies an omission. The spaces between the columns are too wide. If "mutations w.r.t. 1991" and "absolute quantities" are to be used as primary divisions, then the secondary stubs ("real consumption" etc) should be indented. If you think too deeply about the title, are these forecast changes or changes to (earlier) forecasts? Follow Chapman's advice and split the table into two, with headings of the form "Forecast percentage changes 1991/92 in National Economy" and "Forecast quantities…". I leave this redrafting as an exercise for the reader.

A parenthetic remark for those preparing tables of accounts comes from [Townsend 1970]. He suggests that "statements comparing budgets to actual should be written not in the usual terms of higher (lower) but in plain English of better (or worse) than predicted by the budget. This eliminates the mental gear changes between income items (where parentheses are bad) and expense items (where parentheses are good)." Typography helping the reader.

F New examples

The following examples are shown as output, in the expectation that the interested reader will obtain the 'input' from the the author or the editor. If there is sufficient interest, the LaTeX source for the tables can be placed on the CTAN archives.

3 Example 1—Cohabitation

This is taken from *Key Data 88* ([CSO 1988]) which had itself extracted it from *Social Trends*. *Key Data* is a sample of UK government statistics published annually as an educational resource and a guide to the more extensive sources. The table as printed had inconsistencies in the use of italic and upright fonts, and in its indentation.

Table 1 could be further improved by a distinct split and putting the two halves the other way round: "Numbers of

Table 1. C ohabitation

2.12 Percentage of women aged 18–49 cohabiting: by age

	Great Britain			Percentage and numbers		
		1979	1981	1983	1984	1985
	Age group *(percentages)*					
	18–24	4.5	5.6	5.2	7.3	9.1
2.	*25–49*	2.2	2.6	3.2	3.3	3.9
	All aged 18–49	2.7	3.3	3.6	4.2	5.0
	Women in sample *(numbers)*					
	(=100% above)					
	18–24	1,353	1,517	1,191	1,174	1,182
	25–49	4,651	5,007	4,094	4,070	4,182
	All aged 18–49	6,004	6,524	5,285	5,244	5,364

women surveyed in each year, in two age groups" and "Of the samples, percentage cohabiting". Note that the first half is essentially for reference and the second half for demonstration.

4 *Example 2—Vulture meat*

Table 2 is taken from Ibis ([Thibault et al. 1993]), journal of the British Ornithologists' Union. It is a simple table, but appears in a two-column layout and has footnotes. The original used numbers as the footnote markers. I found this disconcerting, as G^2 looks like a numeric power; so I'll use the LaTeX default in a minipage, which also sets the table to the width of an Ibis column.

This table is adequate for reference, and is discussed in an adjacent text. "Comparison of food availability among territories showed significant differences. …" The rows are not in an obvious order, and if the primary aim is to compare the number of, say, sheep in each territory, it might have been better flipped. A marginal total of the number of prey animals, or their biomass, in each territory would have been helpful. It is debatable whether the figures should be rounded, since they are taken from previous studies which are themselves recorded or reported with different precisions.

5 *Example 3—YOPs*

The problem with this table from [Chapman 1986, p11 and, slightly changed, p35] is that one column has an entry spreading over several lines (the brace linking them). The hint for doing this is in [Kopka & Daly 1993, p79], but they show only one optional argument to `\raisebox`. Using both optional arguments (setting height and depth to zero) and putting the 'brace array' on the line opposite its middle, is the easiest way to centralize it vertically. A convenient feature is that you can reset the `\arraystretch` and the brace is still the right size to cover three rows. This example is also set in a minipage but the footnote markers have been reset as numeric. Follow DEK's advice: *don't use footnotes in text.*

Table 3 Entrants to Youth Opportunities Programme in Wales: by type of scheme

Wales 1978 to 1980		Percentages
	1978/79	1979/80
WEEP [1]	89	73
Short Training Course	10	10
Community Service		9
Project based work experience		7
Training Workshops	1	1
Induction and other [2]		1
Total (100%)	15,000	22,000

[1] W ork Experience on Employers' Premises
[2] E mployment induction courses and other remedial and preparatory courses

The table has a large gap between columns 1 and 2, making it difficult to see at a glance whether the brace includes 'Community Service'. The original rubric ([MSC 1979/80]) was:

> "One of the major aims in 1979/80 was to increase the range of provision available to meet the varying needs of unemployed young people. In the early days of the Programme, there was heavy reliance on the Work Experience on Employers' Premises (WEEP) element, but the table reflects the increasing provision that has now been made in the other elements of YOP."

As presented, the first impression is that the number of schemes apparently grew, but each scheme attracted only a small percentage. '1' as a rounded value is very uninformative, especially as the compared figures are about 10 and 80%. There is also a wide gap between the caption '(100%)' and the figures it refers to. A quick sum shows that the numbers entering WEEP went *up* by 20% between the two periods. It's a classic political table; you can fiddle it to either praise or condemn.

References

[Chapman 1986] Chapman M. & B. Mahon (1986)*Plain Figures*. HMSO

[Ehrenberg 1981] Ehrenberg A.S.C. (1981) *The Problem of Numeracy*. The American Statistician, Vol 35, No 2

[Ehrenberg 1978] Ehrenberg A.S.C. (1978) *Data Reduction: Analysing and Interpreting Statistical Data*. John Wiley

[CSO 1988] Central Statistical Office (1988) *Key Data 88*. HMSO London

[Kopka & Daly 1993] Kopka H. & P.W. Daly (1993) *A Guide to LaTeX*. Addison-Wesley

[Kroonenberg 1994] Kroonenberg S. (1994) *Table Design*. Baskerville Vol. 4 No. 4 (reprint of article from NTG journal)

[Lamport 1986] Lamport L. (1986,1994) *LaTeX: User's Guide*. Addison-Wesley

[MSC 1979/80] Manpower Services Commission *Annual Report 1979/80*, para 8.34 and Table 35. HMSO

[Reese forthcoming] Reese R.A. *Dividing a Table Alphabetically*, submitted to TTN

[Reynolds 1983] Reynolds, L. (1983) *Presentation of Data in Science*. Nijhoff, The Hague

[Thibault et al. 1993] Thibault J.-C., Vigne J.-D. & J. Torre (1993) *The Diet of young Lammergeiers in Corsica*. Ibis Vol. 135 No. 1

[Townsend 1970] Townsend R. (1970) *Up the Organization*. Michael Joseph, London (Coronet edition 1971)

[Tufte 1983] Tufte, E. (1983) *The Visual Display of Quantitative Information*. Graphic Press, Conn

Table 2. *Numbers of ungulates in Lammergeier territories (except for wild boar, differences among territories are statistically significant, $\chi^2_{16} = 16.825, P < 0.001$).*

	Territory				
Species	B [a]	G [b]	R [c]	T [c]	V [c]
Sheep	2000	1400	5240	3165	3480
Mouflon	< 200	–	–	< 400	< 5
Goat	570	4142	1880	1510	820
Cattle	2300	6402	7204	6774	965
Pig	3400	7188	1900	400	522
Boar	+ [d]	+	+	+	+

[a] D ubray & Roux(1990)
[b] A nonymous (1989).
[c] D irection ...
[d] $+ =$ present. $- =$ absent.

V HyperTEX: a working standard

Arthur P. Smith
Dept. of Chemistry, BG-10,
University of Washington,
Seattle, WA 98195
asmith@mammoth.chem.washington.edu

A Introduction

Note: this paper was prepared for the American Phsyical Society electronic publishing conference, Los Alamos N.M. October 14–15 1994.

The past year has seen a revolution in the processes of Internet-based information navigation and retrieval with the advent of easy-to-use graphical browsers (in particular Mosaic) based on the World-Wide-Web (WWW). The revolution is a result of two components — first the browsers allow a near-uniform (point-and-click or other method) access to documents in almost any format and from almost any Internet-based source, accessed as regular files or via ftp, gopher, http or one of many other possible methods; along with this the Universal Resource Locator (URL) mechanism provides a surprisingly easy and uniform way to specify the location of any document on the net. Second, for certain classes of documents (html files, or gopher text files) embedded URL's or other addresses are understood to refer to other, external, documents which can be followed according to the interests of the person viewing the document, producing an interconnected web of documents.

The goal of the HyperTEX collaboration is to extend this second privileged class of documents to include documents based on TEX, the word-processing language of choice for mathematical and scientific writing, thus fully incorporating TEX documents into the burgeoning **web** of information on the internet.

B Why HyperTEX?

There already exists one approach for incorporating TEX documents more fully into the **web** — conversion to HTML, as in the program `latex2html` by Nikos Drakos. This can work very well, and is already used in some of the electronic publications in mathematics, but there are also several serious problems with this, aside from the technical issues associated with the complexity of the conversion process. HTML by design allows very little author control of the visual form of a document. This is touted as an advantage because it preserves only the *essential* elements of a document and not the artificialities of a page — in fact HTML documents do not have pages at all, although some of the sense of a *page* is implied by separation of a single document into many files. Aside from loss of author control, there is a practical problem of a lack of mathematical tools in the current implementations of HTML — tables and equations are either difficult to implement or impossible. `latex2html` gets around this by conversion of such things to bitmapped images, but this is an inefficient and expensive process — and goes in just the opposite direction of HTML's theme of extracting the *essence* of a document, making the document essentially unreadable without a good network connection and a computer with a high quality display.

These problems with HTML are compounded if scientific authors attempt to write documents directly in HTML rather than using TEX first — the lack of authoring tools, the absence of macro capabilities, and the ill-defined nature of the language make this an unpleasant task; just dealing with ordinary text is easy, but getting Greek letters, mathematical symbols, equations and tables into your document is not. The one nice feature of HTML is the ease with which figures can be incorporated into a document. But at least PostScript figures can be incorporated into a TEX document with equal ease using modern *dvi* interpreters, and the HyperTEX standard presented here allows arbitrary images and other external documents to be referred to and brought to the screen with a single mouse click.

The point of all this is that hypertext capabilities, and the use of URL's to locate new documents — the main feature of HTML that makes it such a useful network information navigation tool — can be much more easily incorporated into TEX than the mathematical capabilities of TEX and the years of experience embedded in various TEX macro packages can be incorporated into HTML. Whether TeX in general provides a better model for the viewing of on-line information remains to be seen.

C How does it work?

The underlying element of our implementation of HyperTeX is the use of a TeX macro that bypasses the TeX interpretation process and sends a message directly to the *dvi* interpreter that processes TeX output. This is the `\special` command, previously used to define procedures for drawing or including figures in TeX documents. When the characters `\special{`*`string`*`}` appear in the TeX document, the *string* is passed directly without interpretation to the output *dvi* file (preceded by a marker to identify this as a *special* message to the *dvi* interpreter). The *dvi* previewers or processers then interpret this string according to its first few characters. The original HyperTeX specification (due to Paul Ginsparg, Tanmoy Bhattacharya, and me) uses the initial characters *html:* to denote HyperTeX elements in an HTML-like style. David Oliver (`oliver@gang.umass.edu`) has introduced a slightly different specification that uses the initial characters *hyp* to denote his own style of HyperTeX. I will discuss only the original specification in this paper, since as far as they are currently implemented both specifications are essentially equivalent. Note that *dvi* interpreters that do not understand the *html:* or *hyp* special commands will ignore them, or at worst print out warning messages. Therefore *dvi* files processed to include HyperTeX commands are fully compatible with old *dvi* interpreters.

After the initial *html:* string, the specification is identical to a restricted form of HTML. The five arguments we have added to the `\special` command are:

href: html:
name: html:
end: html:
image: html:<img src = "href_string">
base_name: html:<base href = "href_string">

The *href*, *name* and *end* commands are used to do the basic hypertext operations of establishing links between sections of documents. The *image* command is intended (as with current html viewers) to eventually place an image of arbitrary graphical format on the page in the current location. Currently for Xhdvi, *image* brings up an external viewer with the image, if such a viewer is available. The *base_name* command should be used to communicate to the *dvi* viewer the full (URL) location of the current document so that files specified by relative URL's may be retrieved correctly.

The href and name commands must be paired with an end command later in the TeX file — the TeX commands between the two ends of a pair form an *anchor* in the document. In the case of an `\href` command, the *anchor* is to be highlighted in the *dvi* viewer, and when clicked on will cause the scene to shift to the destination specified by *href_string*. The *anchor* associated with a name command represents a possible location to which other hypertext links may refer, either as local references (of the form `href="#name_string"` with the *name_string* identical to the one in the name command) or as part of a URL (of the form *URL#name_string*). Here *href_string* is a valid URL or local identifier, while name_string could be any string at all: the only caveat is that '`"`' characters should be escaped with a backslash (\), and if it looks like a URL name it may cause problems. There may also be problems if LaTeX tries to interpret the *href_string* or *name_string* — in that case preceding the command with `\protect` should usually work. Any defined *name_string* can be referred to in any href referring to the document, in the form `href="URL#name_string"`. Note that anchors may be nested. The only restriction in current implementations is that anchors are truncated at page boundaries.

Because this html-based naming scheme is somewhat unwieldy, although very general, Tanmoy Bhattacharya (`tanmoy@qcd.lanl.gov`) has written several collections of TeX macros to simplify things. The basic package is *hyperbasics.tex* [2] which defines the following simple low level hypertex macros:

- `\href{url}{text}`: text becomes an href anchor referring to *url*.
- `\hname{myname}{text}`: text becomes a name anchor with name *myname*.

plus others that are used to automatically convert LaTeX or other style markup into corresponding names and references.

D How do I use it?

[2] `http://nqcd.lanl.gov/people/tanmoy/hypertex/hyperbasics.tex`

6 *As a reader*

There are currently two *dvi* interpreters that understand the HyperTEX `\specials`: Xhdvi for X windows, and HyperTeXView.app for NextStep. We are proceeding with work on a *dvi*-pdf converter that understands HyperTEX, and we are encouraging work on *dvi* previewers or TEX authoring tools for Macintosh and PC that incorporate HyperTEX elements.

For a TEX document that has already been processed to a *dvi* file with HyperTEX elements, viewing the internal hypertext is almost trivial — you just fire up the *dvi* previewer and navigate by button clicks as with Mosaic or other WWW browsers. To have Xhdvi, for example, brought up automatically from Mosaic when a *dvi* document is referenced, you need to have a `.mailcap` file in your home directory, and create or modify the line:

```
application/x-dvi; xhdvi %s
```

Your machine must already have the TEX essentials on board of course — in particular the pk font files, and the location of those font files needs to be communicated to the previewer. If xdvi is already working for you, Xhdvi should work too. Details for getting Xhdvi working on your machine are provided below.

For jumping to external documents from within the hypertexted *dvi* file, a couple of additional elements are needed, also desribed below for the case of Xhdvi.

7 *As an author*

Here is where the power of TEX's macro capabilities appears. A working internal hypertext document can be made from a LATEX document with a one-line addition to the file, using Tanmoy Bhattacharya's hypertex macros. These macros convert the standard LATEX markup into hypertext links between the different sections of the document, so that references to equations, tables, footnotes, and section headings are in place, and bibliographic references and figures refer back at least to the bibliography entry or figure caption. These in turn may be set to refer to corresponding external documents but this process is not automatic — currently the author will have to add these references by hand, although automatic procedures can be envisioned. With an Internet connection, Xhdvi can be used to preview the document and check that the references actually work, before the document is submitted to the archives.

The macros developed thus far use standard naming conventions for the underlying structures in LATEX and other standard macro packages, so that appending #equation.2.3, #page.7, #figure.4, #table.2, etc. to the URL for any TEX file processed with these packages will go to the right place, allowing easy hypertext reference to the internal structure of other documents.

In order to get started, however, you need to place these macro files in one of the standard areas that your TEX looks for input files (you can modify your TEXINPUTS environment variable to get it to look in your own directories). The needed macro files are itemized in the HyperTEX introductory document at `http://xxx.lanl.gov/hypertex/index.html\#more` and can be obtained in one lump by anonymous ftp. [3]

8 *As an e-print manager*

Since we currently only have *dvi* previewers, an e-print server would have to serve the documents in pre-processed *.dvi* form. This means converting documents to HyperTEX if the author has not already done this, and possibly applying automated insertion of URL's corresponding to references in the bibliographic section. The manager could do this by hand but it might be rather time-consuming.

For ease of use, the best way to serve the documents is probably as a combined package of *dvi* and PostScript files that go together. This requires the e-print manager to create a new content-type associated with this package, and to supply an unpackaging program for the reader to place in their `.mailcap` file, which automatically calls up Xhdvi or another HyperTEX browser on the resultant main *dvi* file. The reason for doing this is that .ps files included by standard macros will not generally be understood as remote documents, at least at the current level of previewer capabilities. Another option in this unpackaging method is to supply the TEX file itself, pipe it through a simple converter to HyperTEX and through TEX itself, and then call one of the HyperTEX viewers. These approaches are already in use at some locations (e.g., CERN).

When the pdf converter is available, the entire document should come as a single pdf file, unless the document refers to non-PostScript images or other inclusions in which case the packaging approach (or use of absolute URL's) remains necessary.

[3] `ftp://snorri.chem.washington.edu/hypertex/hypermacros.Z`

E How do I get it?

Currently the following are available:

A HyperTEX viewer [4] based on xdvi-18, modified by Arthur Smith. Precompiled versions for various UNIX architectures are available in the same directory.

1. HyperTeXview.app, [5] courtesy of Dmitri Linde (also the author of InstantTeX.app) for NextStep, precompiled for Motorola and Intel-based NeXT machines. [6]

The macro and style files listed above by Tanmoy Bhattacharya, available at `ftp://nqcd.lanl.gov/people/tanmoy/hypertex`

F Details on Xhdvi

Xhdvi retains all the features of the latest version of xdvi (version 18) and adopts in addition many of the hypertext features of Mosaic, the most popular WWW browser. Hypertext links are underlined or altered in colour (the underlining can be turned off) and a left-mouse click on a link causes the view to shift to the destination point for the link, as long as the destination is another *dvi* file. If the link is not to a *dvi* file, an external viewer is employed, following the mime and mailcap definitions or using standard defaults if those are not locally defined. A middle mouse click on a link brings up a new viewer whether or not the destination is a *dvi* file — this is intended to be useful to refer back to equations or to bring up footnotes, since the new *dvi* window is small. There are also a large number of keyboard accelerators, all described in detail in the man page.

In general, see the installation notes provided with Xhdvi.

In outline what is needed is:

2. The compiled Xhdvi program — precompiled binaries are available for Sun, NeXT, SGI, HP, IBM RS6000, or you can get the source and compile it yourself. Let me know of any compilation troubles — it's written in C.
1. The TEX fonts, at least in pk format. If xdvi, *dvi*ps or some other *dvi* interpreter are working on your machine then they must be around somewhere.
2. Set up the connections between the Web browser and Xhdvi. If you use mosaic for example,
3. `setenv WWWBROWSER /usr/local/bin/mosaic`
   will let Xhdvi know what to send HTML files to. To let mosaic know to bring up Xhdvi for any *dvi* files, you need to amend in your `.mailcap` file as described above.

   The application defaults file for Xhdvi should be installed in the standard application defaults directory on your machine, or you can take lines from it and modify them for your own taste and put them in your `~/.Xdefaults` file. For example I use the following resource specifications to get a particular size and position of the window with white on black lettering and with the hyperlinks in cyan, and to remove the buttons:

   4. xhdvi*geometry: 800x600-0-0
      xhdvi*foreground: white
      xhdvi*background: black
      xhdvi*highlight: cyan
      xhdvi*expert: true

   You need to have the ghostscript program on your machine and in your default execution path in order to view postscript from Xhdvi. Similarly, other viewers defined in the `.mailcap` file should be available on the machine.
5. You need to install the man page xhdvi.man in `/usr/local/man/man1` and add `/usr/local/man` to your MANPATH environment variable in order for *help* to work from Xhdvi.

G Some examples

This document is available in raw HyperTEX format and in converted *dvi* format via anonymous ftp at the address `ftp://snorri.chem.washington.edu/hypertex`. The HyperTEX version of this paper uses the two-column

[4] `ftp://snorri.chem.washington.edu/hypertex/xhdvi_0.6.tar.Z`
[5] `dmitri@physics.stanford.edu`
[6] S ee `http://xxx.lanl.gov/hypertex/index.html#dvi` for availability.

APS journal style of revtex. The table of contents at the beginning is generated automatically with the LaTeX `\tableofcontents` command.

See also the examples provided by Paul Ginsparg in the HyperTeX introductory document at `http://xxx.lanl.gov/hypertex/index.html`. Some of these are files randomly selected from the HEP archive, including LaTeX, RevTeX, and other formats.

H What still needs to be done?

Unfortunately, at this point reference to networked files (via URL's) suffers from a couple of problems. Xhdvi does not yet include any of the network transport code that ordinary WWW browsers use, and the intention was to avoid having to add this layer of complexity by communications back and forth with a WWW browser. However, such communication is as yet not standardized, and suffers from its own problems. So currently, when Xhdvi comes across a URL reference, it forwards it directly to the WWW browser (defined by environment or Xresource variables) so that a reference to an external *dvi* file would bring up a new instance of the WWW browser which would in turn bring up a new Xhdvi viewer. This is a rather inelegant solution, but it is perhaps sufficient at the moment. A better solution will come along, and it may simply be inclusion of network transport code in the Xhdvi viewer itself, to make it a competing WWW browser…

The other problem is that if brought up by a WWW browser, Xhdvi is not provided with the absolute URL information used in obtaining the *dvi* file it is working on, and so cannot pass this information on to further instances. Therefore, relative URL's in a HyperTeX document (unless they can be guaranteed to be to local files that would have been transported along with the *dvi* file) will not work.

Both of the above are problems intrinsic to current WWW browsers, and we are working on promulgating solutions to these.

I How do I stay in contact?

The Hypertex discussion group is a mailing list based at `snorri.chem.washington.edu` which I maintain. Send me e-mail if you want to join the list, or send queries directly to the mailing list: `hypertex@snorri.chem.washington.edu`.

VI A Moral Tale

Malcolm Clark
6. m.clark@warwick.ac.uk

A Keeping to the straight (and narrow)

Both TEX and LATEX are remarkably smart. From time to time they are either too smart, or not quite smart enough. One area that I have always been dubious about is how subscripts are handled. In the simple case, I find that the unevenness of the baselines in a case like:

$$x_1 y_1^1$$

is both odd and unsettling. The reasons for this are mainly in the font dimensions. This is a subject area which is mostly relegated to a dusty corner, although I do address it a little in my *Plain TEX Primer*. At this point, I have to admit that I strongly believe that the use of LATEX 2ε is to be advocated and preferred, at least until LATEX3 and LATEX4 come along. Using LATEX2.09 is only excusable if you have old documents you still need to process. Anything new should be done in LATEX 2ε. The justification for the use of TEX is more difficult, unless you have particularly well developed prejudices. However, in the explanations that follow I shall start through plain TEX rather than LATEX 2ε. To some extent, TEX is the 'accessible basic LATEX system, which doesn't require wizardry to tailor to one's own preferences' which Siep Kroonenberg said is needed (*Baskerville* 4.4). At this point we need something slim and easily tweakable. I'm therefore using TEX as a sort of 'prototyping' system which allows me to demonstrate things which I will eventually transfer to LATEX 2ε. This is how I think TEX is best used.

B A plain explanation

There are a couple of interactions going on with sub- and superscripts. The first and easiest to explain is the behaviour of two of the font dimensions, 16 and 17, which control the distance between the subscripts and the baseline. If we constrain the two font dimensions to be equal, our apparent 'problem' disappears (see also Knuth, page 179, a double dangerous bend section). Therefore, if we set

```
\fontdimen16\cmsy=\fontdimen17\cmsy
```

we can obtain results where the baselines of the subscripts are aligned:

$$x_1 y_1^1$$

This is reasonably well understood, and the explanation, that we do not wish the superscripts and subscripts to interfere with one another (in a broader aesthetic sense) seems reasonably well grounded. I am not too clear how widely this particular attention to detail is practised in the publishing profession. Perhaps some readers have information on this. I would be particularly interested in publishing houses who do not use the 'Addison-Wesley' approach to typesetting mathematics.

The real stimulus for this note was the common requirement to include pieces of text within a mathematical expression. When the concepts are going to be used frequently, this is unlikely to be very important, since you would probably define a symbol to represent the concept. However, in more informal works it can make the equation much more accessible to the reader. To ease terminology, I shall refer to the relative sizes by the plain TEX terms: 'display style' as in (2), and 'script style' for first level subscripts, diminished in size, as in (3). In plain TEX, display style would be a 10 pt font, and script style would be 7 pt.

For example, it may be convenient to be able to say

$$\log(\text{amplitude}) = -2(2H+1)\log(\text{order}) \tag{1}$$

given by

```
\log(\hbox{amplitude})=
  -2(2H+1)\log(\hbox{order})
```

This is fine until we try to do the same for subscripted text. For example, the 'direct' equivalent of

```
u_{\hbox{ex}}
  =u_{\hbox{maximum Airy wave}}(1-F)
```

gives

$$u_{\hbox{ex}} = u_{\hbox{maximum Airy wave}}(1-F) \tag{2}$$

which is 'subscripted' material but in display style. If we really wanted

$$u_{\rm ex} = u_{\rm maximum\ Airy\ wave}(1-F) \tag{3}$$

how would we obtain it? The answer is fairly straightforward: force the size of the text to `\scriptstyle` size. Unfortunately we cannot do this by employing `\scriptstyle` directly, since it is an instruction which may be used only within maths. Enclosing it in an `\hbox` insulates it from maths. The nearest we can do is to recall that a first level superscript is 7pt, and use `\sevenrm`:

```
u_{\hbox{\sevenrm ex}}=
u_{\hbox{\sevenrm maximum Airy wave}}
(1-F)
```

This is most unsatisfactory. You should never have to declare explicit font changes like this.

I would also agree that all this `\hbox` is a bit of a mouthful, but you are buying convenience for your readers at the expense of some minor personal inconvenience. But I said earlier, if you were going to litter your text with this, you really ought to define a few symbols instead.

There is still a 'small' problem with (2). Let's wind back a little and assume that we are not all that interested in making sure the subscripted text is in script style, and that we are content with display style size for the subscripted textual material. This is perhaps not wholly unreasonable, according to the context. Consider the following equation fragment:

$$a_{\hbox{x}} + a_{\hbox{y}} + a_{\hbox{b}} \tag{4}$$

obtained from

```
a_{\hbox{x}} + a_{\hbox{y}}
  + a_{\hbox{b}}
```

The problem is the lack of alignment in the baselines of the subscripted material, which was also present with the 'ex' of (2). The 'x' and 'y' are fine, but the 'b' has a lower baseline. As I said earlier, I find this a little unsettling. This time however, we have no superscripts interacting with the positioning.

We can solve the problem of aligning the baselines moderately readily by a subterfuge. It just 'happens' that the left and right parentheses in Computer Modern are text characters with the maximum vertical extent. No other text character has a greater ascender or descender. Had our text subscripted material been parenthetical, the baselines would have been aligned:

$$a_{\hbox{(x)}} + a_{\hbox{(y)}} + a_{\hbox{(b)}} \tag{5}$$

This is probably not what we wanted, but we could just have a `\vphantom{()}` with each subscript. It is ugly and crude, but it will do what we require.

$$a_{\hbox{x}\vphantom{()}} + a_{\hbox{y}\vphantom{()}} + a_{\hbox{b}\vphantom{()}} \tag{6}$$

for example:

```
a_{\hbox{x\vphantom{()}}}
  + a_{\hbox{y\vphantom{()}}}
  + a_{\hbox{b\vphantom{()}}}
```

In passing, those depending on their intuition and considering the use of a `\strut` are sadly wrong. Struts which are the size of the 'local' parenthesis would be a boon, but unfortunately they are not so defined.

At this point, the question we could ask is 'why does the displaystyle subscripted material behave differently to the scriptstyle material'. Naturally, all is revealed in the TEX book. Appendix G (Generating Boxes from Formulas) reveals all, but in true K(nu)thonic style it demands great attention to detail. The answer is clearly not just in the font dimensions, or placing them equal would have meant that they all shared the same baseline, and the presence or lack of superscripts would have been irrelevant. Fighting your way through the key section (number 18) we will come to the conclusion that it is the size (mostly the height) of the subscripted material which determines what happens. In the case of material set in displaystyle, the extra height of material with an ascender is sufficient to select a branch which lowers it further. The material without ascenders does not select the option, and therefore we can have irregular baselines. The scriptsize material on the other hand does not branch to this alternative, because it is that much smaller in the first place, and therefore the baselines remain aligned.

In order to force conditions so that the subscripted display style material is guaranteed to align, we can fiddle with the font dimensions. What we cannot do is fiddle with the conditions and rules which are hard wired into TEX. This, I think is one of the weaknesses of TEX. Its monolithic nature precludes plugging in different rule sets – either as improvements, or to accommodate new situations. If we change the two font dimensions to

```
\fontdimen16\tensy=1ex
\fontdimen17\tensy=1ex
```

we can ensure the alignment. Unfortunately, this is less than perfect when we also wish to have 'normal' scriptstyle susbcripts. They appear much too far below the level of the material they subscript. The value of 1 ex is actually a little more than you would need if you never had superscripts to contend with too (about 0.95ex would probably do – you can work it out exactly with the rules in section 18), but as soon as you bring in superscripts you then have to contend with an extra vertical movement designed to ensure that sub- and superscript are apart by at least four times the thickness of a rule.

C Back to the present

How do we translate all this into LATEX 2ε, since that is what you are going to want to use? It turns out that the option of altering the font dimensions is not really one which is open to us. Although the *Companion* does give an example of the use of `\DeclareFontShape` where it demonstrates the syntax of how `\fontdimen`s may be changed, it notes that the example would not work and that 'the best way to solve this problem is to define a virtual font'. I'm afraid that if the solution is to define a virtual font, then I'll change the problem. Another problem lies in the fact that you cannot change the characteristics of already loaded fonts. The fonts we use are almost inevitably preloaded (i.e. LATEX 2ε has already absorbed all their font metric information, including the font dimensions). The only way to do this is therefore to work with tftopl to convert the TEX font metric file to a property list, edit the font dimensions by hand, convert back to a TEX font metric file and then re-create the LATEX 2ε format. This is almost as bad as using virtual fonts. We do not end up with something we can switch on and off at will. You cannot change formats part way through typesetting.

The translations of equations 1 and 2 are done simply by changing the TEX `\hbox` instruction into either an `\mbox` or `\textrm`. In this instance they are equivalent. In the case of 3, we do not have a LATEX 2ε instruction `\scriptstyle`. How then do we obtain the 'correct' size? Things start to become a little more difficult. LATEX 2ε does support a number of font size changing instructions, notably `\small`, `\footnotesize`, `\scriptsize` and `\tiny`. In all three size options, 10pt, 11pt and12pt the relative size gradation is

$$\texttt{\textbackslash small} > \texttt{\textbackslash footnotesize} > \texttt{\textbackslash scriptsize} > \texttt{\textbackslash tiny}$$

That does not indicate which of these sizes corresponds to a TEX scriptstyle, although we could make a fair guess that `\scriptsize` is the likely candidate. There is also an instruction `\scriptstyle`, but this may only be used within math style and trying to combine it with `\textrm` is doomed to failure. You might have thought that `\mathrm` could help, but of course the spacing between letters would be the normal maths spacing between variables. It is obvious that LATEX 2ε has a way of doing this properly, or constructs like

$$x_{\log y} \tag{7}$$

would not work. If we delve into latex2e.ltx, where this and similar operators are defined, we discover (certainly to my surprise) that they do not use the LATEX 2ε font changing mechanisms, but instead use the definition

```
\def\log{\mathop{\rm log}\nolimits}
```

This is the very font changing mechanism which the *Companion* describes with the admonishment: 'we suggest that you refrain from using such commands in new documents'. To be fair, this isn't any sort of LATEX, but TEX, since the `\def` instruction is *not* a LATEX command. [7] To be even more fair, this deficiency has been recognised, and my

[7] I 've tried persuading FM that it should be removed at LATEX3. He smiled.

latex2e.ltx is the beta release. Apparently current versions refer instead to `\operator@font`: not normally accessible to humans. However, although the sizes will be correct, the inter word spacing disappears altogether.

The conclusion is therefore that we have to select one of the font size changing instructions. Before we do that, let's look to see which of these sizes would give us aligned subscripts. The earlier experiments with TEX indicated that the alignment was sensitive to the size. Here are the size changing mechanisms of `\small` (8), `\footnotesize` (9), `\scriptsize` (10), and `\tiny` (11):

$$a_{\mathrm{x}} + a_{\mathrm{y}} + a_{\mathrm{b}} \tag{8}$$

$$a_{\mathrm{x}} + a_{\mathrm{y}} + a_{\mathrm{b}} \tag{9}$$

$$a_{\mathrm{x}} + a_{\mathrm{y}} + a_{\mathrm{b}} \tag{10}$$

$$a_{\mathrm{x}} + a_{\mathrm{y}} + a_{\mathrm{b}} \tag{11}$$

Examination reveals that `\tiny`, and, as anticipated, `\scriptsize` are aligned. The problem stated however, was to have the subscripted material at 'normal' text size. Therefore the easy solution is to incorporate the `\vphantom`s, as in equation (6):

```
a_{\textrm{x\vphantom{()}}}
+ a_{\textrm{y\vphantom{()}}}
+ a_{\textrm{b\vphantom{()}}}
```

Of course to save typing you would use `\newcommand` to make `\vphantom{()}` shorter. Should you wish to reduce the size, then you could do something like

```
a_{\textrm{\scriptsize x}}
   + a_{\textrm{\scriptsize y}}
   + a_{\textrm{\scriptsize b}}
```

D Conclusions

It therefore appears to be cumbersome to come up with some neat fix, and the rather inelegant solution of phantom parentheses seems to be about the least inoffensive we can manage. Do remember that it only works in a guaranteed way with Computer Modern. There is no reason to assume that other maths fonts will share this characteristic. If you want to subscripted material to match in size to other susbcripts, use `\scriptsize`. In some ways the LATEX solutions are tidier and marginally less inelegant than the plain TEX solutions.

E Afterword

There is yet hope. After I wrote this, I was alerted to the amstext package which produces `\scriptsize` text when subscripted, without any of the trickery outlined here. This is outlined on page 227 of *The Companion*. I should really have known, since, as Lamport says on page 52 of *The LATEX User's Guide*, 'sooner or later you'll encounter one (formula) which can't be handled with the commands described so far. . . consider using the amstex package'. Amstext is a sub-package of amstex. RTFM.

If RAB's handy mathematical hints don't get there first, I'll look at amstex more closely in the future. . . >

VII Elements of SGML

Jonathan Fine
203 Coldhams Lane
Cambridge CB1 3HY
J.Fine@pmms.cam.ac.uk

This is the first of a series of articles on various aspects of SGML. It is intended to be a general introduction. Subsequent articles will discuss the SGML concepts of document elements and their content, attributes, entities, markup minimization and data notation. The author welcomes queries and requests. He is also available for professional consultation.

9 Introduction

Here is the opening paragraph of the SGML standard:

> This International Standard specifies a language for document representation referred to as the "Standard Generalized Markup Language" (SGML). SGML can be used for publishing in its broadest definition, ranging from single medium conventional publishing to multi-media data base publishing. SGML can also be used in office document processing when the benefits of human readability and interchange with publishing systems are required.
>
> from *Clause 0, Introduction, ISO 8879, October 1986*

Necessarily, this article will give an incomplete picture of SGML. Here are five professional activities involved in modern publishing. It is the *author* whose words are published. Most of the time we assume for simplicity that it is the author who keys the manuscript into a computer, creating what I shall call a *compuscript*, or *script* for short. This assumption is of course not true for Shakespeare and many other authors. The *designer* will establish the structure of the author's work—perhaps retrospectively—and establish standards for its printed representation. The *typesetter* or application programmer will cause software tools to produce from the author's script printed pages or whatever meeting the designer's requirements. These software tools will have been created by an *implementor* or systems programmer. The *publisher* will have an overall responsibility for, and financial interest in, the whole process.

This quintet—author, designer, typesetter, implementor, and publisher—are all involved in the production of a book before it goes to printer and binder. This production process is generally of little concern (except when it goes wrong) to the final sixth party, the *reader*. However, SGML can be used to offer the reader new electronically published products.

In this article I shall show you, in its entirety, an extremely simple SGML document. By and large I will take the author's point of view, in part in hope of alleviating any distrust there may be amongst the humanities towards technology. I hope that the more technically minded will bear with me during this apparently pedestrian exposition. They may wish to reflect on how SGML allows cooperation and division of responsibility within the production process.

A Field of Application

It is useful to know what SGML can be used for, and what lies outside its province.

> The Standard Generalized Markup Language can be used for documents that are processed by any text processing or word processing system. It is particularly applicable to:
>
> a) Documents that are interchanged among systems with differing text processing languages.
> b) Documents that are processed in more than one way, even when the procedures use the same text processing language.
>
> Documents that exist solely in final imaged form are not within the field of application of this International Standard.
>
> *Clause 2, Field of Application, ISO 8879*

This is the whole of Clause 2 of the Standard, which altogether has 15 clauses and 9 annexes. Unfortunately, they are all much harder and longer than this clause. This clause may be all that the publisher needs to know.

It is well worth noting that SGML is first of all a standard for *documents*, of particular use when documents are interchanged among systems (the author sends the compuscript to the publisher who sends it on to the typesetter) or processed in several ways (for paper or electronic publication, in a journal or a book, or extracts in a secondary journal, or even just a second edition). Just as the ASCII character codes provide a standard for the expression and thus interchange of sequences of characters, so SGML is to provide a standard for the expression of structured or marked-up documents. However (note to Clause 1) the SGML International Standard does not specify standard document types, or standard SGML applications, nor the implementation or architecture of either the application or the electronic storage representation of the documents. It is an abstract standard for documents, deliberately indifferent to the specifics of application and implementation.

B Hello world!

It is traditional to begin the explanation of a computer language with the code that will print the words:

```
Hello world!
```

say on the user's screen. An author might object to using a computer language to write the great English novel. But SGML is quite unlike other computer languages, and anyway, all word processors use their own computer language to represent your documents. Do you know what they are doing to your words? With SGML you do, or at least can if you wish to. You don't need to be a programmer to write an SGML document. SGML is not even a programming language—it is a document structure and markup language. Moreover, SGML does not understand the English language, and so will not correct your spelling or criticize your plot.

Instead, with SGML you make statements, which are called *declarations*. Most other computer languages are concerned with giving instructions to the computer. However, SGML does little more than record your declarations. It also checks that you are doing only what you allowed yourself to do. Though this may sound onerous, according to Hegel "one who will do something great must learn to limit oneself." It can be a healthy discipline.

Back to our example. We wish to express

```
Hello world!
```

in SGML. If this is too trivial for you, replace this text by Shakespeare's sonnet

```
Shall I compare thee to a summer's day?
```

or even the collection of his sonnets, complete with editorial and critical apparatus, and publishing information. It is to such that we wish to add markup, which is defined in the Standard as *text that is added to the data of a document in order to convey information about it.* We shall mark up our message so that it is an SGML document.

What sort of document is it? It is a message. So we mark it up as

```
<message>Hello world!</message>
```

where the added text is markup, sharing with the computer our knowledge, that the original text is a message. The *content* of the message *element* is our original text, which lies between the *start-tag* and the *end-tag*.

Although the author may be satisfied by this, the programmer or typesetter will not. This person would like to know, without reading the whole document (which might be quite long and perhaps still in progress), what elements (tags) are to be found in it. And the author might like to be told when a tag name (called for historic reasons the *generic identifier*) has been misspelt. Before we use a tag—or any other textual markup—we must first declare that it exists to be used.

Although it may appear to be overly fussy, it is arguably to everyone's benefit that all declarations required for a document should appear before any of its text. In any case, SGML insists that a document consists of a *prolog* (which contains all the markup declarations) followed by the *document instance*, which is the author's text, marked up in conformity to the declarations in the prolog. This is important. To repeat: an SGML document consists of the prolog followed by the conforming document instance. It is this requirement which allows documents to be interchanged among systems.

Before our marked up document instance

```
<message>Hello world!</message>
```

can be allowed, we must create a prolog which declares the markup construction(s) that can be used. The text to do this is here needlessly verbose, except that it will later enable the powerful CONCUR and LINK capabilities of SGML.

Here is our message, marked up as an SGML document.

```
<!DOCTYPE message [
  <!ELEMENT message ANY >
]>

<message>Hello world!</message>
```

There are four occurrences of the character string `message`. The first tells us that the document instance to follow is to consist of a `message` element. The second tells us that there are no restrictions on what may appear in a `message` element. `ANY` words or characters, `ANY` elements, and `ANY` other markup constructions, repeated as often as one likes, and in `ANY` order. The third and fourth delimit the content of the `message` element, which the first occurrence of `message` had promised.

There are 55 *reserved names* such as `DOCTYPE`, `ELEMENT` and `ANY` in SGML, which have special rôles in the prolog, and also some special character strings. Neither the author nor the publisher needs to know what they are or what they do.

For more complicated documents, particularly those that are to conform to a house style, the prolog—which declares the elements and structure of the document—cannot be left to chance. In particular, the author should be relieved of responsibilty for the prolog, and not given the impression that it is something that he or she could change, if they so wished. (This last remark is directed to the publisher.) The designer (or someone else) will create a set of declarations which the author is able to invoke simply by placing a line such as:

```
<!DOCTYPE message SYSTEM "message.dtd" >
```

at the top of the compuscript. The publisher (or someone else) should supply the author with guidance and examples as to how the document structure declared by the designer is to be used. The author should not need to consult the invoked prolog. Ideally, the publisher's tag set and its description ideally should, together with general SGML guidance, be enough to allow the author to mark up the document instance. (A proviso. Specialised *data content notations*, such as for mathematics, may require additional non-SGML

C But wait, there's more

The designers of SGML wanted a scheme able to encode the most complicated document structures, which was at the same time easy to learn and implement for the simpler documents. They did this by giving SGML a number of parameters and optional features, to be set even before the prolog was read. For example, the powerful CONCUR feature allows a single document to support two independent tagging schemes! For a historic printed (or manuscript) Bible or other text one might wish to record not only the division of the text into books and verses, but also into pages and lines!

Each SGML document should, to be really official, begin with an SGML declaration, whose purpose is to state which of the optional features are in use, that `<` and `</` (or some other character strings) are used to open start- and end- tags, and so forth.

The SGML declaration will occupy about a page of text. It begins

```
<!SGML "ISO 8879:1986"
```

which specifies the version of SGML being used. Most likely, as it wasn't born perfect, there will at some future time be a revision of SGML, but for now there is only one version to choose from. It is expected that many many documents will share the same SGML declaration, and that all documents on a single system will share the same declaration. For these reasons, the standard allows the system to *imply* the SGML declaration, which means that should no SGML declaration be supplied by the document, the system can provide one of its own choosing. The prolog and document instance which follow had better conform to the chosen SGML declaration. There is also an SGML declaration for each SGML application, which records its SGML capabilities.

Once the various features of SGML have been understood, the SGML declaration becomes a control panel that turns them on and off, sets delimiter strings such as `<` and `</` and `<!`, and describes the character set used by the document. Such is a *concrete syntax*. Until then, it is perhaps best left to one side. The standard contains two SGML declarations, which define the *reference* and *core* concrete syntaxes respectively, at least one of which (Clause 15.3.3) must be supported by any conforming SGML system.

D Summary

In this article we have seen a complete SGML document, from prolog to the final end-tag, and we even had a glimpse of the implied SGML declaration. Different people have different interests in the document. The implementor is particularly interested in the SGML declaration. The designer will control the prolog. The author cares about the words in the document instance. The typesetter has broad concerns. He or she must understand the prolog and thus the tags in the document instance, the typographic qualities of the author's words (but not their meaning, if any), the tools supplied by the implementor, and also the designer's output requirements. The publisher is interested in everyone getting on with everyone else.

This article has said some about SGML, and some about the wider context in which it is used. The next will describe how to create some more complicated documents. We shall also look at SGML from the typesetter or application programmer point of view.

E Exercises

Finally, here are some exercises to test and develop your comprehension and understanding. First you might like to read through the article again, pretending that you are some other person involved in the publishing process. Particularly someone with whom you are having difficulties you do not understand or sympathise with or, failing that, a colleague. You might also like to write down all the *emphasised* definitions, and express them in your own words.

Exercises

1. When was the SGML standard published?
2. Who are the five persons involved in text processing?
3. Who is the sixth?
4. How does SGML differ from other computer languages?
5. What is the purpose of the prolog? Who writes it?
6. What is the purpose of the SGML declaration?
7. What is the purpose of the document instance? Who writes it?
8. How many lines in a sonnet? Is that every sonnet?
9. What is the difference between *mark up* and *markup*?
10. What is a declaration?
11. How is SGML a compromise? Between whom? Is this good or bad?
12. (Experts only) In explaining SGML for beginners, have I made any false statements?

Answers in the next article.

VIII Requiem for Aston

Malcolm Clark
Chris Rowley

The recent move of the UK CTAN node from Aston University to Cambridge University brings to the end a very long relationship between TEX and Aston—a relationship that brought great benefits to both the University and the TEX world.

The beginnings of this association go back to 1987 when, with the active support of the Vice Chancellor, Peter Abbott established a TEX archive based on the VAX systems at Aston—it then amounted to 200MBytes of data.

He also organised experts on various aspects of maintaining such a system into a volunteer group of 'archivists'—and made Aston a thriving and friendly focus for their activity. These volunteers came from around the UK and, later, also from 'mainland Europe', whilst knowledge of the archive and its benefits rapidly spread throughout the world. Thus the university became, in a very real sense, the centre of a world-wide effort to produce and distribute high quality software that was greatly needed and appreciated throughout that world.

Within a few years, the collection of software, together with the traffic created by users, had grown to such a size that it was no longer possible to host it on the existing equipment. To solve this problem of being 'too good', Peter successfully negotiated the donation of equipment by DEC and other suppliers. This enabled the Aston Archive to continue its rapid growth and thus to evolve into a mature, well-managed system.

A significant development was the arrival on Peter's desk of a Sun Sparc in mid-1992. This was quickly put to use as a fully (archive-) functional second platform and proved to be well suited to further development of the service; thus it was that Aston became the first site in the world to implement the then on-going discussions of the TUG Technical Working Group on Archives. During 1992–1993, first George Greenwade implemented this same TWG structure at Sam Houston State University, Texas and then Rainer Schöpf implemented it for DANTE at Universität Stuttgart, Germany (and subsequently on DANTE'S own machine). This pioneering work resulted in the first multi-site archive of this size to be successfully implemented anywhere; it was especially appropriate that the announcement of the completion of this major technical achievement could be made at Aston University, during the international TUG conference in July 1993. Thus the project started by Peter as a service to the UK community had become, in six hectic years, the internationally-acclaimed, state-of-the-art Comprehensive TEX Archive Network (CTAN).

Access to the archive has been further enhanced, and gained a completely new public, through cooperation with Prime Time Freeware, who regularly produce CD-ROMs containing the contents of the archive—or to be precise, almost all: it's now too big to fit on one CD!

Peter was, of course, also involved in many other aspects of making this vast range of high-quality software easily available to as wide a range of people as possible. In particular, he was one of the founders of the UK TEX Users Group and, here again, the reputation of Aston University was enhanced by the many ways in which he was able to support and guide the fledgling organisation. His valuable activities on behalf of the group have, of course, not been diminished at all by his retirement.

Both the archive, in its new role as a CTAN node, and Peter Abbott will continue to provide a large range of services to promote the rapidly growing use of TEX throughout the world but, regrettably, Aston University will no longer be part of this important hub of expertise and service which over many years contributed so much to its reputation as an international centre of academic and technical excellence.

IX Report on 1994 EuroTEX in Gdańsk

Michel Goossens
CERN
m.goossens@cern.ch

After a Fokker 50 took me in about one hour from Copenhagen to Gdańsk, it was about midday on Sunday September 26th, when I stepped out of the plane and was greeted by a beautiful blue sky and summer-like temperatures. A trip by taxi of 41 km around the southern part of Gdańsk brought me in about 30 minutes to the Orle holiday center in Sobieszewo, a resort on the Baltic some 20 km east of the city. The hotel was located 100 metres from the beach, and already straight after lunch I had the pleasure of walking along the wide sandy beach in search of amber and shells, while discussing TEX and other text processing related problems with several colleagues and friends of the TEX world.

The next day, the Monday, was kept free to allow people to register and to meet one another. On arrival everybody was given a copy of the Proceedings, and an extremely useful typographic ruler, and the traditonal mug, with the specially designed EuroTEX94 logo, showing a paper origami boat in the foreground, symbol of Gdańsk and its famous shipyard, on a dark background (the Baltic sea), and the words EuroTEX'94 in a light sky at the top. Very nice stylistic work, indeed!

The conference had a total of 57 participants coming from 15 different countries, with the Polish (19) and German (13) representations being the largest, while there were also TEX users from Belgium, Canada, the Czech Republic, France, Hungary, Lithuania, the Netherlands, Norway, Russia, Slovenia, Switzerland, the United Kingdom, and the United States. All those who had arrived by 1 o'clock in the afternoon were taken by bus on a guided tour of the most interesting parts of Gdańsk.

The formal opening of the conference was on Tuesday morning, and after the usual opening speeches by the organisers, Petr Sojka walked us through the various aspects of hyphenation with TEX and described the significant success that has been obtained recently, especially in the case of multi-lingual documents. Bernd Raichle demonstrated how useful it is to use TEX's mouth to process data and showed how he had applied these ideas to implement a quicksort algorithm. Klaus Lagally, the author of the multi-lingual ArabTEX system, explained in his talk how he solved, staying within standard TEX, the problem of line-breaking inside paragraphs with text that runs both from right to left (like English) and from left to right (like Arabic or Hebrew), thus providing a really portable solution. The techniques described were successfully used to process a forty-page paper containing mixed English and Hebrew with some Arabic and even Latin or other short language fragments, and it showed how flexible and powerful the basic TEX typesetter really is. Just before lunch Marion Neubauer told us about her experiences with converting Word and WordPerfect documents from and to LATEX and I am sure many of those present would agree with her finding that unless the elements of the document are already clearly marked up in the source, the LATEX document obtained was hardly usable, and that converting rather complex documents is in any case a time-consuming process. The answer might be using an editor in conjunction with an internal conversion program.

The afternoon session started with a description by Olga Lapko of the METAFONT package developed at Mir Publishers in Moscow and distributed as part of the CyrTUG-emtex package. It contains the METAFONT sources of a completely redesigned Russian Computer Modern-like font family, which is more adapted to Russian typographic tradition than previous Computer Modern Cyrillic fonts. Yannis Haralambous then gave a detailed overview of the Ω system, a 16-bit extension to TEX that uses the Unicode standard as internal encoding and allows multiple input and output character encodings. He described various applications, including calligraphic poetry, mixtures of languages with many special characters, vowelized Arabic, fully diacriticized Greek, and correctly kerned Khmer. Finally, Kees van der Laan gave us an overview of his BLUe's (Ben Lee User) Format. At the user level this new format is supposed to facilitate the creation, formatting, exchange and maintenance of compuscripts during the complete lifetime of a publication. The format is easily customizable and provides for the possibility of having cross-references using a one-pass process. I found it an interesting approach since Kees introduced many ideas from modern software engineering practice.

That evening, we had the traditional EuroTEX banquet, with a lot of atmosphere, champagne, wine, plenty of beer, good food, guitar playing and singing by several of the participants — in a word (or two) — a hell of an evening, with TEX and LATEX (almost) forgotten and other themes such as family, children, politics, philosophy, "real life", in short, becoming the main subjects of the evening, and it was not before we were all convinced that we had solved all of the world's problems that we went to bed in the early morning hours.

So, the next morning at breakfast, it came as a shock to many of us that there were still a few unsolved problems left (mainly in the area of TEX, of course) and so we decided to continue the conference and turn our attention to the niceties of colour techniques and their realization in LATEX 2ε. I emphasized in my part of the first talk of the day that colour is rarely needed in normal text, and that when it is used, the function of each colour should be unambiguously clear. There exist many (empirical) rules about colour harmonies and only a lot of experience and practice allows one to become an expert in this field and apply colours efficiently. Sebastian Rahtz then showed how LATEX 2ε implements a few simple tools for obtaining colour and he presented some nice examples. Janusz Bień gave an overview of different standards connected with the Polish language (keyboards, character encodings, localisation, fonts layout) and put them into perspective relative to developments of the international standards bodies. Yannis Haralambous described his Tiqwah ("Hope" in Hebrew) system for typesetting Biblical Hebrew, going into some detail on issues of font design, classical Hebrew typography and the user interface. He hoped that his system would help to revive the interest in Biblical Hebrew typography. Vladimir Batagelj gave an introduction to the PostScript language, and presented some of his experience in combining TEX and PostScript. Karel Horák gave an overview on how one can decompose large METAFONT pictures into smaller fragments and described techniques to place them on a page in a seamless way. He stressed the importance of resolution dependence and hoped that the new versions of some of the drivers would eliminate most of the constraints of this powerful approach where one only uses TEX and METAFONT to generate pictures, thus making the whole document fully portable.

Bogusław Jackowski and Marek Ryćko showed some extremely nice and pleasing pictures made with METAFONT (and their paper was rightly given the prize of Best Paper in a ballot amongst all conference participants at the end of the conference). They demonstrated some examples of non-standard METAFONT programming and advocated the creation of libraries of METAFONT routines, that would make the use of METAFONT as a universal drawing tool much more appealing. Éric Picheral, who looks after the Unix part of the GUTenberg TEX archive, gave an overview of the history of that archive, the various steps required to adapt TEX and its companion programs to the needs of the French-speaking TEX user community, and the way the various versions (Unix, PC, Mac) are made available to users worldwide via the Internet (ftp, http/www, gopher).

Lutz Birkhahn discussed his work on developing debugging tools for TEX and presented Tdb, an extension to TEX that provides an interface to the Tk/Tcl X11 toolkit. This allowed him to set up a graphical user interface to allow one to set breakpoints, have stepwise execution, and to look at macro definitions and the value of variables. The last talk of the day was by Philip Taylor, who advocated the virtues of defensive programming for TEX since in the real world one cannot assume that user code or input is correct. Hence it is the task of the programmer to make sure that the results of developed macros are as close as possible to those the users expect. Defensive programming techniques let the programmer anticipate both errors in data and flaws in algorithm design.

During the first talk on the last day of the Conference I had the pleasure of giving a 20-minute talk about the lessons learned when writing *The LATEX Companion*. I once more tried to emphasize the importance of generic markup for all logical elements of the document. Also, the global design of the book should be discussed at an early stage, while formatting decisions should be left to the final stage of running the chapters into pages. Wietse Dol and Erik Frambach then gave a very impressive talk-demonstration of their 4TEX workbench, that also forms the basis of the extremely succesfull NTG CD-ROM. It is without doubt the best integrated TEX system for the MS-DOS world, and many participants who wanted to know more about the system also took part in the full day tutorial that they ran on the Friday. Jörg Knappen discussed work going on to standardize the IPA characters, and advocated the creation of a 256-character IPA font for use with TEX. Jiří Zlatuška talked about work he was doing within the framework of LATEX 2ε to allow different languages and encoding schemes to be used together in a same format, at the same time providing mechanisms to switch freely between languages and encodings. Friedhelm Sowa showed his approach to generate colour pictures, especially on cheap printers. He discussed how the dvi driver must be colour conscious and gave as an example the dvidjc drivers and the latest version of his BM2FONT program, that provides the four primary colours of the pictures by generating four different bitmap images. He showed some quite impressive pictures as examples of his approach but he pointed out that colour is not simple to realize and great care must be taken to obtain the effects one really intends.

The afternoon session began with two presentations about LATEX 2ε, the first by Johannes Braams, who gave a clear introduction to the principles of class files and packages, as he showed using simple examples how it is possible to transform LATEX2.09 styles into LATEX 2ε classes and packages. Dag Langmyhr, in the second talk, gave an explicit example of how to roll one's own complete LATEX 2ε document class, and detailed the various stages of building up the necessary ingredients by borrowing from existing examples, introducing (small) changes into existing constructs, and incorporating the functionality of supplementary packages.

Before the official part of the 1994 EuroTEX Conference came to a close, Philip Taylor and the εTEX and NTS team presented an overview of the present status of these two projects. The first one is based on the existing TEX code, and plans to extend it in various areas, while keeping 100% backward compatibility with TEX for those who want it. The NTS project, on the other hand, seeks to first reimplement TEX in a list language, so that several alternative approaches to the various components that build the system can be more easily tested. In the longer run it might thus be possible to develop a New Typesetting System (hence the name) that will be at least as good as TEX, but that extends or improves TEX in areas where the latter is considered too limited.

These last eighty minutes or so about futures were followed by closing remarks from Włodek Bzyl and Philip Taylor of the Organizing Committee, who announced the winners of the Best Paper contest (see above), who were given a bottle of vodka with tiny pieces of gold floating inside (a local speciality), and the venue of the next Conference, that is to take place next year somewhere in the Netherlands (possibly in the (now) famous town of Maastricht).

Yes, it had been a good conference, and quite different from the 1994 TUG Annual meeting, whose theme was “innovation”, so that many papers described more or less exotic, front-line developments (colour, sophisticated page layout, object-oriented techniques), while in Gdańsk more attention was given to practical issues of typesetting in multiple languages and working with cheap printers and machines (hyphenation, the use of METAFONT, MS-DOS related developments). I can only congratulate the local organizers, especially Wlodek Bzyl and Tomek Przechlewski, for their nice work, and I hope that this conference has contributed to making TEX better known in Poland, and that those present will take with them the “spirit” of Sobieszewo, where it was shown how to put principles into practice to make progress in the field of applying TEX in real-world applications (a copy of the 200-page Proceedings of the EuroTEX94 Conference, which, as mentioned above, were available on the Monday thanks to the hard work of Tomek and Wlodek, can be obtained by sending 15 DM (postage included) to Wlodek Bzyl, Instytut Matematyki, Uniwersytet Gdanski, Wita Stwosza 57, PL 80-952, Poland).

See you next year, some time, somewhere, they were saying. And indeed they will.

X Malcolm's Gleanings

Malcolm Clark
Computing Services, University of Warwick
Coventry, CV4 7AL
`m.clark@warwick.ac.uk`

A True colours?

The boost given to the use of colour in LaTeX 2ε is not an unmixed blessing. There are pitfalls and problems lurking round these new corners. There is always a huge discrepancy between the colours we can produce on a monitor screen and the hard copy version that a printer will produce. It can be very frustrating to spend time and effort getting the colour balances 'just right', and then finding that the hard copy looks nothing like the original. Especially when you have to pay for the hard copy. Why are there discrepancies? The mismatch is in the very nature of the processes involved – it is not because something has gone wrong.

Let's start at the beginning, and assume that we are concerned with the transition from colour on the screen to colour on the page. Perhaps the first thing we should realise is that no two people see colour in quite the same way. Intra-species perception is well-known to be different, but even between individuals (ignoring those with some physiological perceptual malfunction, like colour blindness), response to different colours is different – *i.e.* what different people call 'blue' differs – well, most people agree about blue, but how many agree about 'mauve' or 'puce'? Any perceived colour may be understood as the sum of three primary colours. This gives us a first clue, since it means that we can represent a colour as the sum of three stimuli (a tristimulus), and we can think in terms of characterising any colour by locating a unique point defined by three (orthogonal) axes. In fact no monitors can represent the full range (or *gamut*) of colours. Their actual performance falls short. Even before we start, we cannot hope to represent all possibilities.

The colours we view on a monitor are emitted colours – they are generated by the three electron guns of the device (there are other technologies, but the principle remains much the same). The advantage here is that each model of monitor can be calibrated, and in fact the range of possible colours can be determined. From time to time the colours may drift, and it is possible that adjacent 'identical' machines may appear to have different colour outputs. If you always work on the same machine, you can guard against this. Recalibrating every machine is rather time consuming. Another advantage of using the same machine every time is that the background and lighting is unlikely to change to much.

By virtue of our evolution, our eyes are well adapted to slow changes in light level, and also to slow, but moderate changes in the major illuminant (as daylight varies through the day, for example). Changing the incident lighting does not change the emitted light on the monitor, but since the frame of the monitor and everything else in the room is changed by changing the illumination, you can end up thinking that it looks different. The illumination in many rooms is a bit arbitrary: it may be fluorescent tubes, which have a rather limited spectrum, or it may be normal incandescent lights, which have a different spectrum, and occasionally it is north light (also known as *daylight*) illumination.

This last is an interesting diversion. This is the lighting condition under which most, if not all, standardised colour determinations are performed. Anyone who is working seriously with colour will use north light (most artist's studios, for example, will be oriented to be lit naturally in this way). In fact, this is one of the reasons why the hard copy will appear incorrect: the lighting conditions are likely to have changed. And anyway, the hard copy will look different under different illumination. The hard copy is made visible by reflected light. What is reflected is dependent on the light source. The distortions of sodium street lights are well known, but the same sort of effect is true of fluorescent or incandescent lights. The north light works reasonably well as a standard because of our environmental background, and because it is based on the light source which was available as our eyes evolved.

There is at least one more factor to be considered: the printer. Just as a monitor can only produce a proportion of the possibilities, so too the printer is restricted in its range of possible colours. In fact it is worse: while the monitor can use the linear additivity of the primaries, the printer's primaries are anything but linear, and the mixing is subtractive. Adding the primaries on a monitor gives white: on the printer it will give a theoretical black (more of a muddy purplish darkness). It is therefore rather difficult to transform from the co-ordinates representing the colour on screen to some faithful (whatever that means) rendering on the page. There are lots of dodges which can be adopted. The most comprehensive is to employ some sort of lookup table. This depends on the measurement of many samples – it is a lot of rather tedious work, spread over lots of subjects (remember the underlying psycho physical variation).

Briefly then:

the colours you see on the monitor are a subset of possible colours;

1. the illumination of the room can be important
2. the printer does not have the capability of rendering all the colours visible on the monitor;
3. you will probably view the hard copy under a variety of lighting conditions.

Is there any hope? Some printers are better than others. The machine we use here at Warwick, a Tektronix Phaser, is essentially a wax crayon machine; it is reasonably good, for the price. The best hard copy devices are dye sublimation printers and they are very expensive. If you stick to primary colours, and colour gradation is not a key issue, the Phaser is excellent. With a lot of time and effort you might achieve finer quality work with it. If you are looking for a photographic quality reproduction you would have to use other technologies – and even then, you can fail. In many cases though, hard copy is irrelevant. The display medium is genuinely the monitor screen. If we think in this way, we realise that many other possibilities are introduced, like running video clips, or even pieces of software, within a document or presentation (yes, LaTeX 2_ε does not yet support such extensions explicitly, but a suitable `\special` could). We usually have the feeling that paper is the objective, but it does not have to be. Sometimes it is more convenient, but at other times it is restrictive.

Any of the good ideas for this came from presentations by Chris Lilley of the Computer Graphics Unit at Manchester Computer Centre; the mistakes are mine.

B Trivial pursuit

For many years I have been toying with the notion of creating a special TeX edition of *Trivial Pursuit*. There are those who contend that TeX *is* a trivial pursuit, but I will have nothing to do with this view. To assist in the creation of this game, and to tempt the likes of Waddington's, I will start to include examples of the questions in this column. The first question is in two parts. The first part is borrowed from the ACM 1994 Computer Bowl quiz: "Only one person in all of computing's history has ever won both the ACM Turing Award for lasting technical achievement and the Grace Murray Hoppper Award for work done prior to reaching the age of thirty. Who is that person?"

The second part is this: "The Computer Bowl quiz was created and produced by the Computer Museum, Boston. What link does this have to the answer for the previous question?"

C Dutch bearing gifts

I quickly snapped up my NTS CD-ROM when it was offered through the group. I was motivated by a curious mixture of support for the efforts of another group, and a willingness to get tangled up in a flashily trendy technology, although I have to admit that I'm not yet convinced by CD-ROM technology.

One advantage of the NTS CD-ROM is the booklet which comes with it. Once you have installed the suite, and basically it wants to run from the CD-ROM drive, you find that it is enormously configurable. Because of the many varieties of printers, screens and editors which are available in DOS systems, you could have many happy hours ahead of you getting things right. I get the feeling that it is assumed that you will like fiddling. There is no doubt that great effort went into this, and equally, that you can eventually install all the relevant bits and pieces. I wonder how you would manage to install something useful if you didn't want to use the 4DOS shell which is the default. Clearly the developers thought this an unlikely decision. They say "One might object that using 4DOS batch files deprives the old-fashioned `command.com` users from the benefits of 4TeX. We happen to think that this would only be a mild punishment for not recognizing how good 4DOS really is." Well, pardon me! Oddly, they fail to mention that continued use of 4DOS involves payment of a shareware fee.

Once you do get it installed, you still appear to have great power to customise the options the shell offers. However, I am still unclear how to change the drivers (you are provided with a variety of screen drivers and I really wanted the one which occupied least memory; this is not the default). I was also surprised that among the huge range of choices of formats you are offered, the simple LaTeX 2_ε or plain TeX is not among them. It is rather like going into Macdonald's and trying to get a burger without the limp lettuce or the ketchup. In this case of the formats, the lettuce and ketchup is Babel.

On the whole, I suspect the shell is about as good as you can get with DOS. It appears workmanlike. I'm afraid it does look dated though. Windows is here. The true BLUe TEXie may resent it, and see it as a step towards perdition, but for many of us Windows is the least unacceptable face of the Intel chip's range of operating systems.

But compared to PrimeTime's CD-ROM, this is a joy. At least the NTG's offering has the advantage that you can browse it a bit like you browse CTAN. If there is a useful file there, you can find it and pull it out. The PrimeTime CD-ROM is tarred and feathered (or zipped) so that you can't actually find anything unless you know where it is. It contains almost all the contents of CTAN, but nowhere does it actually give you the date of this snapshot. The sheer mass is daunting. And layered on the top of this is an assumption that UNIX is the way the truth and the life. It makes a point about distributing source code: "Binary-only distribution prevents recipients from modifying or learning from the internals of software." I have to confess that I'm not terribly interested in computer programs. I'm much more interested with what I can do with the program. The days of immersing yourself in the exciting details of source code should surely be passing. The terms 'nerd' and 'propellor-head' or 'techno-weenie' spring to mind, almost unbidden. People you would prefer to avoid at parties. Like the ones who want to tell you the latest exciting details of LATEX 2ε or dvips. If I have to examine the internals to work out why TEX inserts a skip at some particular point I'll drift silently to Quark Xpress. [8]

What is the basis for my rant? First, virtually all the discussion in the handbook is in terms of UNIX. This is partly fair, since PrimeTime admit that their experience is almost wholly with UNIX, and does not really spread to other platforms. Until you actually look at the CD-ROM itself (and yes, it is readable under DOS, Windows and the Mac), you don't realise that it might just be useful to you. There are the tools needed to unzip the many files on platforms other than UNIX. I was mildly amused that the Windows unzip application came as a `.z` file itself (*i.e. zipped*), but since there is also a DOS executable, that is hardly insuperable. The Mac version of unzip came in two versions – one a BinHex file, and the other a self extracting archive. The latter is the one that most of us would prefer. Those of us who enjoy the Mac tend to enjoy it because we don't have to mess around. We can get straight on with the real jobs. In order to use this archive you woud have had to dust down your copy of ResEdit and mess around changing file types: this is close to binary editing. It is not the sort of thing you do every day. So I BinHexed it, and there was the archive, which worked wonderfully. Except. There is always an except. Turning it loose on a part of the CD-ROM, on a genuine zip file, it just wouldn't work until I realised that you had to copy the zip file to your hard disk and then unpack it. Otherwise you get unintelligible errors. This is clearly a failing of this particular implementation of unzip, but it would have been 'friendly' if the manual had pointed it out. After all, someone must have tested it all out, mustn't they? This means you have to waste disc space on the zipped and unzipped files together.

This aside, my main complaint is that to find anything you must first unzip the file you think it might be in. Since the unzipped files would amount to two gigabytes or so, you cannot really just unzip the whole archive and browse. Maybe I'm being uncharitable. After all, if you want a particular driver, you look for that driver, and not another one. Unzip it and off you go. No-one really wants to browse through all the bits of particular drivers, do they? But think, there's all that source to modify and learn from. I can hardly wait.

The text of the handbook is largely taken up with technical details, but has a foreword George Greenwade, the *emminence grise* of CTAN, a note on TUG by Sebastian '*Il Presidente*' Rahtz (in uncharacteristically mellow and benevolent mode), and one of Kees van der Laan's idiosyncratic expositions entitled 'What is TEX…'. There is no doubt of Kees' enthusiasm. He writes at a frenetic pace which often leaves the tedious details of conventional sentence structure far behind. You will either find it exhilarating or debilitating. At times he seems capable of raising himself several feet off the ground purely by belief and pace alone. The ground looms up very large and unfriendly towards the end.

I'm a little bemused by it all. In both cases, opportunities seem to have been missed. Or perhaps worse, TEX is again cast in the light of the avid techno-whizz. Both of these offerings are just barely useful, and save themselves from consignment to one of the outer Hells. What is it that we should be doing? The contents of CTAN are undoubtedly convenient, but the compression of the PrimeTime disc does erect a barrier which is too high. Better to trim it mercilessly but make it more accessible (yes, I know there is a problem with depth of directories: CTAN goes deeper than the CD-ROM format will allow). The NTG CD-ROM, thanks to the shareware component, is just waiting to turn sour. The scrupulously honest will have difficulty creating an alternative integrated TEX system. And anyway, it still isn't Windows! Time to get into the 90s, before the century ends. But at least it is possible to find useful files before you find them disc space.

[8] I n passing, if I had put a fraction of the effort into Quark that I have in TEX and LATEX, I would be emminently employable at any number of publishers. On the other hand, I would probably also have a pony tail and an earring.

XI Maths in LaTeX: Part 2, Getting more Serious

R. A. Bailey
School of Mathematical Sciences
4. Queen Mary and Westfield College
Mile End Road
London E1 4NS

Recall

This is the second in a sequence of tutorials on typesetting Mathematics in LaTeX. It includes some things which can be found in *LaTeX: A Document Preparation System* by Leslie Lamport, but I am gradually working in more things which, while straightforward and necessary for Mathematical work, are not in *The Manual*. In case you missed the first tutorial, two warnings are now repeated.

I expect you, the reader, to do some work. Every so often comes a group of exercises, which you are supposed to do. Use LaTeX to typeset everything in the exercise except sentences in italics, which are instructions. If you are not satisfied that you can do the exercise, then tell me. Either write to me at Queen Mary and Westfield College (my full address is given at the end of this article) with hard copy of your input and output, or email me at `r.a.bailey@qmw.ac.uk` with a copy of the smallest possible piece of LaTeX input file that contains your attempt at the answer. In either case I will include a solution in the following issue of *Baskerville*: you will remain anonymous.

A word on the controversial issue of fonts. Fonts in Mathematics are handled differently in LaTeX 2.09, in NFSS, and in the new standard LaTeX, LaTeX 2_{ε}. Rather than compare these systems every time that I mention fonts, I limit myself to LaTeX 2.09. When you upgrade to LaTeX 2_{ε}, all these commands will still work, so long as you use the standard styles `article`, `report` and `book`.

B What does it mean?

10 Arrows

Arrows are relations. Four of them are

`\leftarrow`	$\leftarrow$	`\longleftarrow`	$\longleftarrow$
`\Leftarrow`	$\Leftarrow$	`\Longleftarrow`	$\Longleftarrow$

and eight others can be obtained by replacing `left` by `right` (all versions) or by `up` or `down` (not the long ones). A full list is given on page 45 of *The Manual*.

Because of its frequent use in defining functions, `\rightarrow` has the short alternative name `\to`. It should not be confused with `\mapsto`, which is also used in defining functions.

`x\mapsto g^{-1}xg` $x \mapsto g^{-1}xg$

The relation `\iff`, which does indeed stand for the relation ‘iff’, is not quite the same as `\Longleftrightarrow`: it has a little more space on either side.

```
gh^{-1} \in K \iff g \in Kh
```

$$gh^{-1} \in K \iff g \in Kh$$

11 Fences

Mathematical typesetters use the word *fence* for anything like a bracket that comes with a mate to enclose part of a formula. TeX calls them *delimiters*. Obvious examples are parentheses, brackets and braces, produced with `(`, `)`, `[`, `]`, `\{` and `\}` respectively. Angle brackets are produced with `\langle` and `\rangle`:

`\langle u,v \rangle` $\langle u, v \rangle$

Don't try to use the keyboard symbols < and > for this: apart from the fact that they do not look right in typeset Maths, (LA)TeX does not think that they are fences and may well split the line between the $<$ and the u.

Use `\lfloor`, `\rfloor`, `\lceil` and `\rceil` to obtain 'floor' and 'ceiling':

`\lceil 3.75 \rceil = 4` $\lceil 3.75 \rceil = 4$

Other fences are shown on page 48 of *The Manual*.

Fences need to grow to enclose large formulas. They will automatically grow to the correct size if you preface the opening fence with `\left` and the closing fence with `\right`. The `\left`s and `\right`s must come in properly matching pairs, but the fences which they qualify need not obviously match. Compare

$\frac{1}{n} \in (0,1]$ with $\frac{1}{3} \in \left(0, \frac{1}{2}\right]$:

the second half-open interval is produced with

`\left( 0, \frac{1}{2} \right]`

The solidus / is really a binary operator, but it is treated by TeX as a fence, partly because traditional typesetters do not put the same amount of space around it as they do around other binary operators, partly because it needs to grow when it is between two tall formulas. But it can't grow unless it is matched with another fence, and it doesn't need one. So TeX makes the full stop into an invisible fence (called the *null delimiter*) when it is preceded by `\left` or `\right`.

```
a(b+c) \left / \frac{1}{xy} \right.
```

$$a(b+c) \left/ \frac{1}{xy} \right.$$

How would the above expression be different if you typed `\left. a(b+c) \right /`...?

Sometimes you need to use `\left` and `\right` just to tell TeX that you are using fences, even if you do not need them to change size. A good example is $|\ \ |$ used for the modulus or cardinality functions. If you type `|+3|`, TeX will typeset the first $|$ as if it is being added to the 3.

`| +3 |`	$\|+3\|$
`\left| +3 \right|`	$\left\|+3\right\|$

12 Standard functions with English names

Some standard functions have written names with two or more letters, based on their full English name. An example is cos for 'cosine'. It is no good simply typing `cos`, for then the output will look like c multiplied by o multiplied by s. So there are standard commands such as `\cos`, `\sin`, `\log`, `\exp` and `\dim`.

`\cos\pi = -1` $\cos\pi = -1$

The following ten standard functions

`\max`	`\sup`	`\limsup`	`\lim`	`\det`
`\min`	`\inf`	`\liminf`	`\gcd`	`\Pr`

can have expressions above and/or below them to show what range of variables they apply to: these are typed in as if they were super- or subscripts.

```
\lim_{x \to 0} \frac{\sin x}{x} = 1
```

$$\lim_{x \to 0} \frac{\sin x}{x} = 1$$

`\max_{i=1}^{n} \theta_i` $\displaystyle\max_{i=1}^{n} \theta_i$

In displayed Maths the range expressions come above and below the name of the standard function; in text they come somewhat to the right: $\max_{i=1}^{n} \theta_i$.

The standard function 'modulo' has two forms:

`7 = 10 \bmod 3`	$7 = 10 \bmod 3$
`7 = 10 \pmod 3`	$7 = 10 \pmod 3$

The remaining standard functions are listed on page 46 of *The Manual*.

13 *Large repeated binary operators*

When a binary operator is commutative and associative it often has a a special symbol to show its repeated application: $\sum$ for repeated $+$, $\bigcap$ for repeated $\cap$, and so on. Some of these have special commands in TEX:

`\sum`	$\sum$	`\bigcap`	$\bigcap$	`\bigoplus`	$\bigoplus$
`\prod`	$\prod$	`\bigcup`	$\bigcup$	`\bigotimes`	$\bigotimes$

(Note that the operator `\sum` is *not* the same as the Greek letter `\Sigma`.) More are shown on page 45 of *The Manual*. Each of these has two sizes: the big one is used in displayed Maths (except inside arrays and fractions) and the small one is used in text. Each can also take sub- and superscripts to show the range of operation: as with the ten listed standard functions, these sub- and superscripts appear above and below the operator in displayed Maths (except inside arrays and fractions) and a little to the right otherwise.

```
x_1 + \cdots + x_n = \sum_{i=1}^{n} x_i
```

$$x_1 + \cdots + x_n = \sum_{i=1}^{n} x_i$$

```
M_1 \vee\cdots\vee M_t = \bigvee_1^t M_i
```

$$M_1 \vee\cdots\vee M_t = \bigvee_1^t M_i$$

But in text: $\alpha_1\alpha_2\ldots\alpha_m = \prod_{j=1}^m \alpha_j$.

14 *Integrals*

There are two integral signs:

`\int`	$\int$	`\oint`	$\oint$

These behave somewhat like the large repeated operators in that they have a larger size in displayed Maths and their limits are typed in as sub- and superscripts. However, these limits stay in the same position even in displayed Maths.

15 *More symbols*

There are some miscellaneous Mathematical symbols that we have not covered elsewhere. Important ones include:

`\emptyset`	$\emptyset$	`\forall`	$\forall$	`\Box`	$\Box$
`\infty`	∞	`\exists`	$\exists$	`\partial`	∂

More are shown on page 45 of *The Manual*.

Unfortunately, some of these, such as `\Box`, are not standard in the new standard LATEX. To continue using such symbols after you have upgraded, you must include the package `amsfonts`, if available, or the package `latexsym`. Some operators and relations have been similarly banished, and can be rescued in just the same way.

16 *Punctuation*

In Maths mode, (LA)TEX treats a full stop as an ordinary symbol, so that decimal points look right. However, the comma and semi-colon are treated as punctuation, and get a little space after them, but not so much as the space they usually get in text. To suppress the space, put the comma or semi-colon in a pair of braces.

The colon is treated as a relation, because of its frequent use in defining sets. To obtain a colon as a piece of punctuation, type `\colon`. Many people think that the punctuation form is more correct for defining functions.

17 *Braces for grouping*

Grouping has two extra properties in Maths mode. In the first place, it can prevent line-breaking. (LA)TEX is usually very good at knowing where to break a line within a Mathematical expression, but it doesn't always do it exactly to your liking. For example, in Exercise 7 in the first tutorial, one equation was broken after a $+$ sign, leaving the single character ε on the next line. This could have been prevented by enclosing the whole right-hand side of the equation in braces: TEX never breaks a line between grouping braces in Maths mode. Use this force sparingly: the more line breaks that you ban the harder is it for TEX to build beautiful paragraphs.

The second property is more subtle: injudiciously placed grouping braces can destroy TEX's fine knowledge of what is a relation, an operator etc., and thus prevent it from applying the proper spacing. `2{+3}` is not the same as `2+3` and neither is `2{+}3`; while `{}-4` is different from `-4`. Likewise, `{\cos} \theta` is not the same as `\cos \theta`: the former turns `\cos` from an operator into a symbol.

Some TeXies recommend always putting `{}` after a command with no argument. That advice can be disastrous in Maths mode. Look at

`\sum{}_{1}^{3} y_j` $\sum{}_{1}^{3} y_j$

and compare

`k \geq{} -b` $k \geq{} -b$

`k \geq   -b` $k \geq -b$

In the first example the sub-and superscripts have been placed on the `{}`, not on the `\sum`: in particular, they do not move to the correct place in displayed Maths. In the second, the minus sign has become a binary operator between `{}` and b instead of a unary operator on b.

This apparently awkward property of braces can be turned to advantage when TeX's first interpretation is not the correct one, as we have already seen in some examples.

18 Ties

The tie `~`, which prevents line breaks both within and without Maths mode, can be used to make your Mathematical text easier to read. A piece of notation consisting of a single letter or symbol should almost always be tied to one of its neighbours. If the notation names a noun, tie it to the noun: `group~$G$` and `element~$h$`. If it is the subject or single direct object of a verb, tie it to the verb:

```
If $v$~annihilates~$W$ then ...
```

If it follows a preposition, tie it to the preposition: `of~$\Lambda$`, `in~$T$`, `onto~$Y$`. If an adjective qualifies it, tie it to the adjective: `positive~$\delta$`. A notational adjective, particularly a number, should be tied to its following noun: `$n$~points`, `$21$~lines`. Be careful about the beginnings and ends of lists:

```
Only one of $A$, $B$ and~$C$ occurs ...
For $i=1$, $2$, \ldots,~$n$, put ...
```

Finally, use a tie if a small piece of notation ends a sentence, to prevent a line beginning something like

B. Therefore …

which makes it look like the second item in a list.

Try to get in the habit of inserting these ties when you first type the text: don't leave them until you see bad line-breaks.

Of course, like all rules in typography, some of the above rules conflict, so you have to use common sense about which ones to follow. Also, there will always be times when the paragraph simply cannot be broken into lines nicely if all the ties are observed: wait until the final run and then relax the least important ones.

Exercises

Exercise 11 The function $f\colon Z \to Z$ defined by $f\colon n \mapsto n^2$ is neither injective nor surjective. However, the function $g\colon R^+ \to R^+$ given by $g(x) = \sqrt[7]{x}$ is both.

Exercise 12 If $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_m\}$ are bases for $\mathcal{X}$ and $\mathcal{Y}$ respectively then

$$\{x_i \otimes y_j : i = 1, \ldots, n,\ j = 1, \ldots, m\}$$

is a basis for $\mathcal{X} \otimes \mathcal{Y}$.

Exercise 13 If $A = \{n \in N : n \text{ is prime}\}$ and if $B = \{n \in N : n \text{ is even}\}$ then $|A \cap B| = 1$.

Exercise 14 If z is any real number then $|{+z}| = |{-z}|$.

Exercise 15 *Express the following use large binary operators instead of dots. Try them both in text and in display.*

$x_1 + \cdots + x_{153}$	$1 + 2 + \cdots + r$
$1 + 1/2 + 1/4 + \cdots$	$p \times (p-1) \times \cdots \times 1$
$\mathcal{U}_1 \otimes \cdots \otimes \mathcal{U}_m$	$T_4 \cup T_6 \cup \cdots \cup T_{10}$
the sum of γ for $\gamma \in \Gamma$	$P_1 \wedge \cdots \wedge P_r$

Exercise 16

$$\prod_{k\geq 0} \frac{1}{(1-q^k z)} = \sum_{n\geq 0} z^n \Bigg/ \prod_{1\leq k\leq n} (1-q^k)$$

Exercise 17 *Redo Exercise 3 from the first tutorial without using built-up fractions. You may want to change the way the square root is shown.*

Exercise 18

$$\sum_{n=1}^{N} n^3 = \left(\frac{N(N+1)}{2}\right)^2$$

Exercise 19 $\int_1^2 \frac{1}{x} dx = [\log x]_1^2 = \log 2.$

Exercise 20 $\lim_{n\to\infty}(1+\frac{x}{n})^n = \exp(x).$

Exercise 21 If Z is a normal random variable with mean μ and variance σ^2 then

$$\Pr(Z < x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{\sigma^2}\right).$$

Exercise 22

$$\frac{\partial e^{-xt}}{\partial t} = -x e^{-xt}.$$

XII Report on European Writing and Computers Conference

Allan Reese

The European Conference on Writing and Computers held in Utrecht in October attracted more than one hundred and fifty assorted academics, predominantly psychologists and educationalists. This conference was formed by combining two events: the biennial meeting of the Special Interest Group on Writing of the European Association on Learning and Instruction (EARLI SIG), and the annual meeting of the European Conference on Computers and Writing (ECW). A third 'event' took place in parallel, an on-line conference hosted on a computer in the United States.

On top of these various groups and starting points, the conference was organized within twelve conference themes ('What is Writing?', 'Writing as a learning tool', 'Effective Instruction', etc.) and with a variety of formats; there were more than a hundred papers, posters, workshops and demonstrations. Participation was like using hypertext; unless you stuck to one theme it was a matter of browsing round, following up leads and trying not to regret the parts you missed. The organizers distributed a bound set of abstacts some weeks before the conference. This was a great aid in planning and is the main record of the conference, as no direct 'Proceedings' are planned. Delegates were instead asked to contribute chapters for three books on themes relevant to the conference. These books are planned for publication in 1995.

The emphasis through the seven ECW conferences has shifted from computers and software, to writing and the writing process. In now linking with a SIG representing mainly academic research, the writing itself ceased to be a focus of interest. Few of these researchers considered writing as a product to be *read*; text and writers are *data*. There were studies of: how children form sentences; the types of grammatical errors made by students learning a second language, or learning *in* a second language; linguistic forms used in academic papers and how these offer cues to a social subtext; comparisons of language used in formal and informal writing. No one attended the conference as an 'author'; everyone was an observer. The poor quality of presentation in many posters was striking — people talking about effective communication themselves ignoring all the necessities of the medium. As usual speakers abused the overhead projector with illegible or unintelligible foils.

The positive side was that within the anarchic and fulsome atmosphere of pure research there were so many opportunities to seek out exhibitors and discuss ideas. Compared to a typically British timetabled and regimented sequence of papers leaving few minutes for questions before proceeding, most of our time was spent in parallel poster sessions where you could spend an hour on twenty topics or just one. One person had been deputed as a 'discussant' on each theme and in the final sessions led a discussion, which helped in shaping all one's impressions into a coherent whole. Necessarily, these discussions did not lead to consensus or firm conclusions!

I looked particularly at tools to assist writers. The presentations caused me to wonder whether there is a strong divide between software for training writers, and that used in the production process. Adult and professional writers now have well-known products such as MS Word and WordPerfect that subsume aids like spell-checkers and thesauruses. The research products on display were not in that league and had been developed in very constrained and particular circumstances: a program to encourage story writing in early teens; a program to guide engineering undergraduates in planning technical reports; programs to teach journalistic style. One contemporary strand (all round the world) is the observation that traditional writing instruction based round constructive feedback and revision is time-consuming and expensive; everyone would like a computer-based alternative that is cheaper and more readily available. The irony is that the software displayed would sink without trace without considerable support from the teaching researchers.

The research shows little sign that computer-aided instruction (CAI) would be superior or faster. CAI may be appropriate in well-structured situations, areas where one might consider using SGML to ensure completeness and adherence to required formats. But can it be used to encourage reflective and original writing? The researchers didn't talk about the software they used for their own writing — again, this odd, clinical decoupling between objective and subjective observation — and the only time I discussed LaTeX was with a UK delegate who had had a book mangled by a publisher. Many of the craft skills mentioned — for example, teaching undergraduates to write a table of contents to assist in planning their report — are well-supported in the LaTeX philosophy; I felt that the psychologists should break out of their research clique and look at pragmatic rather than conceptual solutions.

The keynote address widened the field to include pedagogical applications of general-purpose software. We can encourage students to use commodity software in all subject areas; this is one aspect of information technology as an enabling rather than prescriptive tool. Students of literature, for example, might build a database of ideas and images in poems, and through this discover trends or associations. David Jonassen (Penn State Univ) linked this to the constructivist view of education — don't just 'teach facts' but train students *how* to make their own sense of the world by individually using 'cognitive tools'. The teacher operates as facilitator and mentor, not an authority. It was a stimulating talk and the skills described would assist able students in pre-writing organization of material.

As a writer and teacher of writing, I took comfort that the conference confirmed there is still no magic shortcut to learning the craft. How do you get to be a writer? Several contributions addressed points relating to providing feedback from teacher to student. They discussed the barriers to communication, the social context, psychological and cultural factors. The teacher-pupil relationship is mirrored in professional spheres, such as employer-staff and editor-contributor. It was very interesting, but would relate equally to teaching in any subject. Is the teacher being constructive, or expressing power? Good writers say they are writing for themselves, but it is a gift or a skill to use yourself as a critical audience.

My highlight was Jack Selzer's (also Penn State Univ) paper *Scientific and technical writing in a post-modern era.* What does post-modern mean? It's a jargon term for texts that challenge and break the conventions of 'modern science'. "Where conventional writing is sober and restrained, post-modern ones are playful, extravagant, exuberant.. . . Where conventional scientific writing prizes consensus and agreement, these unconventional ones call for pluralism and voice conflict.. . . [They are] unpredictable and exploratory." Too much writing nowadays is formulaic and conventional, in form and content, to the point of parody. Rules is rules, but writers must understand the rules and not apply them blindly. If students are not to confuse computer writing with computerized writing, these are the texts they should study. Take a random example: *The TEXbook.*

19 Postscript

I use Correct Grammar (CG) as an aid in proof-reading and polishing. It measures the above text as 'fairly difficult', with a US reading level of 14th grade. It found seven passive sentences, of which I changed one and clarified the meaning. It found one real spelling mistake (I'd written "fullsome".) and suggested "post-modern" should be hyphenated, though Selzer had not. CG objected to the noun and verb discordance in "Rules is rules". It found eleven sentences longer than thirty one words, its default for academic writing (I could change that). In this piece I left them all. It suggested one 'run-on sentence'; this sentence had worried me, but I left it — can you spot it? CG suggested "in general" was a weak phrase, and I took it out. It picked up several strings of prepositions, but I thought they were clear and necessary so overruled it. CG missed a phrase that read "like. . . like. . . " but I decided the first time was identifying examples while the second did indicate similarity so made a slight change.

Correct Grammar is a useful adjunct to straight re-reading your text. It's reasonably cheap. One irritation is that it is not TEX-aware. I've suggested several times that the TEX community lobby for this to be added. As *thinking* writers, we are probably the intended market for such a product.

XIII Forthcoming meeting – Portable Electronic documents (LA)TEX, Acrobat, SGML)

This meeting, in association with the BCS electronic publishing specialist group will be held at the Bridewell Theatre (just off Fleet Street), London, on January 19th. Speakers will include the charismatic David Brailsford, the eminent David Barron and our own Jonathan Fine. Programme organizers are Jonathan Fine & Malcolm Clark; the local organizer is Carol Hewlett. We hope to be able to incorporate a visit to St Bride's Printing Library (and other sites of local interest). Costs should be in the region of £35–45 for members. Current details can be found on the WWW page

`http://www.warwick.ac.uk/~cudax/meeting.html`

Full details and a booking form will be circulated with *Baskerville* 4.6.

Text Encoding Initiative publishes guidelines

The Text Encoding Initiative (TEI) has published its 'Guidelines for Electronic Text Encoding and Interchange.' This report is the product of several years' work by over a hundred experts in fields ranging from computational linguistics to Ancient Greek literature. The Guidelines define a format in which electronic text materials can be stored on, or transmitted between, any kind of computer from a personal microcomputer to a university mainframe. The format is independent of the proprietary formats used by commercial software packages.

The TEI came into being as the result of the proliferation of mostly incompatible encoding formats, which was hampering cooperation and reuse of data amongst researchers and teachers. Creating good electronic texts is an expensive and time-consuming business. The object of the TEI was to ensure that such texts, once created, could continue to be useful even after the systems on which they were created had become obsolete. This requirement is a particularly important one in today's rapidly evolving computer industry.

To make them 'future-proof', the TEI Guidelines use an international standard for text encoding known as SGML, the Standard Generalized Markup Language. SGML was originally developed by the publishing industry as a way of reducing the costs of typesetting and reuse of electronic manuscripts but has since become widely used by software developers, publishers, and government agencies. It is one of the enabling technologies which will help the new Digital Libraries take shape.

The TEI Guidelines go beyond many other SGML applications currently in use. Because they aim to serve the needs of researchers as well as teachers and students, they have a particularly ambitious set of goals. They must be both easily extensible and easily simplified. And their aim is to specify methods capable of dealing with all kinds of texts, in all languages and writing systems, from any period in history.

Consequently, the TEI Guidelines provide recommendations not only for the encoding of prose texts, but also for verse, drama and other performance texts, transcripts of spoken material for linguistic research, dictionaries, and terminological data banks.

The Guidelines provide detailed specifications for the documentation of electronic materials, their sources, and their encoding. These specifications will enable future librarians to catalogue electronic texts as efficiently and reliably as they currently catalogue printed texts.

The TEI Guidelines also provide optional facilities which can be added to the set of basic recommendations. These include methods for encoding hypertext links, transcribing primary sources (especially manuscripts), representing text-critical apparatus, analyzing names and dates, representing figures, formulae, tables, and graphics, and categorizing of texts for corpus-linguistic study. The Guidelines also define methods of providing linguistic, literary, or historical analysis and commentary on a text and documenting areas of uncertainty or ambiguity.

The TEI Guidelines have been prepared over a six-year period with grant support from the U.S. National Endowment for the Humanities, Directorate General XIII of the Commission of the European Union, the Andrew W. Mellon Foundation, and the Social Science and Humanities Research Council of Canada. The effort is largely the product of the volunteer work of over a hundred researchers who donated time to share their experience in using computers and to work out the specific recommendations in the Guidelines.

The project is sponsored by three professional societies active in the area of computer applications to text-based research: the Association for Computers and the Humanities, the Association for Literary and Linguistic Computing, and the Association for Computational Linguistics.