(2)

ISTITUTO NAZIONALE DI FISICA NUCLEARE

Centro Nazionale Analisi Fotogrammi

INFN/TC-92/05
17 Marzo 1992

M.L. Luvisetto, E. Ugolini:

LATEX: EASY RECIPES

Servizio Documentazione
dei Laboratori Nazionali di Frascati

INFN – Istituto Nazionale di Fisica Nucleare
Centro Nazionale Analisi Fotogrammi

INFN/TC–92/05
17 Marzo 1992

LaTeX: Easy Recipes

M.L. Luvisetto, E. Ugolini
INFN – Sezione di Bologna

Abstract

In this paper we give basic commands to enable an easy conversion from TeX to LaTeX to users that have a good knowledge of TeX typesetting for technical and scientific articles. Basic concepts are described. Clear and easy ways to typeset the most common items are described. Mathematics and tabular typesetting are described in deeper details.

1 Introduction

It happens frequently that users with a good knowledge of TeX feel the need for a faster method to typeset articles, expecially on technical or scientific subjects requiring tables and other complex features. Thus the need of LaTeX is felt, but its usage is endlessly postponed due to the lack of time to learn new commands and concepts. The present article is devoted to such users and implies at least a basic knowledge of TeX.

With TeX in mind, we will give the guidelines to produce the same (or better) output using LaTeX. The described commands include paper setup (title, authors, page numbers, etc.), page setup (section, subsections), font selection, mathematics, tabular information (item, subitem, tables, etc.), index and bibliographic reference.

A full reference of LaTeX commands is found in [1] to be consulted together with [2]. The commands described here are the most frequently needed ones and are derived from the authors' own experience in typesetting technical and scientific articles. [3]

2 What is LaTeX?

TeX is a procedural language in which all actions are described at a very low level. LaTeX is an hybrid computer language that mixes the low level commands of TeX with higher level statements to describe the document as a logical unit made of structured elements. Thus the outer document unit is the report, book or article, made of chapters, sections, etc. that in turn are made of paraghaphs, sentences, words, etc.

Most of the structures are defined by `\begin` statements and ended by the corresponding `\end` statement. Each unit is enclosed in a structure, the whole document is delimited

INFN – Istituto Nazionale di Fisica Nucleare
Centro Nazionale Analisi Fotogrammi

INFN/TC–92/05
17 Marzo 1992

LaTeX: Easy Recipes

M.L. Luvisetto, E. Ugolini
INFN – Sezione di Bologna

Abstract

In this paper we give basic commands to enable an easy conversion from TeX to LaTeX to users that have a good knowledge of TeX typesetting for technical and scientific articles. Basic concepts are described. Clear and easy ways to typeset the most common items are described. Mathematics and tabular typesetting are described in deeper details.

1 Introduction

It happens frequently that users with a good knowledge of TeX feel the need for a faster method to typeset articles, expecially on technical or scientific subjects requiring tables and other complex features. Thus the need of LaTeX is felt, but its usage is endlessly postponed due to the lack of time to learn new commands and concepts. The present article is devoted to such users and implies at least a basic knowledge of TeX.

With TeX in mind, we will give the guidelines to produce the same (or better) output using LaTeX. The described commands include paper setup (title, authors, page numbers, etc.), page setup (section, subsections), font selection, mathematics, tabular information (item, subitem, tables, etc.), index and bibliographic reference.

A full reference of LaTeX commands is found in [1] to be consulted together with [2]. The commands described here are the most frequently needed ones and are derived from the authors' own experience in typesetting technical and scientific articles. [3]

2 What is LaTeX?

TeX is a procedural language in which all actions are described at a very low level. LaTeX is an hybrid computer language that mixes the low level commands of TeX with higher level statements to describe the document as a logical unit made of structured elements. Thus the outer document unit is the report, book or article, made of chapters, sections, etc. that in turn are made of paraghaphs, sentences, words, etc.

Most of the structures are defined by \begin statements and ended by the corresponding \end statement. Each unit is enclosed in a structure, the whole document is delimited

by \begin{document} and \end{document}, where begin and end have the same function as {...} in TEX. The structure names help in keeping track of each structure and avoid errors when commands are nested.

Structures are also defined by single verbs with special meaning, i.e. \section to define a new section, in this case the logical element is defined by one statement only, not by the usual \begin-\end pair to delimit the structure, that is ended by another delimiter, i.e another \section or \end{document}, etc.

Statements enclosed in \begin-\end are quote, itemize, enumerate, verbatim, etc. Single name commands (similar to TEX macros) are \section, \subsection, \chapter, etc.

The LATEX predefined environment enhance TEX capabilities and offer the user a broader and simpler formatting tool.

3 Document Layout

Each document is made at least of a style definition command (preamble), the document structure and some text (the document body). Supposing that the most frequent kind of document you write is an "article" made of an abstract, a few sections, reference information and a table of contents, and that you want reference and contents at the end of your document, the structure will look like the following:

```
% preamble section
\documentstyle[12pt]{article}  % add other options in [...]
%
% add size and paging options here, if any
%
\title{Any Title}
\author{Any N. Author}
%
\begin{document}     % document structure init
\maketitle           % produce title from definition
%
 \begin{abstract}
   ... abstract text ...
 \end{abstract}
%
 \section{First}
  ... section text ...
%
...
...
%
  \section{Last}
  ... section text ...
%
```

```
 \begin{thebibliography}
  \bibitem{bib:one} ... bib text ...
  ...
  \bibitem{bib:end} ... bib text ...
 \end{thebibliography}
%
 \tableofcontents
%
\end{document}
```

The *preamble* can contain only declarations such as document style, page physical dimensions, title, etc. The style declaration syntax is:

`\documentstyle[`*options*`]{`*style*`}`

The standard styles are `article`, `report` and `book`, any other styles can be locally developped and stored in appropriate `sty` files like `local.sty` and called as arguments of the `documentstyle` command. Style additions can also be included calling the style file in the option part of the command.

Other options include two columns output, two sided documents, font size, etc. as shown below:

`\documentstyle [12pt,twocolumn,twoside,addsty,mathsty]{article}`

where 12pt font is selected for a two-column, two-sided article with style change files loaded from the files `addsty.sty` and `mathsty.sty`.

Style selection acts on page and section numbering, page layout and similar and will be choosen according to the author's need and experience. The `article` style is forseen for reports, while `report` and `book` are foreseen for longer documents and books, and are very similar to each other, therefore what follows refers either to `article` or `book`.

The most evident differences between the two styles is the way in which are handled titles (on a new page for `book`), page numbers (on page bottom for `article`) and section numbers (related to chapter for `book`). The "book" actions can be emulated for "article" by specifying `titlepage` in the style option to have title and abstract on a separate page, by specifying `\pagestyle{headings}` in the preamble to have page numbered on top with heading information as in the present document.

LaTeX changes font size for titles and authors, automatically centers both title and authors, adds document date, either user specified or the current date. If the user needs a different formatting for the title page, the structure:

`\begin{titlepage}` *text* `\end{titlepage}`

leaves an `empty` page for which the author is completely responsible.

If the user wants to alter LaTeX default page size, the following commands in the preamble show some of the parameters that can be changed.

```
\textwidth 16.0cm              % set width, height, margin
\textheight 23.0cm
\oddsidemargin -0.25cm
\topmargin -1.5cm
\evensidemargin 0pt
```

Once defined the document style, the sectioning commands are:

```
\part       \chapter                                    (for book only)
\section    \subsection  \subsubsection
\paragraph  \subparagraph
\appendix
```

In each of the above described structures, positioning, numbering and fonts are fully handled by LaTeX according to the style in use.

4 Input File – Syntax

LaTeX user input file has extension `.tex` such as `demo.tex`. Besides the `.sty` files mentioned in Section 3, LaTeX writes temporary files for index and reference information: `toc` for table of contents, `lof` for list of figures, `aux` for references, etc. To prevent temporary files, insert `\nofiles` in the preamble.

The input format is similar to TeX, with \ as escape character and the same special characters (i.e. # $ % & ~ _ ^ \ { }). Basic TeX macros are common to LaTeX, therefore to type $, just type \$.

Structures (i.e. `\begin-\end` and braces (i.e. {...}) must be balanced. Furthermore control sequences can have optional parameters that are enclosed in square brackets. One example is the sequence that defines the document layout:

```
\documentstyle [twocolumn,twoside,12pt]{article}
```

The options must be specified immediately after `\documentstyle`, multiple options are separated by commas, no blank is allowed inside the square brackets.

Paragraphs are generated by one or more blank lines. In general a LaTeX source code is more readable than TeX equivalent.

Syntax errors produce error messages that are in general produced by TeX, as LaTeX is built on top of TeX and can be either obvious or difficult to understand in the same way as TeX ones.

As in TeX, the document can be splitted into more input files, that are included unconditionally using `\input{`*file-name*`}` or conditionally using `\include{`*file-name*`}` to include **only** the files named in the preamble through the command:

`\includeonly{`*file-1,file-2,file-3...*`}`

If the preamble does not contain an `\includeonly` command all files are included. If `includeonly` has an empty argument list, no file is included.

5 Fonts

Font selection in LaTeX is almost the same as in TeX. The default active font is *roman*, the default inactive font is *italic*. Font size is selected at document level, the default is 10 pt.

Furthermore LaTeX provides an useful tool to emphasize text elements, the \em environment. The \em command switches the default font from the active one to the inactive, thus if the current font is **roman**, the emphasized text is printed in *italic*, and viceversa. For entire sentences or paragraphs, the emphasized mode is declared as a structure, i.e. \begin{em} ... \end{em}. As a consequence of the switching feature, in a long emphasized text typeset in italic, a shorter fragment can be emphasized in roman, as in the following example:

A long emphasized text can include emphasized strings *written in* roman, *inside an italic sentence.*

The above fragment is produced by the following source code:

```
\begin{em} A long emphasized text can include {\em emphasized
strings} written in {\em roman}, inside an italic sentence.\end{em}
```

Other predefined fonts are:

bf	bold	**Bold font**
sf	sans serif	Sans Serif font
sl	slanted	*Slanted font*
sc	small caps	SMALL CAPS FONT
tt	type writer	Typewriter font

Fonts are declared as in TeX (i.e. {\bf *text* }). Also accents and symbols are typeset as in TeX, thus to produce à, just type \'{a}. Examples of other symbols are below:

Œ	\OE
‡	\ddag
£	\pounds

Other fonts are defined for mathematical use, like greek letters (limited set), calligraphic ones (only uppercase), plus the mathematic italic (\mit) that is the default type for maths and mathematical bold (\boldmath) fonts that will be described in deeper details in Section 9. Such fonts can be used only in math mode or inside a math box (\mbox). The font style is shown below:

$\alpha\,\beta\,\gamma$	\alpha;\beta\;\gamma
$\mathcal{ABFH}$	{\cal A B F H}
$\boldsymbol{a + x\pi - \rho}$	{\mbox{\boldmath $a + x\pi - \rho$}

Check the difference between *math italic* and *mathitalic*, the first one is produced by the {\em} environment and the second one by \mit font, automatically used in math mode.

In technical manuals, expecially when reporting computer programs, it is required not only to use a non-proportional font as \tt, but also to reproduce the text as it stands, eventually producing a mark such as ␣ for *required* spaces. For this purpose LaTeX has four commands:

`[\begin{verbatim} ... \end{verbatim}]`,
`[\begin{verbatim*} ... \end{verbatim*}]`,
`\verb, \verb*`

The `verbatim` environment typesets the text on a new line, thus it is used for long insertions, the `\verb` command is used for short strings inside the current line, the text is delimited by any pair of identical characters such as `!` as in the following example:

To display blanks in computer programs type `\verb*!int l,k;!`, *the typeset result will be* `int␣l,k;`

Font size can be changed with commands as `\tiny` or `\small` for smaller fonts, or `\Large` or`\Huge` for bigger fonts, followed by the font definition if different from roman, so we have `\large\bf` for large bold letters. Examples of the `\tiny`, `\Huge` and `\large\bf` follow:

Font Font **Font**

Users can define other fonts not provided in the default set, as in TEX, with the command:

`\newfont{\logof}{logo10 scaled\magsetp1}`

where `\logof` is the new font name that is available in the `logo10` font description file in an enlarged size. To typeset some text in the new font, just call it as any other font: `{\logof \symbol{26}}` to write the symbol with character code 26.

6 Notes

LATEX provides two types of notes: *footnotes* and *marginal notes*. The syntax for footnotes is similar but not identical to TEX. In TEX footnote numbering is required and number generation is not handled, thus the user must take care of footnotes numbering. Both number and text must be enclosed in braces, not required when the number is one character only.

In LATEX the numbering is a positive integer automatic stepped for the next footnote command. The numbering can be changed at user's will as an option. The syntax is:

`\footnote [`*num*`]{`*text*`}`

where *num* is the optional number for the following footnote text.

Furthermore LATEX has the commands:

`\footnotemark[`*num*`]`

`\footnotetext[`*num*`]{`*text*`}`

to handle special cases in which the footnote calling sequence is needed inside a box.

Marginal notes are not numbered and are placed in the free paper margin with the first line even with the line of text in which the note is inserted, as happens here. The note is placed according to the style in use: it is placed on the right for one-sided documents, on the outside margin for two-sided printing, in the nearest margin for multicolumn style. *note*

The margin note is produced by `\marginpar` and different text can be produced for left and right margin as an option. The syntax is:

`\marginpar[`*left_text*`]{`*right_text*`}`

The note position can be changed using new setup declarations that alter the default style:

`\reversemarginpar` places the note in the opposite margin from the default.

`\normalmarginpar` resets the note position to the default.

7 Item Lists

LaTeX has extensive capabilities to handle tabular information, both in the form of tables (see next Section) and in the form of aligned text.

LaTeX defines a set of structures to format lists of items or quotations. The structures are: `quote`, `quotation`, `itemize`, `enumerate`, `description`. Inside the structure, each entry is declared through the `\item` command.

The `quote` environment is used for short quotations, the `quotation` environment for longer ones. The "quoted" text is shifted on the right, as shown here.

> LaTeX is able to handle quotation.
>
> Each quotation is shifted on the right.

The above example is produced by:

```
\begin{quote}
\LaTeX{} is able to handle quotation.

Each quotation is shifted on the right.
\end{quote}
```

More useful in technical documents are the `itemize` and `enumerate` environments described in the following example.

- Each item in the list is marked by a *bullet.*
- Item list can be nested.
  1. Items in enumerated lists are labelled by numerals.
  2. Lists can contain two or more items.
  3. If there is only one item, there is logically no list.
- Blank lines are ignored.
- In the input file, indent lines to show the item list structure.

Item lists can be nested in complex manners, as shown by the above example produced by the following code:

```
\begin{itemize}
 \item Each item in the list is marked by a {\em bullet}.
 \item Item list can be nested.
  \begin{enumerate}
   \item Items in enumerated lists are labelled by numerals.
   \item Lists can contain two or more items.
   \item If there is only one item, there is logically no list.
  \end{enumerate}
 \item Blank lines are ignored.
 \item In the input file, indent lines to show the item list structure.
\end{itemize}
```

Another very useful feature to format item lists is the `declaration` environment. In this environment an item has a name, that is typeset in boldface and is followed by its description, that is typeset as itemized text.

This is an example of the `description` environment:

X nX delete one or 'n' characters starting with the character immediately preceding the current cursor position.

dnG delete all lines starting with the current line up to line 'n'. Action is in the forward or backward direction depending on the position of line 'n' with respect to the current line.

produced by the commands:

```
\begin{description}
 \item[X nX] delete one or 'n' characters ...
 \item[dnG]  delete all lines starting with ...
\end{description}
```

8 Tables

In TEX tables are handled by the "settabs" commands, that can either create fixed width columns or use a template to describe the table fields. LATEX has two environments for tables: the `\tabbing` and the `tabular` environments. The first one emulates TEX commands with template description, while the second one enables the user to create very complex tables in an easy way. Furthermore LATEX code for tables is much more readable than TEX one.

The `tabbing` environment handles tables of any length that span across pages. The tab stops can be set either in a prototype or as columns are typed. The tabbing information is a structure started by `\begin{tabbing}` and ended by `\end{tabbing}`. The command `\=` sets the tab stop, `\>` moves to the next stop. The element of the first column has no tab information, the line is ended by `\\`. If the line represents a prototype, it ends with `\kill`.

As a tabbing example consider the following table describing LATEX fonts:

bf	bold	**Bold font**
sf	sans serif	Sans Serif font
sl	slanted	*Slanted font*
sc	small caps	SMALL CAPS FONT
tt	type writer	`Typewriter font`

This table makes use of a prototype line and is generated by the following code:

```
\begin{tabbing}
font \= small caps font \=        \kill
\verb!bf!  \> bold        \> {\bf Bold font} \\
\verb!sf!  \> sans serif  \> {\sf Sans Serif font} \\
\verb!sl!  \> slanted     \> {\sl Slanted font} \\
\verb!sc!  \> small caps  \> {\sc Small caps font} \\
\verb!tt!  \> type writer \> {\tt Typewriter font} \\
\end{tabbing}
```

The tabular enviroment creates tables that are essentially boxes, that behave like figures and can float around the page but cannot span across pages. Frequently these tables are enclosed in drawn rectangles containing vertical and horizontal lines to separate the columns. The tab stops are handled automatically and defined by &, the position of the items in the column is defined within the tabular enviroment by one of the characters `l r c` to respectively align on the left, on the right or center the item, the line is ended by \\.

A vertical line is drawn with | declared in the tabular specification, the command \hline after \\ draws a horizontal line across the full width of the table. The command \cline{$i-j$} draws a horizontal line across columns i through j, inclusive.

When an argument spans multiple columns, it is produced by the \multicolumn command with the following syntax:

\multicolumn{n}{pos}{$item$}

where n is the number of columns to be spanned, pos defines the position: `l` (for left), `r` (for right), `c` (for centre), $item$ is the text to be typeset.

As an example consider the following table:

Cray Total Gain (millisec)				
Level	Time	Speed-up	Gain	CDC/Cray
0	33.37	–	–	1.36
1	29.86	10.5	10.5	1.52
2	24.19	27.5	19.0	1.87
3	21.92	34.3	9.4	2.07

that was produced by the following code:

```
\bigskip
\begin{tabular}{|c|c|c|c|c|} \hline
\multicolumn{5}{|c|}{Cray Total Gain (millisec)}\\ \hline
```

```
Level & Time   & Speed-up  & Gain & CDC/Cray \\ \hline
0     & 33.37  & --         &  --   & 1.36 \\ \hline
1     & 29.86  & 10.5       & 10.5  & 1.52 \\ \hline
2     & 24.19  & 27.5       & 19.0  & 1.87 \\ \hline
3     & 21.92  & 34.3       &  9.4  & 2.07 \\ \hline
\end{tabular}
\bigskip
```

To have an idea of the easy tabular environment provided by LaTeX, compare the above code with the following TeX code used to produce the same table:

```
% TeX code for table
\relax\centerline{    %\relax\hskip 80pt
\vbox{\tabskip=0pt \offinterlineskip
\def\tablerule{\noalign{\hrule}}
\halign to260pt{\strut#& \vrule#\tabskip=1em plus2em&
  \hfil#& \vrule#& \hfil#\hfil& \vrule#&
  \hfil#\hfil& \vrule#&
  \hfil#\hfil& \vrule#&
  \hfil#\hfil& \vrule#\tabskip=0pt\cr
\noalign{\medskip}\hfil\cr\tablerule
&&\multispan9\hfil Cray Total Gain (millisec)\hfil&\cr\tablerule
&&\omit\hidewidth Level\hidewidth&&
  \omit\hidewidth Time\hidewidth&&
  \omit\hidewidth Speed--up\hidewidth&&
  \omit\hidewidth Gain\hidewidth&&
   \omit\hidewidth CDC/Cray\hidewidth&\cr\tablerule
&&0&&33.37&& -- && -- &&1.36&\cr\tablerule
&&1&&29.86&&10.5&&10.5&&1.52&\cr\tablerule
&&2&&24.19&&27.5&&19.0&&1.87&\cr\tablerule
&&3&&21.92&&34.3&& 9.4&&2.07&\cr\tablerule
\noalign{\medskip}\hfil\cr}}}\par
```

9 Mathematics

Mathematical formulas can appear as *in-text* elements (`math` environment) or as displayed formulas (`displaymath` enviroment). Numbered displayed formulas are produced in the `equation` enviroment. The commands to select the environments are:

in-text maths:	`$...$` or `\(...\)` or `\begin{math} ... \end{math}`
displayed maths:	`\[...\]` or `\begin{displaymath} ... \end{displaymath}`
equation:	`\begin{equation} ... \end{equation}`

Most math elements and symbols are made as in TeX. The unchanged items are: subscripts and superscripts, greek and calligraphic letters, math spacing, symbols, ellipsis,

etc. Most math commands are identical to TEX, such as `\overline` and `\underline`, `\vec`, etc. The same applies to font selection for math, text and scripts.

Fractions are handled in an easy way by the `\frac` command that has two arguments: numerator and denominator.

$$x = \frac{y+z}{y^2+z^2} \qquad \texttt{\textbackslash[x = \textbackslash frac\{y+z\}\{y\^{}\{2\}+z\^{}\{2\}\} \textbackslash]}$$

$$y = \frac{a+b}{1+\frac{a}{a^2+b^2}} \qquad \texttt{\textbackslash[\textbackslash frac\{a+b\}\{1+\textbackslash frac\{a\}\{a\^{}2+b\^{}2\}\}\textbackslash]}$$

Arrays are produced with the array environment, that is simular to the tabular one. The declration specifies the size (number of columns) and the alignement of items: `l r c` for left, right or center. New items are set with `&`, rows are ended with `\\`

$$A = \begin{pmatrix} x-1 & 1 & 0 \\ 0 & x-1 & 1 \\ 0 & 0 & x-1 \end{pmatrix}$$

The above array is typeset with the following commands. Note that array syntax in LATEX is very similar to TEX.

```
\[ A = \left( \begin{array}{ccc}
x-1 &1    &0   \\
0   &x-1  &1   \\
0   &0    &x-1
\end{array} \right) \]
```

The delimiters for arrays are typeset in the same way as TEX: the commands `\left` or `\right` to specify the left or right delimiter followed by the delimiter itself `() [] | {}`, the `\left` and `\right` commands must come in matching pairs, but the delimiters do not need to match in any form.

Long or multiple formulas are displayed with the `eqnarray` environment, that enables line numbering and equation splitting, rows are separated by `\\`, items by `&`, numbering is disabled by `\nonumber`. The following example shows the use of `eqnarray`.

$$\begin{aligned} x &= a + y - c^2 && (1)\\ y &= a^2 + b^2 - x - \\ &\quad xy + c^3 - p && (2) \end{aligned}$$

```
\begin{eqnarray}
x & = & a + y - c^2 \\
y & = & a^2 + b^2 - x -\nonumber \\
  &   & xy + c^3 -p
\end{eqnarray}
```

When symbols must be typeset one above another, use the command `\stackrel`:

$A \stackrel{a'}{\rightarrow} B$ `$ A \stackrel{a'} {\rightarrow} B $`

As already mentioned in the font section, math mode uses *math italic*. There is also a boldface declaration (`\boldmath`) that applays to letters and symbols, but not to subscripts and superscripts. It cannot be used in math mode, so it must be used in a `\mbox` command, i.e. `\mbox {\boldmath $` *formula* `$}`.

Theorems can receive a name, a label and a number when defined by the `\newtheorem` command that takes two arguments: the name and the label. The theorem text is emphasized. The numbering is auto-generated and can be computed within the specified sectional unit using the optional argument. The sectional unit can be a LaTeX predefined as `chapter`, `section`, etc. or the name of a user defined theorem, so that all theorems of the same type are numbered in the same sequence.

```
\newtheorem{guess}{Conjecture}[section]  % define conjecture in section
....
\begin{guess} This is a guess. \end{guess}
```

Conjecture 9.1 *This is a guess.*

10 Definitions

LaTeX provides tools to define new commands (TeX macros). The new commands are defined by `\newcommand`, followed by the name, the *optional* arguments and finally the definition. The name **must** be prefixed by `\`, when used the arguments must be enclosed in braces. In the following example, is shown LaTeX syntax to define and call the macro `\abx` and the typeset formula thus generated.

`\newcommand{\abx}[2]{$#1x+#2$} \abx{5a}{b}` $5ax + b$

In a similar way to change style, fonts, emphasys, etc. the user can define a new environment with `\newenvironment`.

Commands can be defined anywhere, but the definitions must appear before their use.

11 Graphics – Floating Objects

LaTeX has a limited capacity of creating graphic objects and a related capacity to place figures in a *floating* way to avoid splitting between pages. Figures can receive a caption, in this case also caption number is produced. Suppose that you must insert a figure 5cm tall in your text, the LaTeX code is:

```
\begin{figure}
  \vspace{5cm}  % leave space for figure
  \caption{Fractal image.}
\end{figure}
```

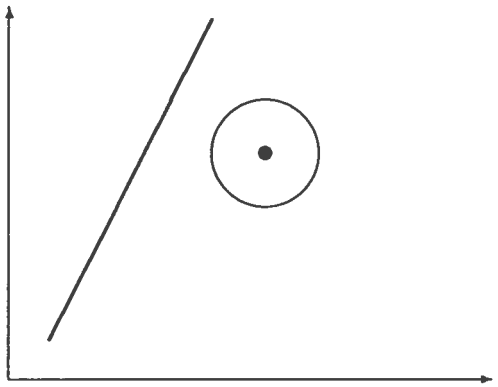

Figure 1: Graphic example

and the produced caption is *Figure 8: Fractal image* centered in the page (for an example check Figure 1)

In the above example we leave blank space to insert the picture at a later time with cut and paste methods. For simple graphics LaTeX is able to draw axes, lines, circles. The coordinate system is expressed in `\unitlength` with default value 1 point (nearly 1/72 inch).

The picture is created with the `picture` environment that specifies picture $x - y$ dimensions and, optionally, origin enclosed in parenthesis:

`\begin{picture}(200,150)(20,10).`

Any graphic and/or text information is positioned in the picture with the `\put` command that is followed by the coordinates in parenthesis and by the object to put in braces: `\put(0,-10){Origin}`. Objects are ordinary text, straight lines, arrows, circles, ovals.

The line and arrow syntax is the same, the command ares `\line` and `\vector`. The command arguments are slope expressed as $(\Delta x, \Delta y)$ and length. For `\circle` the argument is the diameter. For `oval` the arguments are width and height plus an optional argument to draw only half or a quarter of the complete oval. Lines can have two standard thicknesses for the lines: `\thinelines` (default) and `\thicklines`. Both declarations are used in the example.

The source code example of a simple drawing follows. LaTeX output is shown in Figure 1. Note that LaTeX makes use of brackets () when describing graphic objects.

```
\begin{figure}
 \begin{picture}(200,150)
  \put(0,-10){Origin}
  \put(5,5){\vector(1,0){180}}          % x-axis
  \put(5,5){\vector(0,1){140}}          % y-axis
  \thicklines
  \put(20,20){\line(1,2){60}}         % draw line
  \thinlines
  \put(100,90){\circle{40}}             % draw circle
  \put(100,90){\circle*{5}}             % draw filled center
 \end{picture}
```

```
 \caption{Graphic example}\label{fig:ex}
\end{figure}
```

With the same basic criteria, virtual boxes can be created to split the physical page into sub-areas called *boxes*. Text can be placed inside the boxes at center (default), left (`l`) or right (`r`). There are two commands to handle such boxes: `\makebox` and `\framebox`, the second one puts a frame around the box. Both commands have the same syntax i.e. two optional arguments for width and postion, and the text to be framed: `\framebox[`*width*`][`*pos*`]{`*text*`}`. Both commands can define box size and text position as optional arguments. The text to be typeset is enclosed in braces. The syntax and typeset results are shown in the following example:

framebox facility
framebox facility

```
\framebox[4cm]{framebox facility}
\framebox[4cm][l]{framebox facility}
```

Besides the above commands, the `minipage` environment enables the user to split the typeset information in variable size paraghaphs typed as multicolumns side by side inside the current environment. The `minipage` environment is used in the above example with the following commands, in which boxes are created in a nested way. The first `minipage` is 6.5cm wide and the second one is wider (8.0cm). Note that the two `minipage` environments are typeset side by side as normal text with a `\quad` horizontal space.

```
\vspace{10pt}
\noindent
\begin{minipage}[t]{6.5cm}
 \framebox[4cm]{framebox facility}
 \vspace{5pt}
 \framebox[4cm][l]{framebox facility}
\end{minipage}
\quad                          % align second minibox
\begin{minipage}[t]{8.0cm}
... insert same commands as above ...
... in verbatim mode ...
\end{minipage}
```

Text can be positioned in the center, left or right of the page using the `center`, `flushleft` or `flushright` environments. To start new lines in such environments use `\\` as in tabular and array structures.

12 Useful Commands

Besides the environment definitions and the already described commands, LaTeX offers many other useful commands. Also many plain TeX commands can be used in LaTeX without conflict, as `\qquad`, `\quad`,`\noindent`, etc.

TeX page layout commands should be avoided, not to alter the internal setting of the document environment. Also personalized output routines can cause problems. Only

practice and tests can tell which TEX commands can cause trouble in LATEX. In the usual technical and scientific documents, LATEXshould not pose any particular problems.

Some useful commands are listed below.

`\today`	current date
`\ldots`	ellipsis (...)
`\parindent`	width of paragraph indentation
`\parskip`	vertical space between paragraphs
`\baselineskip`	inter-line vertical space
`\hspace{3cm}`	skip 3cm in horizontal direction
`\vspace{5cm}`	skip 5cm in vertical direction
`\hyphenation`	customize hyphenation pattern
`\typeout`	prints message on terminal and log file
`\newpage`	start a new page
`\noindent`	avoid paragraph indentation

13 Reference – Index – Bibliography

LATEX provides an easy way to create cross-references linking the various elements of the document, such as figures, equations, sections. Each element can receive a name through the `\label` command. Any string can be assigned to the name, suggested naming conventions are `eq:euler`, `seq:syntax` and the like to create mnemonic names related to structures and thus more easily identifiable.

Once named the element, it is referenced with the `\ref` command. The name can be defined in any place in the source code (before or after being referenced), but it should be typed immediatly after the referred item, i.e. if the user wants to label a caption to refer it by figure number, the `label` statement must be typed after the caption title:

```
\caption{Graphicexample}\label{fig:ex}.
```

LATEX writes temporary files (see Section 4) to handle references that are resolved on the next run, thus it **must** be run twice to typeset the updated reference information. If the temporary files are missing or possibly not up to date, a warning message is written. As an example consider the following fragment:

Equation 3 is very famous.
Energy equation is

$$E = mc^2 \tag{3}$$

produced by:

```
Equation \ref{eq:ck} is very famous.
Energy equation is \begin{equation} E = mc^2 \label{eq:ck} \end{equation}
```

References can be set also on any page using `\label` to name the text and the command `\pageref` to get the page number of the named text, the source code looks like the following:

```
see page \pageref{fonts} for more details.
...
Predefined \label{fonts} fonts are:
```

Keep label definitions to a reasonably short size to avoid LaTeX problems with internal space. Avoid defining labels that are never used or used too seldom, keep a list of used labels and their meaning to produce a readable and mantainable input file. More than forty label can cause problems.

Finally LaTeX can produce a table of contents and bibliographic reference with auto-labels. The style of this information is, as always, related to the document style. the positioning inside the document is connected to the place in the input file: at the beginning if the command is typed before `\maketitle`, at the end it it is typed before `\end{document}`.

The table of contents is produced by `\tableofcontents`, other index information can be produced for tables and pictures using `\listoffigures` and `\listoftables`.

To produce bibliography, the user defines entries in the `thebibiography` structure, that can have as optional argument the definition of the widest label in the item list. Each item is inserted with `\bibitem` that has the following arguments: an optional label that overides the default numbering scheme, the key-name for citations and the entry text. The items are referenced with the `\cite` command.

The bibliographic data for this documents are generated by the following environment definition:

```
\begin{thebibliography}{99}
\bibitem{bib:la} L. Lamport. \LaTeX: {\em User's Guide and Reference ...
\bibitem{bib:te} D.E.Knuth.{\em The }\TeX {\em book.\/} Addison ...
\bibitem{bib-my}M. L. Luvisetto, E. Ugolini. {\em Introduzione ...
\end{thebibliography}
```

and are called in any place as shown:

```
... for more informations see \cite{bib:la,bib:le} and, in Italian,
\cite{bib-my}.
```

References

[1] L. Lamport. LaTeX: *User's Guide and Reference Manual.* Addison-Wesley, 1986

[2] D. E. Knuth. *The* TeX*book.* Addison-Wesley, 1986

[3] M. L. Luvisetto, E. Ugolini. *Introduzione all'Uso di* TeX: *Linguaggio, Tools, Installazione.* INFN/TC–88/14, 1988

Contents