PRIMVS - Graphic Interpreter
User Reference Manual

[Version: 1.0]

Luigi Capodieci

Center for XRay Lithography
University of Wisconsin, Madison

Contents

Introduction

1.1 PRIMVS : a Graphic Interpreter

PRIMVS is a graphic interpreter built on top of the PLPLOT C-library. Its main purpose is to make available, in an interactive environment, a number of functionalities of the PLPLOT library and to allow the user to generate scientific plots and graphical representations with minimal effort. Standard $x - y$ plots, semilogaritmic and logaritmic plots, 3-dimensional and mesh plots can be realized almost instantaneously using a small number of simple commands. The graphical output can be displayed on a high-resolution terminal or sent to a file in several formats (PostScript, Tektronik, etc.). PRIMVS architecture is aimed at relieving the user from the burden of properly initializing and sequencing C-instructions and PLPLOT procedure calls. At the same time the intepreter guarantees enough flexibility for customization, so that sophisticated plots can be generated. No knowledge of the PLPLOT package or of the C programming language is necessary in order to use PRIMVS . Learning overhead is minimized by the presence of many examples that accompany this Manual.

PRIMVS kernel is writtent in C and C++, and can be compiled and installed on computers running Unix. The current version of PRIMVS is for use within the community of the Center for X-Ray Lithography at the University of Wisconsin-Madison. After the initial testing phase several commands have been added to the interpreter, based on users' feedback and preferences. Any comments or suggerstions are strongly encouraged. The author will promptly respond to any questions or problems that might arise. Please direct all communication to:

Luigi Capodieci
Center for X-Ray Lithography
University of Wisconsin-Madison
Madison, WI 53705

Internet:
lc@xraylith.wisc.edu

1.2 Obtaining a copy of PRIMVS

Please refer to the *Release Notes* in the Appendix.

1.3 Installing and Using PRIMVS

Please refer to the *Release Notes* in the Appendix.

1.4 Contents and Organization of this Manual

This Manual contains a Tutorial section, a Reference section and a Technical Appendix.

The first section describes a step by step procedure for generating and displaying (or printing) scientific plots and graphics. Commands are introduced in the same order they should appear in a PRIMVS command file, thus leading the user to a painless understanding of the correct sequence of operations.

The second section cointains a listing, in alphabetical order of all, the commands available in PRIMVS . The syntax of each command, a description of all the relative parameters and of all options is provided, along with a significant example.

Finally the Technical Appendix includes the current Release Notes and details about installing and using PRIMVS on different computing systems.

1.5 Acknowledgements

The author disclaims all warranties with regard to this software, including all implied warranties of merchantability and fitness, In no event shall the authors be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

PRIMVS is based on the PLPLOT 4.0 plotting library. The following is the acknowledgment statement and disclaimer relative to PLPLOT 4.0.

PLPLOT 4.0 was created through the effort of many individuals and funding agencies. We would like to acknowledge the support (financial and otherwise) of the following institutions:

The Institute for Fusion Studies, University of Texas at Austin

The Scientific and Technology Agency of Japan

Japan Atomic Energy Research Institute

Duke University

Universite de Nice

National Energy Research Supercomputer Center

Los Alamos National Labs

The authors disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness, In no event shall the authors be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

The PLPLOT source code, except header files and those files explicitly granting permission, may not be used in a commercial software package without consent of the authors. You are allowed and encouraged to include the PLPLOT object library and header files in a commercial package provided that: (1) it is explicitly and prominently stated that the PLPLOT library is freely available, and (2) the full copyrights on the PLPLOT package be displayed somewhere in the documentation for the package.

PLPLOT was first derived from the excellent PGPLOT graphics package by T. J. Pearson. All parts of PLPLOT not explicitly marked by a copyright are assumed to derive sufficiently from the original to be covered by the PGPLOT copyright:

The code in PLPLOT not derived from PGPLOT is Copyright 1992 by Maurice J. LeBrun and Geoff Furnish of the University of Texas at Austin and Tony Richardson of Duke University. Unless otherwise specified, code written by us as a part of this package may be freely copied, modified and redistributed without fee provided that all copyright notices are preserved intact on all copies and modified copies.

The startup code for plrender.c is from `xterm.c` of the X-windows Release 5.0 distribution, and we reproduce its copyright here:

Tutorial: Creating a Plot

2.1 General Concepts and Definitions

The plotting surface in PRIMVS is called the **page**. A page is for instance a window of a X terminal or a 8 by 11 sheet of paper. Plots are placed on rectangular **regions** defined on the page. Regions can be placed everywhere on the page and can have any dimension within the page.

World-coordinates, i.e. numeric values for X, Y and Z, are associated with a region so that data values can be mapped onto the region. Data is read from files (structured in form of X, Y, Z columns). The user can specify the format of the data to be read.

Plots can be annotated by surrounding a region with a **box** i.e. a frame specifying axis, grid, numerical and textual labels, along with tick and subtick markers. Text can be written anywhere and with any orientation inside and/or outside the box. Facilities are available for changing color, style and width of the lines. A character set of more than a 1000 symbols (from the PLPLOT library) is also implemented.

Although several plots can be placed on a **single page**, PRIMVS allows the user to specify a **multipage** mode, in which a page is divided into a set of **subpages**. Each subpage is treated as a single page and regions can be defined within page boundaries. This feature is useful for grouping families of plots.

The following conventions will be used for describing command syntax: any user-supplied parameter will be enclosed in circumflex marks, and square bracketts will indicate optional parameters, i.e.

```
command(keyword,<user-supplied-parameter>[,<optional-parameter>])
```

Anything not enclosed in circumflex marks must be used verbatim, when the command is issued.

2.2 Initializing the Page

The first step consists of creating a blank page. A plotting session is normally started by the command `initpage`. The command is used to define the graphic device on which the plot is to appear (for instance an X window terminal or a PostScript file). If a file device is selected, the corresponding filename must also be specified. The syntax for `initpage` is:

```
initpage(<interactive-device>)
```

in the case of an interactive device or:

```
initpage(<file-device>[,"<filename>"])
```

in the case of a file device. PRIMVS recognizes a number of interactive and file devices such as: `xwin` (X window terminal), `xterm` (X terminal), `tekt` (Tektronik terminal), `ps` (PostScript Fle), `xfig` (Xfig file). A filename can be specified only in the case of a file device. For example the command:

```
initpage(xwin)
```

creates a blank page for plotting operations on an X window. In order to create a blank page on a Postscript file named "primus_out.ps", the following command can be used:

```
initpage(ps,"primus_out.ps")
```

Figure 2.1: Multipage Initialization

If a file device is chosen, the user can specify an orientation using the commands:

```
orientation(landscape) or orientation(portrait)
```

An alternative way to initialize a plotting session is provided by the command:

```
initmultipage(<x-pages>,<y-pages>,<device>[,"<filename>"])
```

This command divides the page into subpages (`<x-pages>` in the X direction and `<y-pages>` in the Y direction). The device is specified as in the single page initialization. For example the command:

```
initmultipage(3,2,xwin)
```

creates a blank page divided into 6 subpages (3 in the horizontal direction and 2 in the vertical direction) as shown in Figure 2.1. Each subpage is treated as an independent page and plots are placed on the current subpage. The page can be changed using:

```
subpage(<page-number>)
```

This command can be issued only in multipage mode.

The TUTORIAl SECTION of this manual is scheduled to be completed by the end of SUMMER 1993

2.3 Defining the Region and the Coordinates

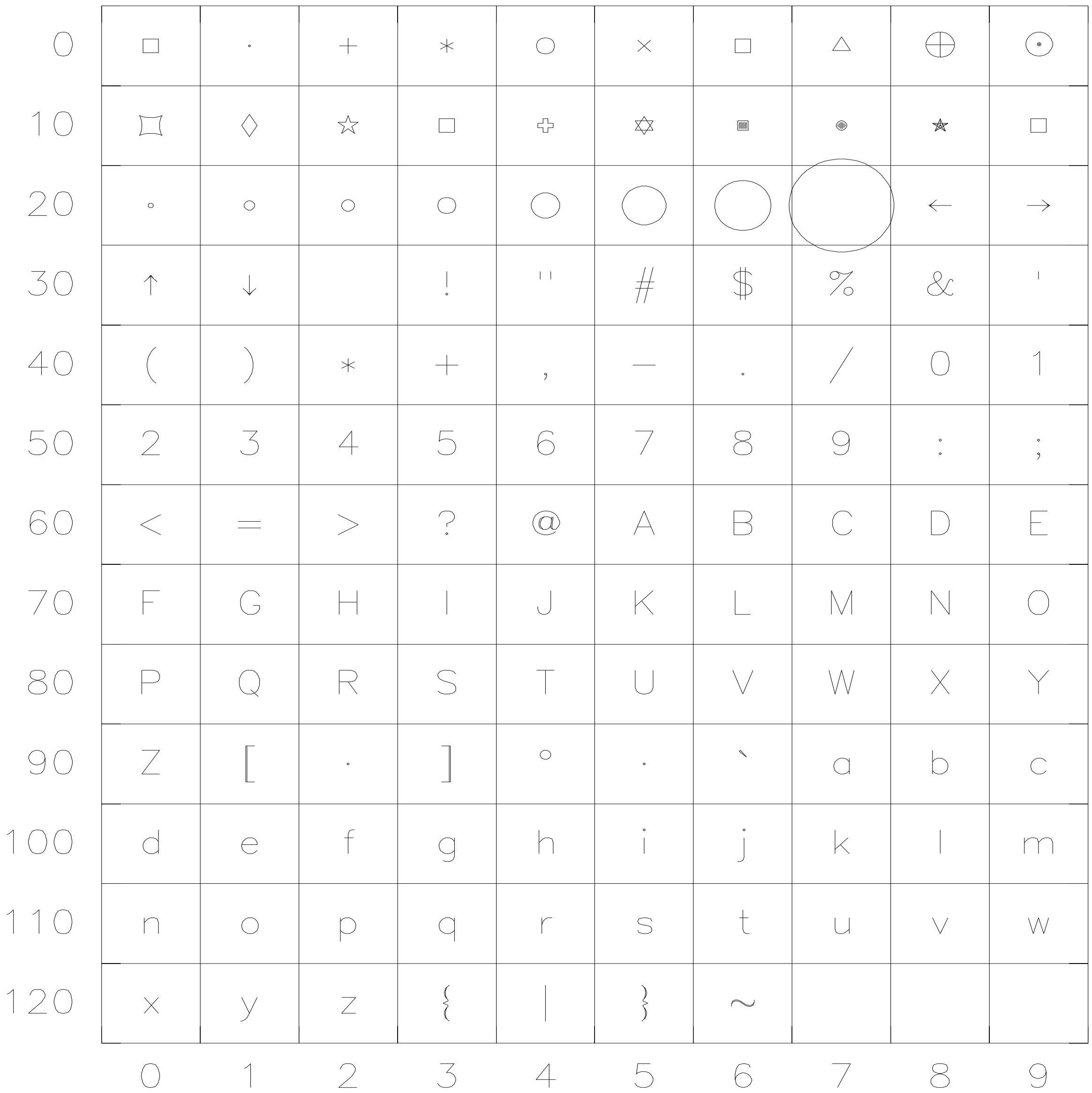

Figure 2.2: PRIMVS Symbol Set

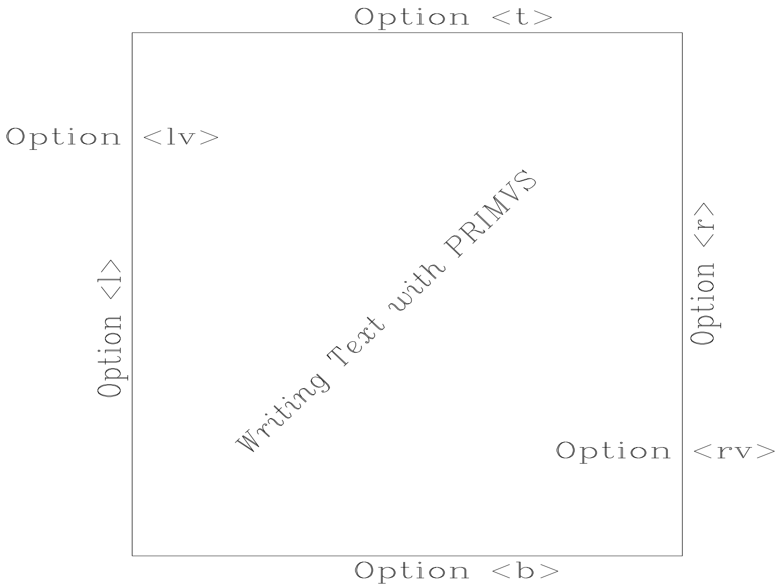

Figure 2.3: Writing Text with PRIMVS

PRIMVS **Command Reference**

All the PRIMVS Commands are listed below.

at("<command_file>")

Executes the PRIMVS commands contained in the command file. If the command file is not in the current directory the complete path must be specified. The at command supports multiple *nesting*, i.e. a command file can contain invocation of command files and so on. Recursion is not allowed.

autoscale

Sets the autoscale flag for all subsequent plotting operations (two and three dimensional). User's definitions of x, y and z ranges are overridden. The values associated with the borders of the plotting region are the minima and the maxima of the plotting intervals. The command noautoscale is used to disable autoscaling.

background

Sets the background color for all subsequent plotting operations. This command can be issued only before initializing a device. Default backgrounds are black for terminal devices (i.e. XWIN) and white for file devices (i.e. PostScript).

```
box("<x_options>",<x_tickinterval>,<x_ticksubinterval>,
    "<y_options>",<y_tickinterval>,<y_ticksubinterval>)
```

Draws a box around the currently defined region, and labels it with user coordinate values appropriate to the previously defined ranges. The character strings <x_options> and <y_options> specify how the box should be drawn as described below. If ticks and/or subticks are to be drawn for a particular axis, the tick intervals and number of subintervals may be specified explicitly, or they may be defaulted by setting the appropriate arguments to zero.

`x_options`:

Character string specifying options for horizontal axis. The string can include any combination of the following letters (upper or lower case) in any order:

- `a` : Draws axis, X-axis is horizontal line `y=0`, and Y-axis is vertical line `x=0`.
- `b` : Draws bottom (X) or left (Y) edge of frame.
- `c` : Draws top (X) or right (Y) edge of frame.
- `g` : Draws a grid at the major tick interval.
- `i` : Inverts tick marks, so they are drawn outwards, rather than inwards.
- `l` : Labels axis logarithmically. This only affects the labels, not the data, and so it is necessary to compute the logarithms of data points before passing them to any of the drawing routines.
- `m` : Writes numeric labels at major tick intervals in the unconventional location (above box for X, right of box for Y).
- `n` : Writes numeric labels at major tick intervals in the conventional location (below box for X, left of box for Y).
- `s` : Enables subticks between major ticks, only valid if `t` is also specified.
- `t` : Draws major ticks.

`x_tickinterval`:

User coordinate interval between major ticks on the x axis. If it is set to zero, a suitable tick interval is generated.

x_ticksubinterval:

Number of subintervals between major x axis ticks for minor ticks. If it is set to zero, a suitable minor tick interval is generated.

y_options:

Character string specifying options for vertical axis. The string can include any combination of the letters defined above for x_options, and in addition may contain:

- v : Write numeric labels for vertical axis parallel to the base of the graph, rather than parallel to the axis.

ytick (real, input):

World coordinate interval between major ticks on the y axis. If it is set to zero, a suitable tick interval is generated.

y_tickinterval:

User coordinate interval between major ticks on the y axis. If it is set to zero, a suitable tick interval is generated.

y_ticksubinterval:

Number of subintervals between major y axis ticks for minor ticks. If it is set to zero, a suitable minor tick interval is generated.

```
box3d("<x_options>",<x_tickinterval>,"<y_options>",<y_tickinterval>,
      "<z_options>",<z_tickinterval>)
```

Draws axes, numeric and text labels for a three-dimensional surface plot. Numeric values and labels can be defined with the xyzrange and label3d commands, or alternatively autoscaling can be used. The character strings x_options, y_options, and z_options specify how the three dimensional axis should be drawn, as described below. If the values of the axis ticks are set to zero suitable tick intervals are generated. For the meaning of the parameters: base and height please refer to the viewbox command.

x_options:

Character string specifying options for the x axis. The string can include any combination of the following letters (upper or lower case) in any order:

- b : Draws axis at base, at height z=zmin where zmin is defined either by xyzrange or autoscale. This character must be specified in order to use any of the other options.
- i : Inverts tick marks, so they are drawn downwards, rather than upwards.
- l : Labels axis logarithmically. This only affects the labels, not the data, and so it is necessary to compute the logarithms of data points before passing them to any of the drawing commands.
- n : Writes numeric labels at major tick intervals.
- s : Enables subticks between major ticks, only valid if t is also specified.
- t : Draws major ticks.
- u : If this is specified, the text label for the axis is written under the axis.

x_tickinterval:

User coordinate interval between major ticks on the x axis. If it is set to zero, a suitable tick interval is generated.

y_options:

Character string specifying options for the y axis. The string is interpreted in the same way as x_options.

y_tickinterval:

User coordinate interval between major ticks on the y axis. If it is set to zero, a suitable tick interval is generated.

z_options:

Character string specifying options for the z axis. The string can include any combination of the following letters (upper or lower case) in any order:

- `b` : Draws z axis to the left of the surface plot.
- `c` : Draws z axis to the right of the surface plot.
- `d` : Draws grid lines parallel to the x-y plane behind the figure. These lines are not drawn until after `plot3d` is used because of the need for hidden line removal.
- `i` : Inverts tick marks, so they are drawn away from the center.
- `l` : Labels axis logarithmically. This only affects the labels, not the data, and so it is necessary to compute the logarithms of data points before passing them to any of the drawing commands.
- `m` : Writes numeric labels at major tick intervals on the right-hand verical axis.
- `n` : Writes numeric labels at major tick intervals on the left-hand verical axis.
- `s` : Enables subticks between major ticks, only valid if `t` is also specified.
- `t` : Draws major ticks.
- `u` : If this is specified, the text label is written beside the left-hand axis.
- `v` : If this is specified, the text label is written beside the right-hand axis.

`z_tickinterval`:

User coordinate interval between major ticks on the z axis. If it is set to zero, a suitable tick interval is generated.

`breakpage`

Starts a new blank page on the current device, using previously specified mode (single or multi page).

`closepage`

Closes the current page, and the previuos pages if `breakpage` has been used. This command is required before reinitializing a new device. Window refreshing for interactive devices is active only after the page has been closed.

```
color(<a_color>)
```

Sets the color for subsequent lines. Color may be specified either by name or by number.

`a_color`:

Integer representing the color or name of a color. The defaults at present are (these may change):

0 coral

1 red

2 yellow

3 green

4 aquamarine

5 pink

6 wheat

7 grey

8 brown

9 blue

10 BlueViolet

11 cyan

12 turquoise

13 magenta

14 salmon

15 white (default foreground)

16 black (default background)

```
colorbar(<x_LowerLeftCorner>,<y_LowerLeftCorner>,
         <x_UpperRightCorner>,<y_UpperRightCorner>)
```

Draws a bar representing the intensity levels for a color plot. The color levels are mapped onto the interval $[zmin, zmax]$. This range can be defined by the user or by autoscaling. In the current implementation the color interval is between dark brown (low intensity) and red (high intensity), with a linear interpolation in the (H, L, S) space. The position of the bar is given in relative coordinates between 0.0 and 1.0 analogously to the regionp command.

COMMENT

The user can insert comments in PRIMVS command files using the # symbol. All the characters between # and *carriage return* are ignored by the PRIMVS interpreter.

exit

Terminates PRIMVS execution (both interactive and batch) and returns to the Unix shell.

format(x:<x_column>,y:<Y_column>)

Formats subsequent file reading operations, for two-dimensional plots. A data file is structured as a set of columns containing data points. The format command instructs the reading procedure to collect the values of X or Y from the relative column (default X values from column 1, Y values from column 2)

greybar(<x_LowerLeftCorner>,<y_LowerLeftCorner>,
<x_UpperRightCorner>,<y_UpperRightCorner>)

Draws a bar representing the intensity levels for a greyscale plot. The greyscale levels are mapped onto the interval $[zmin, zmax]$. This range can be defined by the user or by autoscaling. In the current implementation the greyscale interval is between black (low intensity) and white (high intensity), with a linear interpolation in the (R, G, B) space. The position of the bar is given in relative coordinates between 0.0 and 1.0 analogously to the regionp command.

```
gtext(<x>,<y>,<dx>,<dy>,<just>,"<the_text>"
```

Writes text at a specified position and inclination within the region. Text is clipped at the boundaries. The reference point of a string lies along a line passing through the string at half the height of a capital letter. The position of the reference point along this line is determined by `just`, the reference point is placed at user coordinates `(x, y)` within the region. The inclination of the string is specified in terms of differences of user coordinates making it easy to write text parallel to a line in a graph.

`x`:

x coordinate of reference point of string.

`y`:

y coordinate of reference point of string.

`dx`:

Together with `dy`, this specifies the inclination of the string. The baseline of the string is parallel to a line joining `(x, y)` to `(x+dx, y+dy)`.

`dy`:

Together with `dx`, this specifies the inclination of the string.

`just`:

Specifies the position of the string relative to its reference point. If `just=0`, the reference point is at the left and if `just=1`, it is at the right of the string. Other values of `just` give intermediate justifications.

`the_text`:

The string to be written out.

`initmultipage(<pages_on_x>,<pages_on_y>,<device> [,"<filename>"])`

Initializes the first page for plotting operations, on the chosen device, using subpaging. The page is divided into a number of subpages (the total number of subpages is: `<pages_on_x>` times `<pages_on_y>`). If the device is a file device a filename can be specified. Interactive devices include: `xwin` (X Window terminal), `xterm` (X terminal) and `tekt` (Tektronik terminal). File devices include: `ps` (PostScript) and `xfig` (X FIG format). If a file name is not specified the default output is `primvs.out`.

`initpage(<device> [,"<filename>"])`

Initializes the first page for plotting operations, on the chosen device. If the device is a file device a filename can be specified. Interactive devices include: `xwin` (X Window terminal), `xterm` (X terminal) and `tekt` (Tektronik terminal). File devices include: `ps` (PostScript) and `xfig` (X FIG format). If a file name is not specified the default output is `primvs.out`.

`label("<x_label>","<y_label>","<top_label>"`

Writes simple labels for two dimensional plots. Use `text` or `gtext` for more complex labels and text. Numeric labelling is specified by `box`.

`x_label`:

Label for horizontal axis.

`y_label`:

Label for vertical axis.

`top_label`:

Title of graph (drawn on top of the region).

```
label3d("<x_label>","<y_label>","<z_label>"
```

Writes simple labels for three dimensional plots. Use `text` or `gtext` for more complex labels and text. Numeric labelling is specified by `box3d`. Labels are drawn parallel to the x,y,z axis according to the chosen orientation of the three dimensional box.

`x_label`:

Label for x axis.

`y_label`:

Label for y axis.

`z_label`:

Label for z axis.

```
line(<x1>,<y1>,<x2>,<y2>)
```

Draws a line (using the current style and width) joining the two points $(x1, y1)$,$(x2, y2)$. The coordinates of the two points are the user coordinates associated with the current page.

```
linestyle(<a_style>)
```

Specifies one of **eight** predefined styles for drawing a line.

```
linewidth(<a_width>)
```

Specifies the device dependent width for subsequent line drawing. In the current implementation the pen width can be changed only for PostScript devices.

`noautoscale`

Disables autoscaling for all subsequent plotting operations. It must be issued after a previous execution of `autoscale` if user's ranges need to be defined.

`orientation(<an_orientation>)`

Sets the orientation of the page. It must be executed before initializing the device and the page. The two possible values for `orientation` are: `landscape` (default) and `portrait`. For interactive high resolution devices the command has no effect.

`pause`

Suspends PRIMVS execution (both interactive and batch) until the user strikes the `enter` key. It can be used during an interactive session in order to display one plot at a time.

`plot3d("<function_datafile>" ,<data_format>,<3D_options>)`

Generates a three dimensional plot of the function contained in the datafile. The `data_format` parameter specifies the structure of the data file. The value 0 indicates a `XRayLith` format, as described below:

1	# Number of Steps (Unused)
0.05	# Delta X
0.03	# Delta Y
256	# Number of X values
123	# Number of Y values
5.0	# Gap value (Unused)
0.4567	# Z value at point (X_1, Y_1)
1.3256	# Z value at point (X_2, Y_1)
-0.8712	# Z value at point (X_3, Y_1)
....	# Z value at point (X_i, Y_j)
....	# Z value at point (X_{i+1}, Y_j)
0.3211	# Z value at point (X_{256}, Y_{123})

The number of x and y values along with the x and y steps define a grid in the x-y plane. A value of z must be specified for each point in the grid. The `3D_options` parameter determines how the surface is represented. A value of 1 causes lines to be drawn showing `z` as a function of `x` for each value of `y[j]`. A value of 2 causes lines to be drawn showing `z` as a function of y for eachvalue of `x[i]`. A value of 3 causes a network of lines to be drawn.

Before a three dimensional plot can be generated, ranges for the x,y,z variables must be defined, either using `autoscale` or `xyzrange`. The user can also modify the viewpoint of the 3D plot by using the commands: `view` and `viewbox`.

The value 1 for the `data_format` indicates a X Y Z format for the data file (it will be implemented in the next release of PRIMVS).

`plotcolor("<function_datafile>" ,<data_format>)`

Generates an intensity level color plot for the function contained in the datafile. The parameter `data_format` has the same meaning as in the `plot3d format`. Each value of the function $z(x, y)$ is mapped onto a colormap of 100 colors linearly distributed in the (H, L, S) space. The interval $[zmin, zmax]$ can be defined by the user with `xyzrange` or by autoscaling. For each point in the x-y grid a colored rectangle is plotted. The rectangle has dimensions delta X and delta Y and is centered at (X, Y) resulting into a continuos intensity level color plot. The user can vary the plotting resolution by using the `resolution` command.

`plotcont("<function_datafile>" ,<data_format> , <nlevels>)`

Generates a contour plot for the function contained in the datafile. The parameter `data_format` has the same meaning as in the `plot3d format`. The parameter `nlevels` determines how many contour levels are to be plotted. These levels are equally spaced in the interval $[zmin, zmax]$ that can be defined by the user with `xyzrange` or by autoscaling.

`plotgrey("<function_datafile>" ,<data_format>)`

Generates an intensity level greyscale plot for the function contained in the datafile. The parameter `data_format` has the same meaning as in the `plot3d format`. Each value of the function $z(x, y)$ is mapped onto a greyscale of 100 grey levels linearly distributed in the (R, G, B) space. The interval $[zmin, zmax]$ can be defined by the user with `xyzrange` or by autoscaling. For each point in the x-y grid a grey rectangle is plotted. The rectangle has dimensions delta X and delta Y and is centered at (X, Y) resulting into a continuos intensity level greyscale plot. The user can vary the plotting resolution by using the `resolution` command.

```
plotl("<function_datafile>" [,<style>])
```

Plots the function defined by the data points in the specified file. The points are linearly connected, resulting in a continuos curve. A line style can be chosen among 8 predefined ones.

```
plotp("<function_datafile>" [,<symbol>])
```

Plots the function defined by the data points in the specified file. The points are drawn using the currently selected `symbol`. Alternatively a symbol value can be specified as an optional parameter.

```
regionp(<x_LowerLeftCorner>,<y_LowerLeftCorner>,
        <x_UpperRightCorner>,<y_UpperRightCorner>)
```

Defines a rectangular region on the page. The coordinates of the lower left corner and upper right corner of the rectangle are specified in percentage with respect to the page size (from $[0, 0]$ to $[1, 1]$). This command is used for relative positioning on the page for any device, independently of the actual page size. The command is required before starting plotting operations.

```
regionm(<x_LowerLeftCorner>,<y_LowerLeftCorner>,
        <x_UpperRightCorner>,<y_UpperRightCorner>)
```

Defines a rectangular region on the page. The coordinates of the lower left corner and upper right corner of the rectangle are specified in millimeters. This command is used only for absolute positioning on the page, when the page size is known.

```
regionr(<x_LowerLeftCorner>,<y_LowerLeftCorner>,
        <x_UpperRightCorner>,<y_UpperRightCorner>,<Y:X_Ratio>)
```

Defines a rectangular region on the page. The region is chosen to be the largest with the given aspect ratio, that fits in the rectangle whose coordinates (lower left corner and upper right corner) are specified in percentage with respect to the page size (from $[0,0]$ to $[1,1]$). The aspect ratio value specifies the ratio between the length of the Y axis and the lenght of the X axis. This command is used for relative positioning on the page for any device, independently of the actual page size, but unlike the `regionp` command it does not reserve space for labels at the edges of the region.

```
resolution(<res_value>)
```

Specifies a resolution for `plotcolor` and `plotgrey` commands. The resolution value represents the distance between lines in the solid rectangles drawn to display intensity levels. A value of 1.0 is the minimum value allowed. Values larger than 1.0 reduce the resolution and increase plotting speed.

```
setcolor(<color_flag>)
```

Turns color plotting ON and OFF. A value of 1 (default) enables color plotting. A value of 0 disables color plotting. The command must be issued before initializing the device. Using `setcolor` is recommended when black and white PostScript files are to be created.

`scalechar(<scaling_factor>)`

Scales character size for subsequent text drawing. If the value of the `scaling_factor` is 0 the character size is set to its default. Otherwise the specified factor is applied to the default in order to obtain the actual size.

`scalesym(<scaling_factor>)`

Scales symbol size for subsequent plotting with `plotp`. If the value of the `scaling_factor` is 0 the symbol size is set to its default. Otherwise the specified factor is applied to the default in order to obtain the actual size.

`subpage(<subpage_number>)`

Moves to the specified subpage. All subsequent plotting operations will take place on the chosen subpage. It can be used only in multipage mode.

`symbol(<a_symbol>)`

Specifies a symbol for the `plotp` command. Available symbols are represented in Figure 2.2.

`! [sysexec: execute Unix command]`

Executes all Unix commands between ! and *carriage return* then returns control to the PRIMVS interpreter.

`system`

Escapes to Unix Shell. After shell execution is terminated by the `exit` command, control is returned to the PRIMVS interpreter.

```
text("<side>",<disp>,<pos>,<just>,"<the_text>")
```

Writes text at a specified position relative to the region boundaries. Text may be written inside or outside the region, but is clipped at the subpage boundaries. The reference point of a string lies along a line passing through the string at half the height of a capital letter. The position of the reference point along this line is determined by `just`, and the position of the reference point relative to the region is set by `disp` and `pos`.

`side`:

Specifies the side of the region along which the text is to be written. The string must be one of:

- b: Bottom of region.
- l: Left of region, text written parallel to edge.
- lv: Left of region, text written at right angles to edge.
- r: Right of region, text written parallel to edge.
- rv: Right of region, text written at right angles to edge.
- t: Top of region.

`disp`:

Position of the reference point of string, measured outwards from the specified region edge in units of the current character height. Use negative `disp` to write within the region.

`pos`:

Position of the reference point of string along the specified edge, expressed as a fraction of the length of the edge.

`just`:

Specifies the position of the string relative to its reference point. If `just=0`, the reference point is at the left and if `just=1`, it is at the right of the string. Other values of `just` give intermediate justifications.

the_text:

The string to be written out.

`view(<altitude>,<azimuth>)`

Defines a view point for a three dimensional plot. The parameter altitude specifies the viewing altitude in degrees above the xy plane. Altitude values must be restricted to the interval $0°$ - $90°$. The parameter azimuth specifies the viewing azimuth in degrees. When azimuth is $0°$, the observer is looking face onto the zx plane and, as azimuth is increased (up to $360°$) the observer moves clockwise around the three dimensional box, when viewed from above the xy plane.

`viewbox(<basex>,<basey>,<height>)`

Defines a *normalized box* for a three dimesional plot. 3D plotting is carried out in two stages. First the cuboid $[(xmin, xmax), (ymin, ymax), (zmin, zmax)]$ expressed in *world coordinates* is mapped onto a normalized box of dimensions $(basex, basey, height)$, expressed in *world coordinates*. The normalized box is centered about the origin in the x and y directions, but rests on the plane $z = 0$. Then this box is viewed by an observer located at altitude and azimuth specified by the command view. The command viewbox has effect only if autoscaling is disabled. In this case the user must define a set of coordinates for the tridimensional box using xyzrange. Using viewbox allows for more flexibility in 3D plotting especially in cases when the dimensions of the *world cuboid* on the three axes are very different. When autoscaling is enabled a suitable normalized box is automatically generated by the plot3d command.

`xyrange(<xmin>,<xmax>,<ymin>,<ymax>)`

Defines a system of user coordinates to be associated with the current region. The command is required before starting two-dimensional plotting operations. Alternatively the autoscale command can be used.

`xyzrange(<xmin>,<xmax>,<ymin>,<ymax>,<zmin>,<zmax>)`

Defines a system of user coordinates to be associated with the current region. The command is required before starting three-dimensional plotting operations. Alternatively the `autoscale` command can be used.

`Escape Sequences in Text`

It is possible to include escape sequences in the text to be plotted. These are character sequences that are interpreted as instructions to change fonts, draw superscripts and subscripts, draw non-ASCII (e.g. Greek), and so on. All escape sequences start with a number symbol (#). The effect of a font switch is limited to the string in which the switch itself appears. The default font can be changed by using the `font` command.

The following escape sequences are defined:

- #u: move up to the superscript position (ended with #d)
- #d: move down to subscript position (ended with #u)
- #b: backspace (to allow overprinting)
- ##: number symbol
- #+: toggle overline mode
- #-: toggle underline mode
- #gx: Greek letter corresponding to Roman letter `x` (see below)
- #fn: switch to normal font
- #fr: switch to Roman font
- #fi: switch to italic font
- #fs: switch to script font
- #(nnn): Hershey character nnn (1 to 4 decimal digits)

Roman	A	B	G	D	E	Z	Y	H	I	K	L	M
Greek	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M
Roman	N	C	O	P	R	S	T	U	F	X	Q	W
Greek	N	Ξ	O	Π	P	Σ	T	Υ	Φ	X	Ψ	Ω
Roman	a	b	g	d	e	z	y	h	i	k	l	m
Greek	α	β	γ	δ	ϵ	ζ	η	θ	ι	κ	λ	μ
Roman	n	c	o	p	r	s	t	u	f	x	q	w
Greek	ν	ξ	o	π	ρ	σ	τ	υ	ϕ	χ	ψ	ω

Table 3.1: Roman characters corresponding to Greek characters

Sections of text can have an underline or overline appended. For example, the string

$$\overline{\mathrm{S}}(\underline{\mathrm{freq}})$$

is obtained by specifying ¨#+S#+(#-freq#-)”.

Greek letters are obtained by #g followed by a Roman letter.

Table 3.1 shows how these letters map into Greek characters.