Contents

1 Allplo

The program Allplo has a long history starting in the 70th on a PDP computer from Digital Equipment. The structure was determined by the 32kByte core memory. The basis of the code was the Benson-library producing the plot commands for Benson plotters. The last one used was a flatbed plotter with 7 pens of different colors.
To that time the program (developed by C.P.Koehler, E. Meissner using fortran66) worked only interactively. The larger core memory of the VAX family made the batch version possible (H. Spiering, M. Alflen). R. Jakobi wrote a Pascal program to manipulate the Benson files and A. Vef started the xben program (written in C), which took the Benson plot files to an X-window screen and translated the plot to an .ps file format.
In the next step Allplo were merged into xben (H. Spiering) and got the name xalpl. Still there is the file .ben, which is deleted at the end of the program. The benson file becomes visible leaving the program by ^c.
In Fig. 1 some features are described by a text written with allplo.

2 Commands inside plot window

- a: execute allplo and plot benson file
  Working with allplo typically means changing the batch file xx.bat and re-plotting to see what the changes look like. The command {a} executes the batch file and re-plots.
- c: if the plot is zoomed, a re plot will restore the original picture. Typing c conserves the zoomed size when executing command a. Again c (toggle) detaches this command. This is seen typing a.
- f: format of the paper - portrait/landscape
  The command toggles between portrait and landscape. Some information (zoom , position of the cursor) is lost.
- g: geometry of the plot
  The command opens a window asking for actions by typing numbers. The focus is directed to that window as long as there is no interrupt from the mouse.
- i: fit to paper format (choice:1 in command g)
  (the position of the cursor is lost.)
- l: load layout file
  Some values determining the actual figure are read from file (see {s})
- m: midway of paper (choice:6 in command g)
  Command number 6 of command {g} there is called "center plot".
- p: print postscript file
  The default fill name is that of the last data file with the extension .ps.
- q: quit the program
- r: restore, right mouse button = r
- s: store layout file
  (see load layout file)
- z: zoom , middle mouse button
  The header of the window asks for the corners of the clip.
- H: more help (for batch file and more)
  This dvi file.

3 Editing text

3.1 Example

The following text of Fig. 1 is written by the text facility of allplo. The batch file textexample.bat calls the text file example.txt and a .dat file fortext.dat, which defines the dimensions by to points (34 0 0 and 34 15 28) stopped at with pen up(34).

The following text is written with allplo.

Allplo has been developed in Mainz on a PDP11 for plotting Mössbauer spectra. The features for writing text were extended several times until allplo became out of use replaced by professional plot programs.

The text editing features of allplo met the requirements to add some information inside the plots of the data (i.e. T=10 K). The simple structure of allplo data files enabled to plot everything what can be realized by vectorgraphic. This way people plotted large headerlines for posters (to that time posters were pasted), flow or elektronic circuit diagrams (therefore the name allplo).

The text editing features of allplo shall be ilustrated by the subsequent text.

The title is written **boldface** started by $n, n the number of repetition of displaced vectors of the letters. The displacement can be visualized by the zoom (see commands inside plotwindow for allplo). Comparison of the title and the word boldface: → commands end with the end of the line or by $blank. The same way @ (= @0) ends the color n. Indices are often used: mms^{-1} and X_n. There are no brackets as in LATEX so that an index of an index is not possible to write. X_n^m uses the $p, $b, $f commands: position, back to position, forward to end of current text.

Text can be boxed ,the full text of the line including preceding blanks. B4 fills with blue and @1 writes white.

The underlined text is written jumping one line back by: −u3; and preceding blanks. Many special characters of the keyboard are available. Others can be typed by LATEX type commands like $\backslash for \ .

A root even of a complicated expression can be constructed by $\sqrt: $\sqrt{\frac{2*5+1}{b*5+2}}$ Not so easy, constructed by trial and error using $h, $H, $\overscore etc.

Figure 1: Text written by allplo

3.2 Global line commands

The text of each line starts with a semicolon, separating the global commands for the line from the text. The global commands of the first line define the start position by the x,y-coordinates of the frame, the width and hight of the letters hx,hy, the rotation angle α and the distance to continuation lines, which start with a "+" or "&" followed by commands of table 1 (capital F makes no sense). For continuation lines bare numbers (x,y,hx,hy,α, and distance) are not expected.
If all six numbers are 0 the values of hx,hy,dist are replaced by their default values. If there are less then 6 numbers the missing numbers are put to zero and treated as before. Two subsequent commas (,,) are replaced by (,0,).

Table 1: The commands significant for the whole text line are listed. They are shown as a reminder in the batch file by the string "x,y,hx,hy,angle,dist,b,c,$,@,u,o,f,F/n;". The command c, which disables the x-value, has no succeeding number (a number at this place is interpreted as one of the 6 numbers).

bn	: box of color n
c	: center x coordinate
$n	: bold multiple n
on	: overline of color n
un	: underline of color n
@n	: color n of signs and characters
fn	: frame of color n of the text line
Fn	: frame of color n including all text lines

The behavior of the command capital F has to be commented. If F appears several times only the last one is active, that means the evaluation of the frame coordinates always starts with the command F and ends with the last line. The frame includes all text lines from the last global command F. Lower case f can be used for each line.

3.3 Local commands

Three groups of editing commands beginning with a \,$, and @. The \-commands selects fonts (figure 2 and 4) and specifies some commands according to LATEX rules.

Table 2: The use of "\" follows the rules of LATEX. Since the "at" symbol @ is used as $ as a command specifier it needs also a \ to be recognized as a letter to be printed.

\, 2/5	: blank
\! -1/5	: blank
\"a \"o \"u	: ä ö ü umlauts
\$	: $
\\	: \
\@	: @

By the $-command position and size of letters are manipulated. The @-command will be typically used to change the color (0,...,9). The possibility to put the whole text to upper or lower case (independent of the typing) may be of lesser usefulness.

Table 3: Four different fonts are available (see figure 2). There are further 22 characters defined (figure 2) which can be easily extended. In addition the full symbol table of figure 4 can be included in the text. The command is case sensitive in order to differentiate between open and filled symbols.

\R	OR	\r	: ROMAN (default)
\I	OR	\i	: ITALIC
\S	OR	\s	: SCRIPT
\G	OR	\g	: GREEK
\Z	OR	\z	: SIGN (ZEICHEN) 1-22
\y			: Symbol 1-33
\Y			: filled Symbol 1-33

Table 4: The "@-commands" allow to switch the whole text to upper or lower case and to change the color of the text (as the global command @n). "@ "-command restores all global settings. @c comes back to the text as it is typed (also executed by "@ ").

@blank			: back to case standard and color of the line
@u	OR	@U	: only capital letters
@l	OR	@L	: only lower case letters
@c	OR	@C	: case standard
@n		n = 0...	: color

Table 5: The "$-commands" allow to manipulate the position (index type), size and multiple drawing of single letters. The three commands $p,b,f were introduced in order to write upper and lower indices at the same x-position. The "$ "-command restores the global command $m of the line, the size of the letters as given by hx,hy, and removes the shift for the indices.

$blank			: back to default values
$^			: upper index
$_			: lower index
$P	OR	$p	: actual position (memorized)
$B	OR	$b	: back to the memorized position
$F	OR	$f	: jump to the actual position of the text
$H		*1.2	: increase of the size of the letter
$h		/1.2	: decrease of the size of the letter
$m		m	: multiple drawing (m fold - bold)

4 Vector fonts

The four vector fonts available are shown in figure 2. The assignment of the Greek

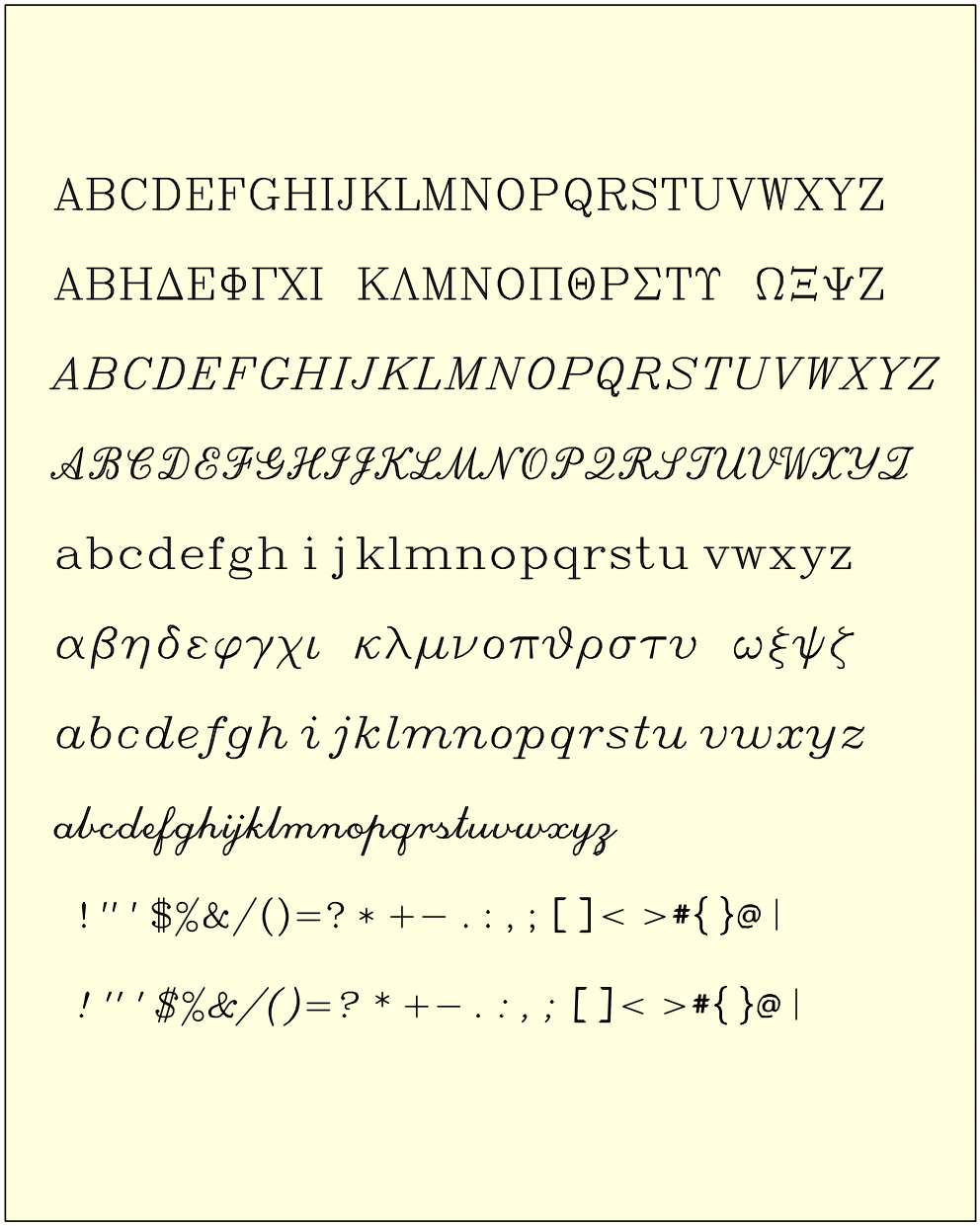

Figure 2: Four different fonts are available: Roman, Greek, Italic, and Scrip. The first 4 lines show the capital letters and the second 4 lines the lower case one. Two styles (Roman,Greek) and (Italic, Script) are for the extra characters shown in row 9 and 10, respectively.

letters to the 26 letters a,...,z is indicated by their positions. The letters J and V cannot be assigned. The extra characters are directly obtained from the keybord except \,$, and @ which act as control characters. They are obtained by a preceding \.
Few extra characters and symbols to be used as characters in the text are listet in figure 3. The first 24 characters are obtained by \z or \Z and the number given below the character (see commands of table 1). \z switches to the character mode as \r to

Table 6: The 43 characters in figure 3 are also available by LATEX command-strings for special characters: $\command-string

Int	Aring	underscore
hat	rightarrow	leftarrow
leftrightarrow	Rightarrow	Leftarrow
Leftrightarrow	vert	uparrow
downarrow	updownarrow	cdot
parallel	neq	sim
perp	approx	hbar
at	backslash	number
section	infty	int
oint	sqrt	overscore
frac	nabla	partial
cap	supset	subset
cup	tilde	propto
geqq	leqq	pm
mp	langle	rangle

the Roman characters (\z1 gives the integral sign, "\z 1" gives a blank and the integral sign, \z22 24 gives @ #, \z22\z24 gives @#).
The 33 symbols, which are are available for the plot routines, are handled in the same way as the characters above. \y switches to open symbols as shown in the figure and \Y to filled symbols, some of them are shown in figure 4 (also demonstrating implemented the colors). Symbols 1-12 cannot be filled. It is not useful to ask for \Y1,...,12 symbols because they are written several times by the filling procedure. The pen width parameter(see table 7) determines the distance between the lines in the filling procedure (the loop number). If the pen width is larger then the pixel diameter, the filling becomes incomplete.

5 Symbols

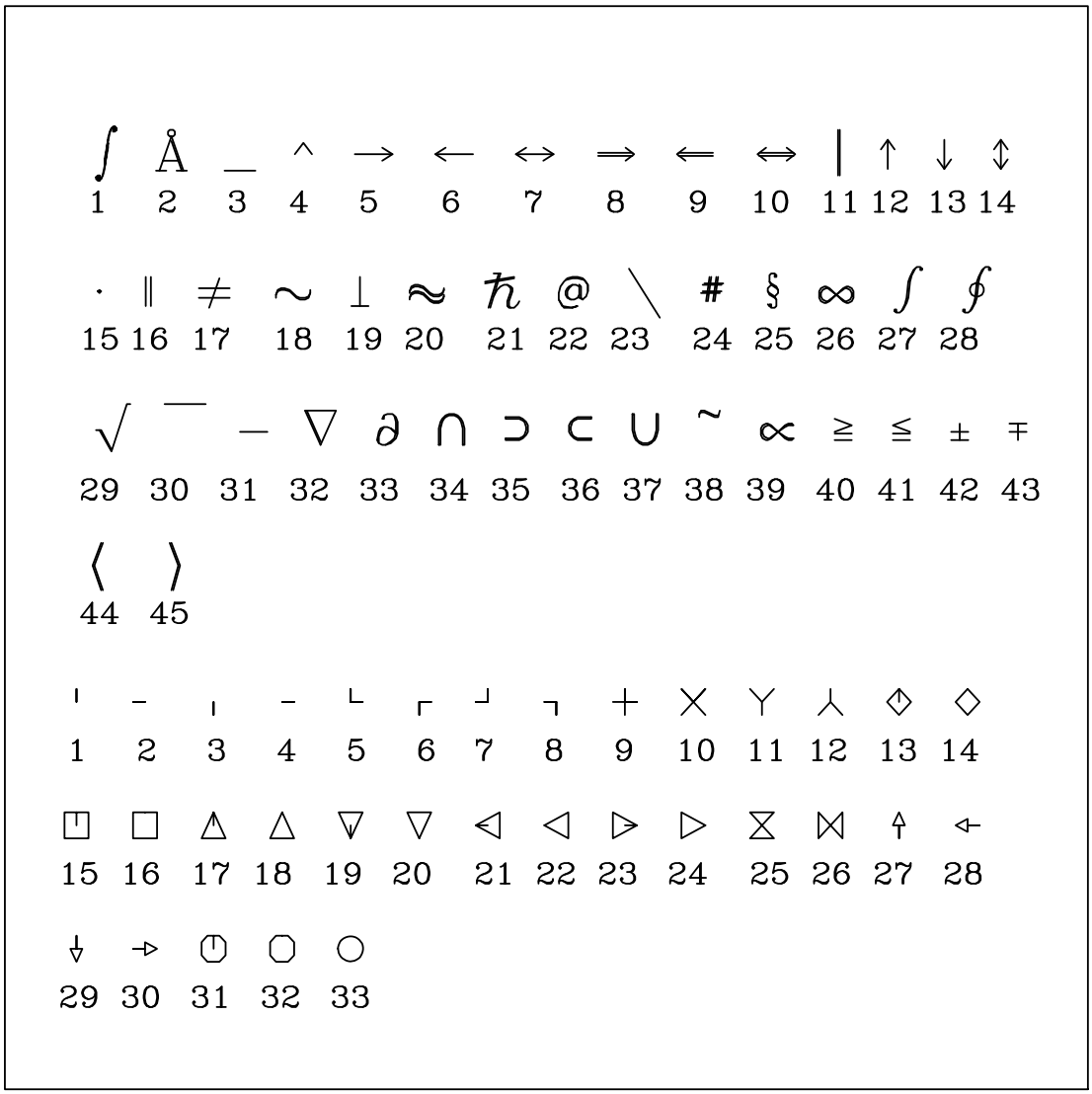

Figure 3: Allplo has 43 special characters which are addressed by \z or \Z and the number given below. Three characters are also obtained directly from the keybord:\ by \\, @, and #. The 33 symbols are addressed by \y and \Y for the filled version.

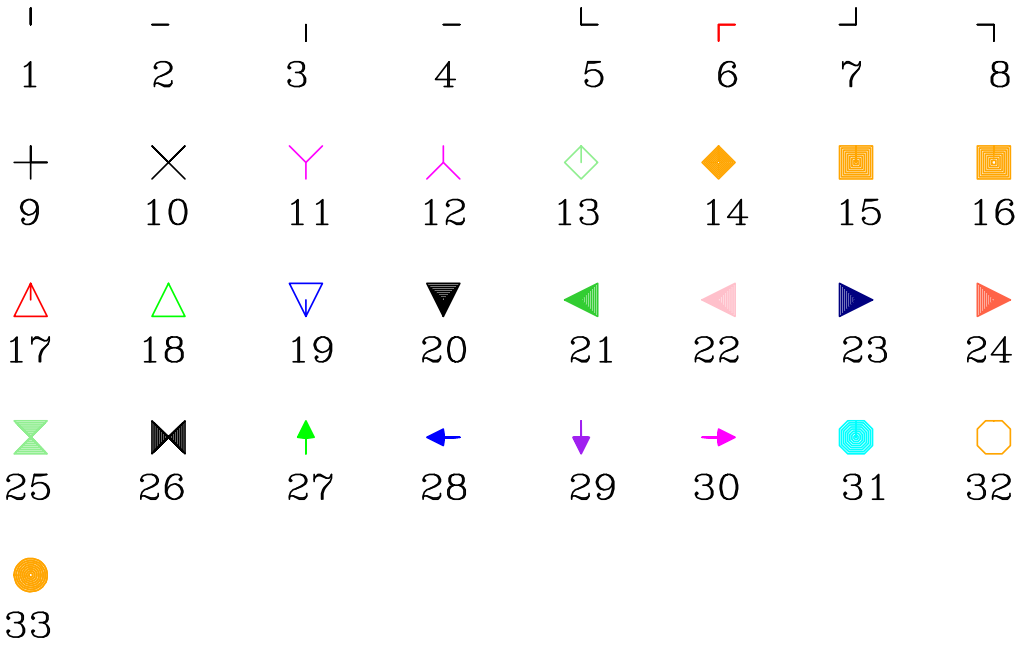

Figure 4: Allplo provides 33 symbols for labeling a point of a drawing. The symbols can be colored by 9 different colors shwon from symbol number 25-33. There is the option for symbols covering closed areas (13-33) to be filled. Symbol numbers 17-19 and 32 show only their contours.

Figure 5: Colors of the color file x11_ps_colors.txt defined by the user.

6 ALLPLO batch file

Table 7: The ALLPLO command file (batch file) used for figure 2 showing the available fonts. The file fonts.dat is very simple as it fixes the dimension of the frame of the figure by two adjacent points. The description of the command lines of the (batch file are available by {he} at each line starting xalpl with a new batch file.

***ALLPLO command file *** 04.01.08 user: spiering			
origin x,y	[0.00, 0.00]	width xw,yw	[15.00,10.00]
borderwidth bx,by	[0.40,0.40]	character cw,ch	[0.40,0.40]
color frame (0..9)	[0]		
x-axis		(none,Both,bottom,top)	[n]
break(none,bottom,top)	[n]	(left,right)	[l]
y-axis		(none,Both,left,right)	[n]
break(none,left,right)	[n]	(bottom,top)	[b]
data file name	[fonts.dat]		
structure		(allplo,mosfun,effi)	[allplo]
error bars (y/n)	[y]	clipping (y/n)	[n]
title (*...=no, #=fdf)	[title]		
name of x-axis (#=fdf)	[x-axis]		
name of y-axis (#=fdf)	[y-axis]		
symbol sw,sh	[0.40,0.40]	pen width	[0.03]
symbols	[34]		
x-data	(x ,log(x),10**x: /lgx/pwx)		[ ]
x-range(0.00,10.0)	[0.0,10.0]	(log scale y/n)	[ ]
scale (l,r,stepw,tick,pw)		[-1.0 4.0 0.5 5 0]	
y-data	(y ,log(y),10**y: /lgy/pwy)		[ ]
y-range(0.00,4.50)	[0.00,4.50]	(log scale y/n)	[ ]
scale (l,r,stepw,tick,pw)		[0.0 4.5 0.5 5 0]	
x-axis text size*	[1.0]	values, scale (y,n)	[y,y]
inscription	(normal,top,bottom,inverted)		[n]
y-axis text size*	[1.0]	values, scale (y,n)	[y,y]
inscription		(normal,left,right,inverted)	[n]
text: c/x,y,hx,hy,n_bf,n_col,l_sep,u/bn_col; text		file= [ ]	
0.,10.0,.5,0.5;ABCDEFGHIJKLMNOPQRSTUVWXYZ			
termination line for extra text (first letter *)			
more y/n	[n]	use old frame/scaling	[n]

7 ALLPLO data file

The ALLPLO data file begins with a comment line which is taken as the title of the plot (see table 7). The second and third lines are the name of the x- and y-axes, respectively. Data points start from the 4th line on.

Table 9: The ALLPLO data file used to plot figure 4 with the 33 symbols.

title		
name of x-axis		
name of y-axis		
1	1	10
2	2	10
3	3	10
4	4	10
5	5	10
2006	6	10
7	7	10
8	8	10
9	1	8
10	2	8
6011	3	8
6012	4	8
30013	5	8
8114	6	8
8115	7	8
8116	8	8
2017	1	6
3018	2	6
4019	3	6
120	4	6
31121	5	6
18122	6	6
23123	7	6
27124	8	6
30125	1	4
0126	2	4
3127	3	4
4128	4	4
12129	5	4
6130	6	4
7131	7	4
8032	8	4
8133	1	2

The numbers (first column) of the symbols are defined as follows: Symbol numbers range from 0 - 34. The shape of symbols 1 - 33 are shown in fig. 4. Symbol number

34 moves to the coordinate without drawing a line (for old plotters with a pen the 34 moves the lifted pen). Symbol number 0 moves (with pen down) by drawing a line to the destination point. The two input lines 34 0 0 and 0 1 1 draw a line from (0,0) to (1,1).
In order to define colors and to draw filles symbols the symbol numbers n are extended. n+100 fills the symbol (if possible) :133 gives a filled circle. n+1000*ic draws symbol n with color given by number ic. 2133 gives a red filled circle.
The move by symbol 0 produces a black line. Negative numbers -1000*ic give colored lines (color ic).
It is also possible to connect symbols by a line. The input line -2033 1 1 starts a red line from the actual position to (1,1) and draws an open red circle.