GreG
Grenoble Graphic

A **GILDAS** working group software

February 7, 2002

The GILDAS working group is a collaborative project of the Observatoire de Grenoble (1,3) and IRAM (2), and comprises: G. Buisson[1], L. Desbats[1], G. Duvert[1], T. Forveille[1], R. Gras[3], S. Guilloteau[1,2], R. Lucas[1,2], and P. Valiron[1].

(1) Laboratoire d'Astrophysique
Observatoire de Grenoble
BP 53 X
414 Rue de la Piscine
F-38402 Saint Martin d'Hères CEDEX

(2) Institut de Radio Astronomie Millimétrique
300 Rue de la Piscine
F-38406 Saint Martin d'Hères

(3) CEPHAG
Observatoire de Grenoble
F-38402 Saint Martin d'Hères CEDEX

Contributions from and invaluable discussions with J. Cernicharo, P. Begou[3], S. Delahaye[3], A. Dutrey[1,2], C. Kahane[1], P. Monger, J.L. Monin[1], R. Neri[2] and all **GILDAS** users are gratefully acknowledged.

Contents

1 Introduction

1.1 Philosophy

GreG is a graphic utility package whose goal is to help people prepare plots of their data or of the results of theoretical computations. Major goals were device independence as much as possible `AND` efficient interactive use. It was kept in mind that drawings are usually prepared interactively and then the final result is sent to a good quality printer or saved for later use. Quick look on data is also often wanted. Hence, the design goals were both a simple, fast utility for simple things and a powerful, flexible system producing high quality plots. In addition, it was soon realised that a stand-alone utility was unadapted for complex, repetitive integrated plots as often required in theoretical modelling. Accordingly, the library version became part of the design goals.

With the advent of the X-Window system, GREG has been totally rewritten to take advantage of the multi-window and color facilities of X-Window. This new version also handles bitmap display of images, contour filling, etc..., with hardcopy facilities on PostScript devices.

1.2 Getting started

Beginners should of course read Section 2, the beginner's guide. While doing this, you need only retain the topics with which you are concerned at first view. If anything seems to be lacking, then read the other topics.

After a few trials (say a few days of normal usage or a few drawings), most users will efficiently benefit from reading carefully Section 3. With some more practice, they will soon reach normal proficiency in **GreG** standard usage and be able to tackle the most funny things.

Section 4, the "advanced" user's guide, is important for those who want to get the best of **GreG** with colours, images, etc... It is also important for the application programmer who may use these features to provide a top layer for integrated applications.

Section 5,6,7 contains descriptions of all **GreG** commands on a command by command basis. This is a copy of the internal help. Please keep in mind that because of **GreG** evolution it may not be completely up to date.

At this stage, model builders (as opposed to observers) will most likely want to go further by using the `GREGLIB` and thus reading Section 8. This may be done sooner, but it is not recommended to use the `GREGLIB` before the basic concepts of **GreG** have been mastered.

The documentation on the **SIC** monitor is now provided separately in the manual **SIC** documentation, and at least the first chapter of this manual should be read by the beginners. The monitor offers many flexible capabilities such as procedures, symbols, vector mathematics, and so on.

The other sections concern only programmers of special purpose tasks, or system builders, and their lecture will be almost necessarily complemented by reading the **SIC** manual also. Note however that if you have some specific graphic problem, it is most likely solved somewhere in the **GreG** package, so that the answer may well be under these sections.

Know what you want ? Then go ahead, and good luck.

2 GreG CookBook

This chapter is a very simplified introductory manual to **GreG** designed for occasional user who just want an introduction or better, a recipe. It just describes by means of a commented procedure, how to obtain a standard plot for the most usual cases. A user will often want to do more than described here. This is usually not only possible, but even easy. However, such user will have to understand fully the basic concepts of **GreG** and to read more or less completely the **GreG** `MANUAL`.

Before starting, remember that **GreG** keeps track of all effective commands in an internal buffer for replay capabilities used mostly for repetitive actions and in a log file for post-session control. This log file can be replayed by **GreG** as any other valid procedure of **GreG** commands, thus making of **GreG** a safe system protected against system (or internal...) crashes.

You must know also that **GreG** uses virtual memory to store all plot coordinates in a device independent way. Hence a plot is entirely independent of any graphic device. You can turn off your graphic device, provided you do not exit from **GreG** your plot is still available. This Virtual Plot storage also implies that your plot and the way you display it are really two different things. While it may seem confusing at the beginning, this make possible very nice features such as internal Zooming, corrections of erroneous part of the plot and a fast Hardcopy system.

2.1 Starting, Syntax and Help

GreG is started by typing `GREG` on your computer. You should normally get a few text lines, then the prompt `GreG>` by which **GreG** indicates it is at your service. If that does not work, ask your system manager, he should know.

The command syntax is describe in the **SIC** manual. Commands have positional arguments, and the argument list can be followed by options (names preceded by a slash /), each of which may also have a list of argument. Arguments and options are separated by spaces. If your are lost, do not hesitate to type `HELP`. **GreG** is largely self documented, and it is quite a good idea to use this facility rather than always refer to the manual.

2.2 Coordinates

There are several coordinate systems in **GreG** two of which are the usual basic coordinate systems of any graphic package :

- *the paper or Physical Coordinate System*, whose units are arbitrarily defined as "centimeters" (the size of such "centimeter" may vary from one graphic support to the other, but the dimension is always that of a distance)
- *the User Coordinate System*, whose units may be anything from "Butterflies per Megaparsec cube" to "Molecules per Kelvin degree and per km/s", or anything sensible like this.

Plotting data is only a way to match these two coordinate systems, and additional work such as labelling and annotation may be most conveniently referred to one or the other system. **GreG** handles the transformation of coordinates and allows access to each system individually.

2.3 Demonstration Procedure for GreG capabilities

All commands given here are followed by a comment area (after the exclamation mark `!`) in which a number refers to the detailed explanation after the text of the procedure. Comment lines are also in the body of the procedure for clarity.

```
!
! Basic plot : a simple curve
DEVICE XLANDSCAPE WHITE                    !1
SET PLOT LANDSCAPE                         !2
SET FONT SIMPLEX                           !3
COLUMN X 1 Y 2 /FILE gag_demo:greg1.tst    !4
LIMITS                                     !5
BOX                                        !6
LABEL "Offset frequency (s\\U-1\\D)" /X    !7
LABEL "Signal strength (Jy)" /Y            !8
CONNECT                                    !9
!
! More difficult: Markers and Errorbars
CLEAR PLOT                                 !10
LIMITS -50 50 * *                          !11
SET MARKER 4 1 .1                          !12
POINTS                                     !13
CURVE                                      !14
COLUMN Z 3                                 !15
ERRORBAR Y                                 !16
LABEL "Offset frequency (s\\U-1\\D)" /X    !17
LABEL "Signal strength (Jy)" /Y            !18
SET CENTERING 5                            !19
DRAW TEXT 0.0000E+00 10.70 "Demonstration of GREG : Curve and Errorb-
ars" /USER                                 !20
!
HARDCOPY /PLOT                             !21
```

1. First, define the graphic support to which you want to send all graphic output. Here this is an X-Window terminal in landscape mode, with white background: a new window with the title <`GREG` will appear. Note that it is not necessary to define a graphic support to begin drawing data. You may define (or change) your graphic support at any moment during the session without losing your plot.

2. Specify the plot page has a "landscape" aspect (30 by 21 cm).

3. Specify the character font is the simplex one (the faster to plot).

4. Then read your data. The data file has a "Table"-like organisation one column describes all the values of a given variable for all the data points, while one line contains the values of all the variables for a given data point. You can access any part of the input table. The data is read in list-directed format, so that values must be written in formatted way, with spaces, tabs or commas as separators. Here, the X value is read in column 1 (the Y value in column 2) from lines 4 to 30 of the input file `gag_demo:greg1.tst`. `gag_demo:` is a *logical name* translated to a directory path by the software, so that `gag_demo:greg1.tst` will be expanded to something like `/users/soft/gag/demo/greg1.tst`.

5. Define the values of the X and Y coordinates at the corners of the plot window. Here, the automatic setting is used; it computes extrema of X and Y values, then add some 10

percent margin to get the plot limits. `LIMITS` is a very fundamental command, because it defines the conversion formula between the user coordinates and the physical coordinates. As **GreG** almost always plots in User coordinates, no plot action should be issued before command `LIMITS`.

6. Draw labelled axis around the plot window. This is useful in any plot, but contrary to `LIMITS`, this command is not compulsory. It can be issued at any moment (after `LIMITS` of course).
7. Write the caption of the X axis, in particular the type and units of the user coordinate.
8. Same as above for the Y axis. User coordinates can be anything as you can see...
9. Now connect the data points by straight lines to get a simple curve. If your data is not sorted, it will be quite a mess... Note that command `BOX, LABEL` and `CONNECT` could appear in any order, but after the command `LIMITS`. At this stage, your plot looks like the figure 1.

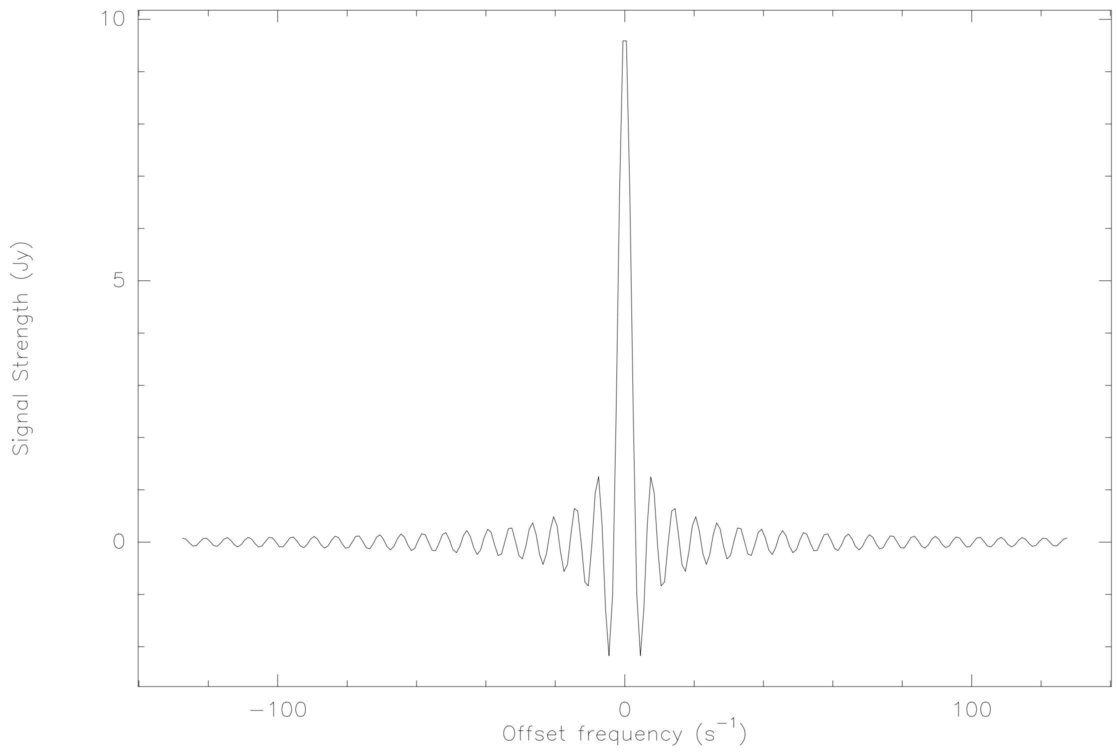

Figure 1: The X-Window graphic window just before clearing the plot in the demonstration procedure.

10. This command not only clears the graphic screen, but also destroys the current drawing. A new empty window is re-created afterwards.
11. Define the values of the X and Y coordinates at the corners of the plot window. Here, manual setting is used for X axis, and automatic setting for Y axis.

12. Define the type of marker you want. A marker is a regular polygon defined by its number of summits (0 to infinite), the type of polygon (0 to 4) and diameter in physical units (which on a paper output will be centimeters). Try the different values for the type to see what it means.

13. Now, plot a marker as defined above at each data point. Note that you do not need to read again your data, neither to define again the user coordinates, the last values have been retained.

14. and connect the data points by a smooth curve.

15. Read data into the Z buffer from column 3. You need not redefine the data file, neither the line range which are the same as in command number 2. The Z buffer is similar to the X and Y arrays into **GreG** but used in different ways.

16. Use the values in the Z buffer to draw symmetric error bars along the Y axis at each data point. The Z values will define the length of each side of the bar in Y user coordinates.

17. Same effect as above, but since the plot has been cleared...

18. idem...

19. idem...

20. Specify the text centering mode

21. write some legend to the figure

22. This plot seems good, make a hardcopy to save it. With the option /PLOT the command sends the hardcopy to the default system printer. This command has many other possibilities, such as creating PostScript files. Your plot should look like figure 2.

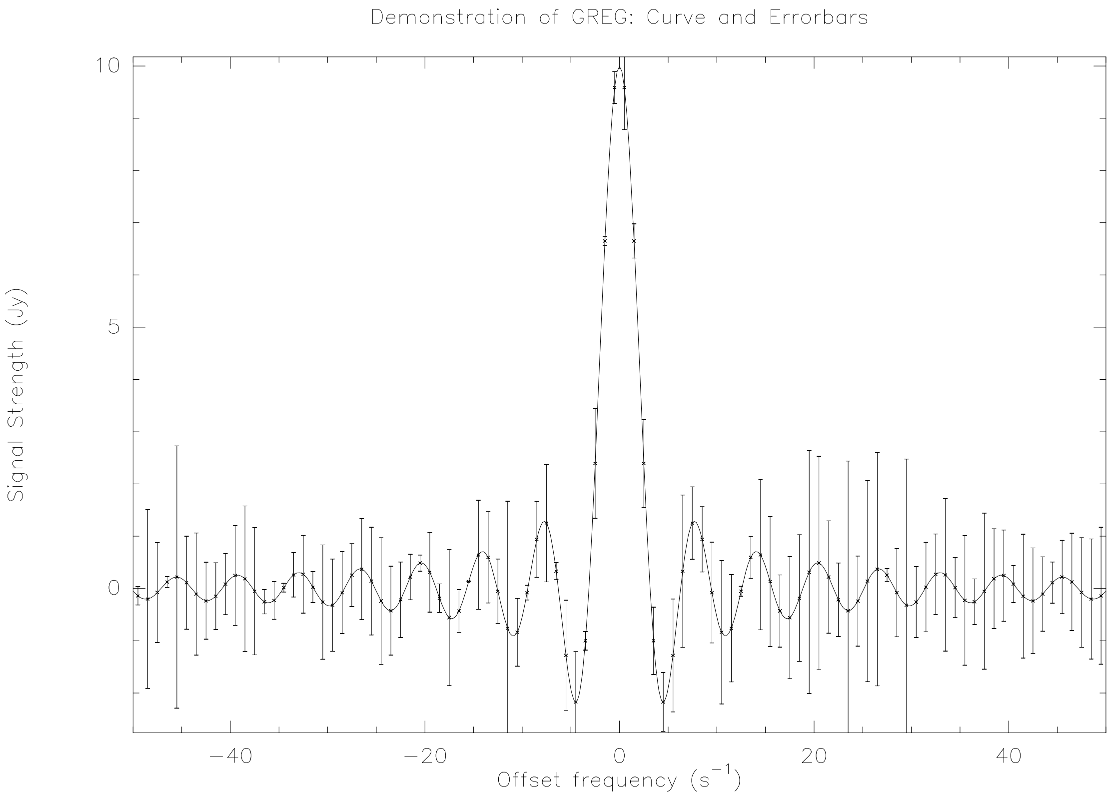

Figure 2: the printed output of the simple demonstration procedure.

You have now mastered most of the 1 dimensional capabilities. At this point, you may want to add some title or any other comment on your plot. Then you will need to get a hardcopy of your drawing and to send it to a plotter.

2.4 Annotations or Using the Cursor

Most additional annotations can be done using the cursor, which is called by command `DRAW`. In command `DRAW`, the user can specify coordinates in three available systems :

- The USER coordinate system, as specified above, which can be defined as the default system by command `SET COORDINATE USER`
- The BOX coordinate system. In this system, coordinates are offsets in physical units from one of the 9 most remarkable points in the box (the corners, the center and the middle of sides) numbered according to a standard numeric keypad notation as on VT100 terminals. This system can be specified using command `SET COORDINATE BOX N`, where N can take values 1-9 to specify the corner. The default value N=0 may also be used, it behaves differently when the command `DRAW` is used with the cursor or explicitly. When using the cursor in `DRAW`, the nearest remarkable point will automatically be used as the reference point. If you use the explicit form of command `DRAW`, 0 behaves as 1.
- The `CHARACTER` coordinate system, which is essentially the same as the `BOX` system except that offsets are specified in units of character size. This system can be specified with the

command `SET CHARACTER N`, where N has the same meaning and default value as in BOX. The character size is specified by the command `SET CHARACTER Size`.

By typing DRAW, the cursor appears on the screen. By hitting the appropriate key on your keyboard, you can obtain different actions :

- `R for RELOCATE` defines the current cursor position as the new pen position. Nothing visible happens because this is a pen up movement.
- `L for LINE` draws a line from the current pen position to the cursor position (pen down movement).
- `M for MARKER` draws a graphic marker of the current type and size centered at the cursor position.
- `A for ARROW` draws an arrow from the last pen position to the cursor position.
- `T for TEXT` prompts you for a string to be written at the cursor position (with the current centering option). Enter your text, then type `RETURN`, and the text will appear on the graphic screen.
- `C for ''Centered TEXT''` allows to override the current centering option. It prompts you for a string, then for the centering option you want to use.
- `E for EXIT` allows you to escape from this forever looping command.
- `D for DELETE` destroys the vectors drawn in the last operation. This is equivalent to `CLEAR SEGMENT`, except that only the segments created by the current `DRAW` command can be deleted in this way. Note that the plot is not refreshed. To refresh the plot, hit E to `EXIT` from the `DRAW` loop and use the command `ZOOM REFRESH`.

After each of the precedent action but `EXIT`, the cursor position becomes the current pen position. Anything else usually gives you the cursor coordinates. However some letters may be used to add new possibilities in command `DRAW`. Never press `RETURN` or `^Z` while the cursor is on, as this causes sometimes dramatic effects...

Note that by default, clipping within the box is turned off when you use command `DRAW`. Clipping can be enforced for the actions `LINE, ARROW` or `MARKER` by using the `/CLIP` option when you type the command. The action `TEXT` is never clipped.

When there is no cursor available (no graphic device active, or no cursor on the graphic device), to use command `DRAW` you must type explicitly the complete command as follows :

- `DRAW RELOCATE Xc Yc [/BOX N] [/CHARACTER N] [/USER] [/CLIP]` to relocate the pen at position (Xc,Yc). The coordinates is the coordinate system specified in the option, or the default coordinate system as defined by command `SET COORDINATE`.
- `DRAW LINE Xc Yc` to draw a line
- `DRAW ARROW Xc Yc` to draw an arrow
- `DRAW MARKER Xc Yc` to draw a marker

- `DRAW TEXT Xc Yc ''Text to be written'' I` to write the string "Text to be written" at (Xc,Yc) where I is the centering option used (this explicit form also corresponds to the code C used with the cursor).

This explicit form can also be used on interactive devices. The cursor will not be called in such case. There is no explicit form for D and E. In interactive mode, the explicit command corresponding to the cursor action is written to the Log File and to the internal stack. Accordingly, the stack can be replayed to produce the same results without any interaction with the cursor. The choice of the coordinate system is beyond the scope of this cookbook; for a single plot it should not matter.

2.5 Getting Hardcopies

2.5.1 The easy way

Now that you have obtained a wonderful plot on the screen, you may want to draw it on a paper sheet. As shown in the procedure, command `HARDCOPY /PLOT` prints out such a hardcopy on the default printer of your installation. That is the easy way, but it should have been configured properly by your system manager (or your local **GILDAS** expert) to work properly.

2.5.2 Printable (PostScript or others) Files

Instead of getting an immediate paper copy, you may prefer to create a file in some graphic format, such as PostScript, HPGL or others, for later printing or insertion in other documents.

To get a PostScript file from your **GreG** plot, use the following command

```
HARDCOPY plot.ps/DEVICE PS FAST
```

which create a `plot.ps` file containing a PostScript translation of the plot. The `/DEVICE PS` option indicates that the format of the file is PostScript, and the `FAST` argument indicates that line thickness and dash pattern should be generated by PostScript rather than by **GreG** (this produces smaller files, but at the expense of small clipping errors in case of very wide lines). Two other variants of the `PS` device exist: `GREY` for greyscale plots, and `COLOR` for color devices. Device EPS is similar to device PS, but the `BoundingBox` (in PostScript sense) is computed from the the plot boundaries rather than from the Plot Page size.

A `/DEVICE HPGL` option would have created an output file using the HP-GL command language. So far, PS and HPGL are the only supported output formats. Others (such as PCL for example) may come later.

2.5.3 Metacode File

Rather than preparing an output file in PostScript or HPGL, you can also produce it in **GreG** format directly. This is done using the `EXPORT` command. The file produced can be later reinserted in a **GreG** plot for further processing using the `IMPORT` command. These commands are described in more details in the section "Plot Structure"

2.5.4 Raster Devices

For raster devices (not supporting vectors), a somewhat different method is used to get a printout. First, a vector file (default extension .vec) is created, and it is then processed by another utility to produce a drawing. Several different utilities are provided to do so. Their use and availability will depend on your own site specificities.

- `PLXY` is used to provide a bit-file for `LXY11 Dec` printers (also called `Printronix 300`). This bit-file extension is `.PLT` and the name is the same as the input metacode file.
- `VTXY` is used to plot on Versatec.
- LA100 is used to create bit-file for an `LA100` Decwriter (extension `.LA100`).
- And others for other supports...

2.6 Internal Zooming

GreG has the possibility of zooming a plot being created. Use command `ZOOM` to do it. `ZOOM X1 X2 Y1 Y2` will compute scale factors to match the designed area (in Physical Coordinates) to the screen, preserving the aspect ratio. `ZOOM OFF` restores the full screen, `ZOOM REFRESH` redraws the plot with the current zooming factor. Without arguments `ZOOM` calls the cursor, and several keys can be used to control the zooming factor :

- "0" turns off zooming and displays the full plot page. The previous zooming factor is not destroyed, so that typing a space (see below) will restore a Viewing Window of same size as the last one centered on the cursor position. The cursor position is not modified by the command itself.
- " " (space) executes the standing zooming operations. With "0", it is the only effective command. The zoomed area is centered around the cursor position and can be delimited using command B.
- "Z" increases the zooming factor by 1.414 and draws the corresponding box around the zoomed area. The zooming is deferred until " " has been struck, allowing zooming factors to be changed by any power of square root of 2. The box drawn around the zoomed area is not a part of the current plot, and will not be visible in a Hardcopy for example.
- "-" decreases the zooming factor by 1.414, and draws the corresponding box if visible.
- "B" draws a box delimiting the area to be zoomed.
- "A" to erase the alphanumeric screen
- "E" to exit...

Any other gives you the cursor position. Zooming can be used for example to enlarge part of a complex drawing to make precise annotations. Note that the zooming has no effect on the hardcopy.

On X-Window systems, the zoomed area normally appears in another window. The mouse can be used instead of the keyboard for some actions:

- left button: as spacebar, execute the zoom
- middle button: increase zooming factor by 1.414
- right button: exit

3 GreG Manual

3.1 Basic Concepts

GreG uses a graphic library with a virtual memory storage for all normal plot actions. Accordingly, the "plot" is a notion entirely independent of any graphic device, and it can be generated without any. The only thing necessary to define a plot is the *Plot Page* size in a conventional unit called the Physical Unit. Plot coordinates on the Plot Page are called *Physical Coordinates*, or *World Coordinates*. By convention, the Physical Unit will be called a centimeter ; this is meaningful because when you produce a drawing from a metacode file using a specific driver such as `PLXY`, a Physical Unit will match a centimeter if you use the `/EXACT` option.

3.1.1 Coordinate Systems

GreG is often used (but far from being limited to) for plotting two dimensional graphs showing the dependence of one variable on another. A typical graph has data points, represented by special markers such as diamonds or stars, and possibly with error bars, or perhaps plotted on the same scale a theoretical model drawn as a smooth curve. The graph must be labelled with two axes to indicate the coordinates. The other major **GreG** application, namely contouring, also requires such a labelling of the plotting area.

The meaning of the coordinates is entirely defined by the user. For example in an astronomical map it may be Right Ascension Offset in arcsec along the X-axis and Declination Offset in arcsec along the Y-axis; a correlation study between visual extinction and molecular content will have the visual extinction Av as abscissa (no units) and ^{13}`CO` column density as ordinate (molecules per square centimeter).

Throughout this document, these coordinates will be referred as the *User Coordinates*. **GreG** maps a selected region of the User Coordinate Space onto a specified rectangle, called the *Box*, of the *Plot Page*.

All data plotting is done in User Coordinates and clipped in the Box, except for the perspective algorithm which uses a specific sequence. Other reference systems are available for annotations (see command `DRAW`).

3.1.2 The Plot Page

GreG is a completely device independent graphic system. This means that a sequence of plot actions will give exactly the same drawing on any device, except possibly for a global scale factor (contrary to many systems where the plot is distorted by some X/Y aspect ratio). It is then possible to define a unique Physical Coordinate unit. By convention in **GreG** this *Physical Unit* corresponds to 1 true centimeter on a paper output, if you use the `/EXACT` option in the metacode device driver. The Plot Page is defined in terms of such units.

A default Plot Page is defined at **GreG** initialization. It is an A4 format page (30 by 21 cm) with the `LANDSCAPE` orientation, i.e. with the largest dimension along the X coordinate. Plot pages of other dimensions can be defined using command

`SET PLOT_PAGE SizeX SizeY`

Another frequently used Plot Page corresponds to the A4 format with the `PORTRAIT` orientation (long dimension corresponding to Y) which can be accessed by command `SET PLOT_PAGE PORTRAIT`.

3.1.3 The Viewing Window

Only a part of the Plot Page need be displayed on the graphic device (if any). This part is called the *Viewing Window*, (not to be confused with the Box). This window can be modified at any time using the command `ZOOM`. When you define a new plot page, the window is automatically reset to the full plot page. Changing device does not modify the Viewing Window.

This window has no effect at all on the metacode file produced by command `HARDCOPY`, which always contain the full plot page.

3.1.4 The Box

The Box is defined by the position of its corner in Physical Coordinates in the Plot Page by command

`SET BOX_LOCATION Xmin Xmax Ymin Ymax`

The default Box for the default Plot Page can be accessed by typing `SET BOX LANDSCAPE` or `SET BOX PORTRAIT` as required. This default box is however usually inadequate for maps, where the User Coordinates in X and Y may be related. The box must of course stay within the limits of the PLOT_PAGE, so that attempts to define a PORTRAIT page in a LANDSCAPE page (or vice versa) will fail: an apropriate SET PLOT_PAGE should have been issued before the SET BOX command.

3.1.5 The User Coordinates

There is a single command to define the correspondence between the User Coordinates and the Physical Coordinates.

`LIMITS Xleft Xright Ylow Yup`

by which you specify the User Coordinates values at the left and right ends of the X-axis and low and up ends of the Y-axis. Note that Xleft may be greater than Xright. **GreG** stores all User Coordinates as Real*8 numbers to allow high precision labelling.

If you have no specific idea about the range of values spanned by your data, just type

`LIMITS`

GreG then automatically computes the extrema of the input data (read by COLUMN see section 3.2) and adds some reasonable margin to set the limits. You can mix automatic and fixed limits using the "wild" value "*"

`LIMITS * Xright * Yup`

will computes automatic limits for the left end of the X axis and low end of Y axis.

The previous examples set *Linear Conversion* formula between the User Coordinates and the Physical Coordinates. You can obtain a *Logarithmic Conversion* formula for the abscissa or the ordinate using the options `/XLOG` or `/YLOG` respectively. Note that, contrary to most systems, **GreG** includes a *true* logarithmic conversion : you do *NOT* have to provide the logarithm of your data. Also, error bars will be correctly plotted for the logarithmic conversion : all the necessary computations are done into **GreG** .

3.2 The Data Structure

GreG has space reserved into internal buffers for the user's data. Four buffers are available, those with conventional names X Y Z and the Regular Grid array. Buffers X and Y are one dimensional arrays used to store the X and Y data for graphs Y=f(X) (or pairs (X,Y) in some applications). Z is also a one dimensional array and is used for storing additional values such as error bars corresponding to the X (or Y) data. The Regular Grid array is a two dimensional array used to store regularly spaced data (maps) for two-dimensional applications like contouring. Additional buffers are used for specific applications, such as the Polygon buffer.

3.2.1 One-Dimensional Arrays X Y Z

A single command, `COLUMN` is used to specify the input file, select part of this input file, and load data in the X Y Z buffers.

Formatted Files : The input file must be organised in a Table-like way, with formatted numbers arranged in column and lines. **GreG** uses list-directed format to read this file, so that the numbers need only be separated by spaces, tabs or commas. However, a single complete logical line of the table must be contained in a logical record of the input file. Any column, and any range of lines, can then be accessed typing

`COLUMN X Nx Y Ny Z Nz [/FILE Infile] [/LINES Lmin Lmax]`

where Nx is the column number from which the X buffer is to be read (resp. Ny and Nz), Infile is the input file name, and Lmin Lmax define the range of lines to select. By default, the last connected file is used. Changing the input file resets Lmin and Lmax to select the complete file. The order of the X Y Z keywords is not compulsory, and some may be missing ; however each keyword must be immediately followed by its associate column number. There is an implementation dependent limit (usually 50000) on the number of lines that can be read by one COLUMN command in a formatted file. To go over that limit, you can convert the formatted file to a table (usually a good idea), or work by pieces using /LINES.

Table Format : Tables are ensembles of columns strictly equivalent to the formatted files. The only difference is that they are unformatted, and hence access time is typically 50 times faster... The number of lines is fixed, but the number of columns may be extended indefinitely. In fact, a table may be consider as a 2-D image (and vice-versa if you want) but does not require "axis" information... Tables can be accessed in command `COLUMN` using option `/TABLE` instead of `/FILE`. Using tables is recommended whenever a large number of rows or columns. Two programs can be used to convert from tables to formatted files (`GILDAS_RUN:LIST.EXE`) and vice versa (`GILDAS_RUN:TABLE.EXE`).

3.2.2 Two-Dimensional Regular Grid Array

For mapping purpose, one need to store RG=f(I,J) data into the Regular Grid array and to define a linear correspondence between the User Coordinate Space and the indices I and J, X=x(I) and Y=y(I). All this is done using command `RGDATA` to read a purposely formatted file. Contrary to the X Y Z buffers which are Real*8, the Regular Grid RG array is declared Real*4 as there is usually no meaning in defining contours with very high precision.

The Regular Grid array is loaded using command `GREG2\RGDATA Name [/VARIABLE]` where Name is a variable or the input file name if the `/VARIABLE` option is not present.

Variables and Images: The **SIC** image format is the equivalent for maps of the table format for columns. Images are regularly sampled 2-d, 3-d or 4-d data which require conversion formula for the coordinates along each axis. Data values are real (single precision) or double precision. Access time to **SIC** images is typically 20 times faster than the `RGDATA` format. **GreG** can read images using the **SIC** command `DEFINE IMAGE` (see the **SIC** manual for details).

Images are then available as multi-dimensional variables, whose names can be specified as argument to a `RGDATA/VARIABLE` command. Any information about projection, coordinate system, etc...associated with the image is given to **GreG** to define astronomically correct plots. The *GRAPHIC* program, a superset of **GreG**, also incorporates an `IMAGE` command for similar purpose. **GreG** can write Images, by saving the Regular Grid array as an image (see `WRITE IMAGE`). Header information (conversion formulae, projection information...) is written according to the current parameters defined by command `SET`.

RGDATA File Format : A somewhat obsolete way of initializing the RGDATA array is rge so-called "RGDATA file format". Suitably formatted files can be read by the `RGDATA` command when no `/VARIABLE` option is specified. A header is read first to find the array dimensions, and then the array, using a user-specified format. The default format is `Z8.8` and can be changed using the `/FORMAT Expression` option. . `Expression` must be a valid FORTRAN format. It is possible to select only a subset of the input array, using the option `/SUBSET IX1 IY1 IX2 IY2`, where IX1 and IY1 are the pixel values of the bottom left corner, IX2 IY2 those of the top right corner of the area to be selected.

The input file for the RG buffer must be a fixed length formatted sequential file with 80-Bytes record length. The first four records are used to describe the correspondence between indices and User Coordinates and must give the following values

- `Record 1 NX XREF XVAL XINC`
  describing the X axis Coordinates, corresponding to indice I, where
  - NX is the number of (I) pixels
  - XREF is the X-axis reference pixel. XREF is a real, that is one can place the reference pixel on non-integer values.
  - XVAL is the User Coordinate (abscissa) at the reference pixel
  - XINC is the User Coordinate increment per pixel along the X-axis, which can be negative.
- `Record 2 ''Any comment you want for the X axis'`
  is a comment line for bookkeeping. It may be empty but must be present. This comment will be written by **GreG** when it finds it (as an alphanumeric comment, not on the plot). It may help you remember the X coordinate type for example, or simply the file content.
- `Record 3 NY YREF YVAL YINC`
  similar to record 1, but for the Y axis Coordinate (indice J).
- `Record 4 ''Any comment you want for the Y axis''`
  similar to record 2.

All the following records contain the RG array, written in the standard Fortran ordering (I varies first) in format 20A4. The variables XREF, XVAL, XINC, YREF, YVAL and YINC are declared Real*8 to provide accurate conversion formulae. The conversion formulae are thus

```
X(I) = XINC*(I-XREF)+XVAL
Y(J) = YINC*(J-YREF)+YVAL
```

respectively for X and Y coordinates.

3.2.3 Using Variables

Instead of reading data into the X, Y, Z or Regular Grid buffers, it is possible to use directly **SIC** variables in some **GreG** commands. Currently, the following commands support **SIC** variables as extra arguments :

```
GREG1\CONNECT [Xv Yv]
GREG1\CURVE [Xv Yv] [/VARIABLE Z [Zv]]
GREG1\HISTOGRAM [Xv Yv]
GREG1\ERRORBAR Type [Xv Yv Zv [Orv]]
GREG1\POINTS [Xv Yv] [/SIZE maxsize [Zv]]
GREG1\VALUES [Xv Yv Zv]
```

where, if not specified, the Xv Yv Zv variables default to the X Y Z buffers, which are known as **SIC** variables of the same names.

SIC variables are defined using command `SIC\DEFINE`, and can be external tables or images. See **SIC** documentation, or use `HELP DEFINE` within **GreG** for more details.

The option `/VARIABLE` of command `RGDATA` indicates that the argument is the name of a known **SIC** variable, instead of a file name. This can be used to contour directly 2-D **SIC** variables. Moreover, the Regular Grid is known as a **SIC** variable named RG, hence it is possible to assign values to RG within **GreG** . This does not modify the pixel to user coordinates conversion formula.

3.3 The Pen Attributes

GreG can draw different style of lines : solid lines or dashed ones, thin or thick ones, dim or bright ones, and possibly coloured ones on devices supporting this possibility. All these capabilities are controlled by means of virtual pen attributes.

3.3.1 Pen definition

A virtual pen is defined by its conventional number (0 to 15), and its attributes. Currently, the attributes can be

- COLOUR, an integer code for the colour. The integer can take values from 0 to 15 with undefined conventions, because the colour table is dynamically selectable on some devices. (e.g. ARGS). On a non-color device with default terminal settings, it corresponds to an intensity level. 0 is the brightest available intensity. Values 1 to 15 correspond to increasing intensities, scaled to cover the full range of available intensities on the current device. As most devices have three intensity levels, 1-5 will give the dimmest intensity, 6-10 the intermediate one and 11-15 the brightest (i.e. the same as 0).

- DASHED pattern, an integer from 1 to 7 with the following significance

  1. Solid line
  2. Short dashed line
  3. Dotted line
  4. Short dash - Dot line pattern
  5. Long dashed line

6. Long dash - Dot
7. Short dash - Long dash

The first four attributes corresponds to the standard GKS (Graphic Kernel System) convention, and the hardware generation of the output device may be used. It is not used for the other ones.

- WEIGHT, an integer from 1 to 5 giving the thickening factor of the line. Hardware generation may be used.

A pen is loaded and defined by command
`PENCIL Ipen /DASHED Idash /WEIGHT Iweig /COLOUR IColo /DEFAULT`
All subsequent plots will be done with this virtual pen, until a new command `PENCIL` is issued. If the attributes are not given, the previous ones (resulting from the last definition) are used. If a device has not the required capability, the code is ignored. The `/DEFAULT` option can be used to reset the default settings of the attributes which are not specified. If no pen number is specified, the `/DEFAULT` option reset all default attributes to every pen.

3.3.2 Pen Usage and Default Settings

Pen number 0 is preset with no thickening, colour 0, solid line. Pen number 1,2,3 are preset with no thickening, colour 4,8,12 respectively (in order to match the three intensity levels available on most devices) and solid line. Pen number 15 is preset with no thickening, colour 0 dashed pattern number 2. All other pens have the same default attributes as Pen 0.

In general, a plot command uses the current pen with all its attributes. Exceptions are

- The dashed pattern is ignored by the character plotting routine (commands `LABEL` and `DRAW TEXT`)
- The dashed pattern is ignored by the marker plotting routine (commands `POINTS, DRAW MARKER` and `DRAW ARROW`)
- The dashed pattern is ignored by the axis plotting routine. (commands `BOX` and `AXIS`)
- Contour plots use Pen number 0 for positive contours and pen number 15 for negative ones (command `RGMAP`), unless specified in option. This is described in more detail later.

3.4 Characters and Markers

For labelling and annotations of the plots, **GreG** includes a complex character set with special symbols, and the possibility of plotting any regular polygon. The characters and markers are always drawn without the dashed pattern, as mentioned above.

3.4.1 The Character Set

GreG includes two basic character sets generated by software. The `SIMPLEX` set is the default for interactive devices. The `DUPLEX` set is nicer, but slower and requires a higher resolution device ; typically it will be reserved for quality publication plots. Except for the aspect of the characters, there is no difference between the `SIMPLEX` and `DUPLEX` fonts, that is the sizes and proportional spacings are identical. The default font may be overridden by `SET FONT SIMPLEX or DUPLEX`. In each quality, complete roman, script and greek alphabets are available. Special characters are used to access specific symbols not available on a keyboard, including mathematical and astronomical symbols. The table of correspondence is shown in Section 7. As a rule, astronomical symbols are obtained in the greek character set, and mathematical symbols in the script set, the roman set having of course an exact correspondence with the keyboard, except for the double quote (key not available because of **SIC** character handling) which is obtained using the counter-quote, degree symbol which is obtained using the circumflex symbol, and the back-slash which is not available.

3.4.2 Handling Text Strings

A text string (plotted by command `LABEL` or `DRAW TEXT`) can include the escape character "\" which is used to generate superscripts and subscripts and to monitor the font used. This escape character may be used in 2 ways

```
\\X - set mode X until other mode change sequence
\X  - set mode X for next character only
```

where X can be any of the following values and causes the following actions :

```
N - Switch to current default character set
1 - Switch to Simplex character set
2 - Switch to Duplex character set
R - Roman font
G - Greek font
S - Script font
I - Toggle italics
U - Increase superscript/subscript level (move Up)
D - Decrease superscript/subscript level (move Down)
B - Backspace over previous character
```

Superscript and subscripts are handled using a "level" philosophy. \U increases the level by 1, \D decreases it by 1 and the size of the characters is (0.7)**ABS(level) times the current defined size. The italic mode is toggled by the \I escape code. The character size is defined using command `SET CHARACTER`.

3.4.3 Centering Facility

Labels may be automatically positioned with respect to a given position. This is done using the Centering code, which can be specified in the `/CENTERING` option of command `LABEL`, as the 5-th argument of command `DRAW TEXT`, or defaulted to a value set by command `SET CENTERING` N. This code N is an integer in the range [0-9] with the following meaning :

- 1 to 9 : The character string is put in position 1 to 9 with respect to the current pen position (`LABEL`) or coordinates specified (`DRAW TEXT`). The position code corresponds to a standard numeric keypad as on most terminals, 1 being lower left, 2 lower middle, etc...
- 0 : For command `LABEL`, it corresponds to code 6, i.e. the label is just written on the right of the current position. For command `DRAW TEXT`, the centering is computed according to the location of the current position with respect to the Box. This location gives a code in range 1-9 with the same conventions as above. Hence a string written at the left of the box will be automatically left adjusted (code 4) while a string written inside the box will be centered and so on.

3.4.4 Graphic Markers

In addition to the character set which includes symbols, **GreG** can plot any regular polygon under 4 different styles of any size. The current type of graphic marker is set by command

```
SET MARKER Nsides Istyle Size
```

where Nsides is the number of sides (or of summits) of the regular polygon, Size is the diameter of the marker in physical units, and Istyle defines one of the four possible styles according to

- 0 - Regular empty polygon
- 1 - Vertex connection only. The sides of the polygon are not drawn, but the radii connecting the polygon center to the summits are plotted.
- 2 - Fancy starred marker. The star with Nsides summits is drawn. The inner radius of the star is half the outer radius.
- 3 - Filled regular polygon. The filling may be a rather slow process on some devices. Also the filling may fail for very high resolution supports (e.g. a pen plotter with a very thin pencil). If this happens, please notify the authors of **GreG** .

The number of sides is not limited, but the larger this number, the slower the plot will be...

3.5 GREG1 : One Dimensional Problems

By "one dimensional data" we mean any distribution of (X,Y) data points. This obviously includes functions Y=f(X) and X=f(Y), but also more general cases. Plotting such kind of data is the simplest thing **GreG** can do. This can be done basically in four modes :

1. The broken line mode using command `CONNECT`, in which data points are simply connected by straight lines. Note that data must be ordered.
2. The curve mode using command `CURVE`, in which the data is interpolated to produce a smooth aspect. Again, the data must be adequately sorted.
3. The histogram mode using command `HISTOGRAM`. As above, data must be ordered. Note that this capability is restricted to functions Y=f(X).
4. The marker mode using command `POINTS`, in which a graphic marker is plotted at each data point. The size of the markers may depend on values in the Z buffer. Contrary to the other commands, the data need not be ordered at all.

In addition to these four commands, the command `ERRORBAR` which uses the Z buffer to define errors may also be relevant to one-dimensional data. A facility to sort data by ascending order is given by means of command `SORT`.

All these commands belong to the so-called "GREG1" language (in the **SIC** meaning) ; this language also includes all the utility functions of **GreG** such as pen definitions, annotations, coordinate conversions, etc...

3.6 GREG2 : Two Dimensional Problems

We call "two dimensional data" any data represented by a function f(X,Y). In practice, this representation may be regularly or randomly sampled. The first case corresponds to the Regular Grid array for which RG=f(I,J) and linear conversion formulae exist to convert between I and X and J and Y respectively. The second corresponds to triplets (X,Y,Z) stored in the corresponding buffers. Basic handling of 2-D data includes contouring, perspective view, bitmap processing, and extraction of one dimensional data.

Only regularly sampled 2-D data (i.e. a Regular Grid) can be conveniently handled. The limits of a Regular Grid can be defined as the plot limits using command `LIMITS /RGDATA`. In this case, **GreG** computes the user coordinates corresponding to pixels 0.5 and NX+0.5 in X (same in Y) and defines these values as the user coordinate limits. This means that the Box will exactly cover the *Image* corresponding to the given map.

For randomly sampled data, **GreG** provides a way of interpolating them on a Regular Grid by command `RANDOM_MAP`. All other commands are provided to work exclusively on the Regular Grid array. This includes :

- A general contouring facility using command `RGMAP`
- A simple perspective view using command `PERSPECTIVE`
- A bitmap display capability for X-Window terminals and PostScript output, command `PLOT`
- A resampling facility either to produce finer contours, or to produce maps with the same sampling, and hence comparable pixel by pixel.
- Limited map analysis facilities : pixel value interrogation with command `DRAW VALUE`, statistics using command `MEAN`, masking part of a map using command `MASK`, computations of global and local extrema using command `EXTREMA`, strip extraction using command `STRIP`. For more general image processing facilities, the user is referred to the *GILDAS* documentation.

All these commands belong to the "GREG2" language.

3.7 Astronomical Mapping

"No matter how tricky you are, you cannot pave a sphere with square tiles."
(Attributed to A. Einstein)

About 101% of the difficulties encountered in astronomical mapping comes from this very profound evidence. The remaining few percents (within the errors) come from the multiplicity of coordinate systems used by the astronomers. For small field mapping, the sphere can reasonably be approximated by its tangent plane at the field center, but for fields larger than a few degrees (depending on the required positional accuracy), curvature effects become important.

As spherical plotters are not easily commercially available (and their output hardly accepted by journal editors), the big problem is to represent part of a sphere on a plane sheet. **GreG** offers several facilities to deal with this problem, by means of commands `PROJECTION`, `CONVERT` and `GRID` in the `GREG2` language.

The current philosophy to handle this problem is to make all plotting *in relative coordinates on the projection plane.* No drawing is done in absolute coordinates on the sphere. This was decided because the plot page is plane indeed. Another apparently restrictive convention in **GreG** projections is that all angles are internally in the natural angular unit Radian. Accordingly, when you give a map to be contoured, the map coordinates should be offsets in radians from the projection center. For user convenience, it is possible to specify limits in Degrees, Minutes or Seconds using the command `SET ANGLE_UNIT`, but this command has no effect on internal conversion formulae. Again, the rationale behind this convention is that we are in fact working in the projection plane, where angles have no meaning but only their projections remain.

However, it is *always possible to bypass* these *apparent* restrictions provided your understand what a conversion formula and a projection are. As an example is always much better than lengthy discussion, assume you want to overlay a contour map to the Equatorial and Galactic grids relevant to the mapped area. Unfortunately, the map coordinates are in Degrees from a point situated at (0.5,0.5) ("degrees" in projection) from the projection center (note that this sentence is complete nonsense, because you cannot have "degrees" in the projection plane...)

```
SET ANGLE_UNIT/DEFAULT                          ! (1)
LIMITS 5 -5 -5 5                                ! (2)
RGDATA MYMAP                                    ! (3)
LEVELS -1 1 TO 5                                ! (3)
RGMAP/BLANKING -2 .1                            ! (3)
!
SET SYSTEM EQUATORIAL                           ! (4)
PROJECTION 6:25:30 35:40.5 0.0/TYPE GNOMONIC    ! (5)
SET ANGLE_UNIT DEGREES                          ! (6)
LIMITS 4.5 -5.5 -5.5 4.5                        ! (7)
SHOW LIMITS                                     ! (7)
PEN 1                                           !
GRID 2 2                                        ! (8)
PEN 2                                           !
GRID 2 2/ALTERNATE                              ! (9)
```

1. Make sure to work as if you had no problem at all
2. Define the map limits. As you are not worrying about the angular units, the internal limits will be just what you type.
3. Read your map and draw your contours. This is the usual process for any map.
4. Here start the specific astronomical problem. You know that your coordinate system is Equatorial. Specify it.
5. Define the projection in this system. Note that Right Ascension is in Hours, and Declination in Degrees as usual for Equatorial coordinates. The projection type depends on your problem of course.
6. Specify that your will *now be giving limits in Degrees*
7. Give the limits of the map with respect to the projection center. The system automatically converts the values typed in to internal limits in radians, as you can check by the `SHOW` command. Note how you handle the shift between the (0,0) of your map and the projection center which is always the (0,0) of the projection plane (compare with command (2))

8. Everything is now correct to plot the grid of meridians and parallels over your map.

9. You may even plot a *Galactic* grid just by specifying the `/ALTERNATE` option in the `GRID` command.

As shown in this example, the *only* effect of the `SET ANGLE_UNIT` command is to force automatic conversion of the typed limits to radians. The same result would have been obtained by typing

```
LIMITS 4.5 -5.5 -5.5 4.5 DEGREE
```

instead of commands (6) and (7).

You can work in three different systems :

- `UNKNOWN`, for which **GreG** makes no assumption at all about the meaning of the unprojected coordinates. In particular, there is no alternate system in this case.

- `EQUATORIAL`, which **GreG** assumes to be at epoch 1950.0 . The alternate system is `GALACTIC` of course.

- `GALACTIC`, with `EQUATORIAL 1950.0` system as alternate.

When you measure positions with the cursor in the last two systems, both galactic and equatorial coordinates are given. For user convenience, sexagesimal notations in Hours Minutes Seconds for Right Ascension and Degrees, Minutes Seconds for Declination are used, while decimal notations with angle in degrees are used for L and B (as well as if the system is `UNKNOWN`).

Approximate absolute labelling in Hours and Degrees (sexagesimal notation) can be obtained for the Box by specifying the option `/ABSOLUTE` to command `BOX` if the system is `EQUATORIAL`. If the system is `GALACTIC` or `UNKNOWN`, this option produces absolute labelling in Degrees. Note that it is only meaningful for small fields of view. Without this option, `BOX` produces relative labelling in current angle units.

In addition, command `CONVERT` can convert absolute positions or projected coordinates from a different projection (in any angular units), to projected coordinates (in radians) in the current projection. This command may be used to plot star positions on a map, and so on.

3.8 The Blanking Capability

Experimental data are seldom completely sampled. For example, a spectrometer may have a bad channel, an image bad pixels and so on. To handle this problem, **GreG** includes the notion of *Blanking Value*. Most **GreG** data functions handle this blanking value which is defined using command

```
SET BLANKING Bval Eval
```

where Bval is the blanking value and Eval the tolerance for blanked data, necessary because of limited accuracy. Once a blanking value has been defined, all functions liable to handle it effectively use it. The default value can be overridden using the appropriate option `/BLANKING`. However some functions which should do it still do not handle blanked values ; this currently includes `PERSPECTIVE`.

To turn off the blanking value checking, you must specify a negative value for Eval, or simply use the command `SET BLANKING/DEFAULT`.

The blanking value can be used in two different ways :

1. Avoiding bad points in a curve. In this case it is assumed that points with the Y value blanked are meaningless. Note that this feature allows you to plot a complete set of curves in a single operation : you just need to specify a blanked point to separate each curve.

2. Avoiding bad pixels in an image. In this case, pixels with the blanked value are not considered (for example in command `MEAN`)

One command considers both aspects at the same time : this is command `RGMAP/KEEP Blanking_bis`, for which the default blanking is used to check bad pixels and the Blanking_bis value to force pen up between contours (or parts of a single contour).

4 Advanced Users Guide

Most of the features described in this section (with the exception of the PLOT and RGMAP/GREY commands), are available through commands of the **GTVL** language, which is the command language supporting the facilities offered by the graphic library on which **GreG** is build. This library may also be accessed directly through other programs.

4.1 Greyscale and Color plots

GreG now has the ability to display color or greyscale images, overlaid with contours if needed. **GreG** also has area filling capabilities. These possibilities are best exploited on X-Window terminals and PostScript printers. They are usually not available on any other devices.

4.1.1 Bitmap display of images

The PLOT creates a bitmap display of a 2-D images, very much like the command RGMAP draws contour maps. As all other commands in the **GreG** software, the PLOT command essentially works in user coordinates. The PLOT command is essentially used in the following way:

```
    Greg> SET BOX LOCATION Gx1 Gx2 Gy1 Gy2  ! Define box position
    Greg> RGDATA Varname /VARIABLE          ! Get characteristics from image.
    Greg> LIMITS/RGDATA                     ! Get limits
    Greg> PLOT Varname                      ! Plot the array in the box
! And then other commands
    Greg> BOX
    Greg> RGMAP                             ! Draw contour levels (...)
```

which shows the similarity between the PLOT and RGMAP commands.

The PLOT command has several options, among which the most important is the /SCALING Type Low_cut High_cut option. Type indicates which type of transfer function is desired between the image and the bitmap, and can be either LINEAR or LOGARITHMIC. Low_cut and High_cut are respectively the smallest and largest values plotted. Values below (above) are truncated to Low_cut (resp. High_cut.

Blanking values are recognized by the PLOT command. The values used for blanking are derived from the variable characteristics, or if not set, by the current **GreG** blanking, or from the /BLANKING option.

The PLOT command also has less frequently used options. See the internal help for details. The sample procedure below produces figure 3:

```
SIC\DEFINE IMAGE A GAG_DEMO:SMALL READ
GREG1\LIMITS/RGDATA A
GREG1\SET BOX_LOCATION MATCH
GREG2\PLOT
GREG1\BOX/ABSOLUTE
GREG1\LABEL "Right Ascension"/X
GREG1\SET ORIENTATION 90
GREG1\DRAW TEXT 1.000 .0000E+00 "Declination" 5/CHARACTER 6
GREG1\SET ORIENTATION 0
GREG1\DRAW TEXT .0000E+00 1.000 "Example of PLOT command" 2/BOX 8
GTVL\HARDCOPY FIG3.PS/DEVICE PS GREY
```

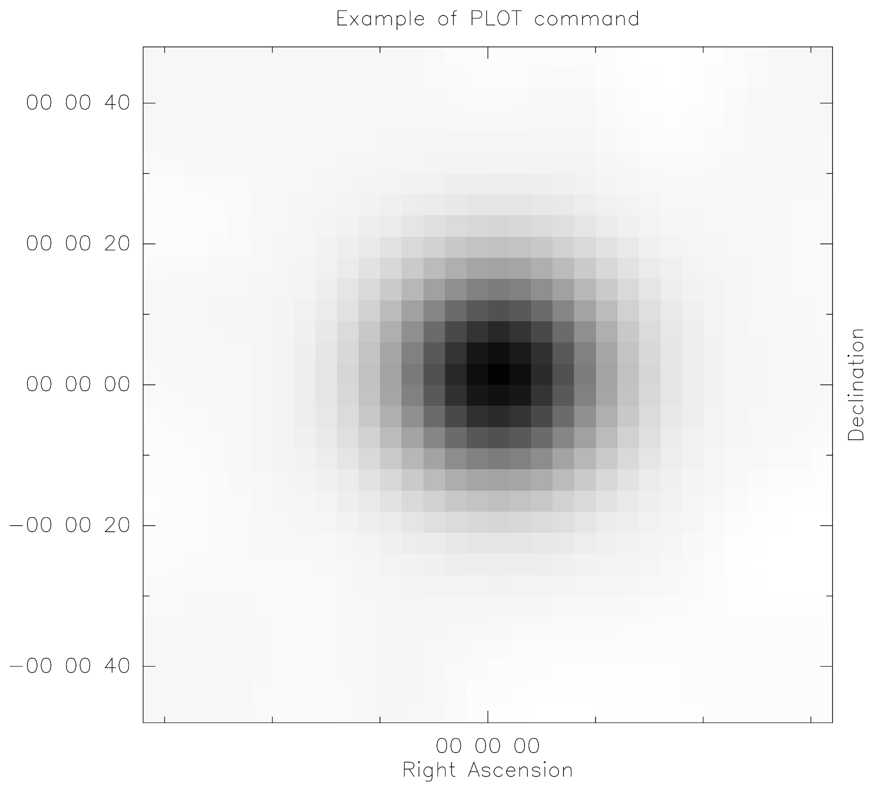

Figure 3: An example of bitmap display in **GreG**

4.1.2 Filled Areas

Connected to the bitmap capability, **GreG** can also produce filled curves, histograms or pie charts. The option `/FILL` of commands `CONNECT` `HISTOGRAM` and `ELLIPSE` are used for hat.

- `CONNECT/FILL Colour`
  closes the polygon and fills it with the current or specified Colour. This is incompatible with blanking values.

- `HISTOGRAM/FILL Colour`
  fills the histogram with the current or specified colour. It is intended to be used in conjunction with the `/BASE` option.

- `ELLIPSE/FILL Colour`
  fills the ellipse with the current or specified Colour. If the `/ARC` option is also present, the corresponding "part of the pie" is filled rather than the subtended arc.

So far, no filling pattern is available.

Contours can also be filled automatically, using command `RGMAP /GREY`. The `RGMAP/GREY` command does not use the current pen, but scans pencil colours from 8 to 24, changing the pencil colour by 1 for each new contour. Colours can later be edited using the look-up table facilities provided by **GreG** (see below).

Finally, the polygon defined by command `POLYGON` can be filled using the `POLYGON/FILL Colour` command, where `Colour` is a pen color (0 through 23).

4.1.3 Bitmap Color Editing

GreG offers the possibility of editing the color look-up table (LUT) for bitmaps. The colors are represented by three **SIC** variables:

- `HUE`
  The `HUE` array, with values between 0 and 360, represent the color on a standard color cycle (from Red to Yellow, Green, Blue, Violet and Red again).
- `SATURATION` The `SATURATION` array, with values between 0 and 1, indicates the fraction of the original (saturated) color mixed with white. 0 is pure white, 1 a purely saturated color.
- `VALUE` The `VALUE` array, with values between 0 and 1, indicates the intensity. 0 is black, 1 is maximum lightness.

These variable can be modified by any standard **SIC** expression. The new values get transferred to the display by typing the command `LUT`. This command also accepts other arguments:

- `COLOR` Load a default color table
- `WHITE` Load a black and white color table (White background, black sources)
- `BLACK` Load a black and white color table (Black background, white sources).
- `LUT` Load the color table defined by the arrays. Same as without any argument.
- `EDIT` Calls an interactive look-up table editor (on X-Window systems onlyt) *not yet implemented.*
- `Any other` Red Green Blue values are read from the formatted file specified as argument. The RGB representation is converted to HSV before being used.

The color associated with the blanking value can be modified using the `B_HUE, B_SATURATION` and `B_VALUE` scalar variables.

4.1.4 Pen Colors

GreG uses 24 colors for the pencils. Colors 0-7 are fixed colors, with the following values:

```
0 Black or White  (depending on window background color)
1 Red
2 Green
3 Blue
4 Cyan
5 Yellow
6 Magenta
7 White or Black  (the window background color)
```

Colors 8-23 can be modified by the user through the `P_HUE, P_SATURATION, P_VALUE` arrays and `LUT /PEN` command, in the same way as the bitmap colors. The default values for these arrays correspond to grey intensities from white to black.

4.1.5 Devices and Hardcopies

The color facilities are available only under X-Window systems. Moreover, not all X-Window like devices recognized by **GreG** effectively use the color for bitmaps. Only the `DEVICE IMAGE` does. This restriction exists because the color map allocate by **GreG** is a private color map. As a consequence, no more than a single **GreG** with `DEVICE IMAGE` can operate at any one time on an X-Window terminal. The pencil colors (pencils from 8 to 23) can always be edited. The number of bitmap colors is usually 128, but can be less if many color applications already use most of the available colors of the X-Window terminal.

For hardcopies, color is supported only the PostScript format. Two options exists for the type of PostSsript created: `GREY` or `COLOR`. The `GREY` version, used by `HARDCOPY/DEVICE PS GREY`, has a fixed transfer function of greyscales. The transfer function is non-linear, and reasonably suited for most astronomical images. The `LUT` command has absolutely no influence on this type of HARDCOPY. The `COLOR` version, used by `HARDCOPY/DEVICE PS COLOR`, uses the current `LUT` to produce the PostScript output. On Greyscale printers, the colors are converted to grey intensities using standard formulas (what you would see by replacing a color display by a monochrome one). Colors are used on color printers.

4.2 Plot Architecture and Multi-Window Capabilities

4.2.1 Directories

GreG incorporates a fairly sophisticated plot architecture, which allows the user to organize its plot in logical elements. The organisation is like a directory tree, with graphic segments attached to a "directory". Directories as a whole (and all their subdirectories) can be manipulated as single entities. While this organisation is mostly transparent for simple plots, it can help organize more complex drawings, specially when the user uses the capability to save subtrees and reincorporate them later in a plot.

One or more graphic segments are created for each **GreG** command. The graphic segment is the smallest entity recognized in the plot. The directory architecture is under user-control, although some applications layered on **GreG** may automatically create their own plot organisation. Directories can be created using command `CREATE DIRECTORY`. The top directory is named `'<'`, and cannot be accessed by the user (you cannot type `CD <`). Directories attached to the top directory are named "main directories". To create a main directory, use the following syntax:

```
CREATE DIRECTORY <MAIN
```

where `MAIN` is the name of the directory to be created. The command

```
CREATE DIRECTORY SUB
```

creates a subdirectory from the current working directory, for example `<MAIN<SUB`.

The user can move across the directory structure using command `CHANGE DIRECTORY` (abbreviation `CD`), print his current working directory using command `DISPLAY DIRECTORY` (abbreviation `PWD`). Newly created graphic segments are attached to the current working directory. A whole directory can be erased using command `CLEAR DIRECTORY`, saved on a metacode file for later use by command `EXPORT`.

Directories and segments have attributes (pen colour and thickness, transfer function for images, visibility and depth) which can be edited using command `CHANGE`. Directories and segment can be made invisible by the `CHANGE VISIBILITY` command; this is a reversible action until the `COMPRESS` command has been used to effectively destroy all invisible segments and directories.

Each directory may have its own coordinate system attached. When created, subdirectories inherit the coordinate system from their parent directory. The coordinate system is automatically modified by the `SET BOX` and `LIMITS` commands typed within the directory. This feature allows to maintain several coordinate systems in a plot. However, the user must then realize that user coordinates may change when changing from one directory to another, as for example:

```
LIMIT 0 1 0 1
CREATE DIRECTORY NEW
CHANGE DIRECTORY NEW
LIMIT 0 2 0 2
DRAW RELOCATE 0.5 0.5/USER
CHANGE DIRECTORY ..
DRAW      ! User coordinates are now (0.25, 0.25)
```

4.2.2 Windows

Each directory may have one or several X-Window windows attached to it to display its content. Each window has its own magnification factor. Windows are created by command `CREATE WINDOW`, which create a new window for the current directory. They can be deleted by command `CLEAR WINDOW`.

The combination of multi-window capabilities with directory structure allows the user to save a plot in one window, and create a new version of it in another window for comparison. Each window has its own color table, although because of terminal limitations, only one color table is active at a time.

4.2.3 The `CLEAR` command

Because of the multi-window and directory capabilities, the `CLEAR` command has many possible arguments.

- `CLEAR ALPHA` erases the alphanumeric screen alone, without affecting the plot. If possible, rather than erasing the alpha screen, the graphic screen (current window) is raised up.
- `CLEAR DIRECTORY [DirName]` makes the named directory (or by default the current directory) invisible. All the attached structure becomes invisible. The Graphic screen is not updated, use command `ZOOM REFRESH` to do it.
- `CLEAR GRAPHIC` lowers the graphic windows, popping up the alpha screen.
- `CLEAR PLOT` is equivalent to either `CLEAR TREE` or `CLEAR WHOLE`, depending on user preference as expressed in command `CHANGE CLEAR`.
- `CLEAR SEGMENT` destroys the last graphic segment, i.e. the part of the plot corresponding to the last graphic command. Can be used repetitively. The Graphic screen is not updated, use command ZOOM REFRESH to do it.
- `CLEAR SEGMENT SegName` makes the named segment invisible. The Graphic screen is not updated, use command `ZOOM REFRESH` to do it. This is equivalent to `CHANGE VISIBILITY NameSeg OFF`.
- `CLEAR TREE` deletes the current directory, that is all subdirectories descending from the current main directory. Other parallel trees (if any) are not destroyed.

- `CLEAR WHOLE` deletes the whole plot, all main directories and their subdirectories. A new (empty) main directory is automatically recreated with one associated window.
- `CLEAR WINDOW` delete the current graphic window. Beware that this may leave the plot with no associated window.

4.2.4 Plotting Depth

Another feature which is essential with the bitmap and area filling capabilities is the *depth* of the graphic segment. Because bitmaps or filled areas are opaque, it is important to know in which order they should be displayed. **GreG** uses the *depth* attribute of the graphic segment to do so. *Depths* range between 1 and infinity. Depth 1 is the top layer: graphic segments at depth 1 are in front of any graphic segment at other depths. Depths 2 to infinity are plotted in this order. This means that segment at depth 2 are behind any other segment, segments at depth 3 are behind segments at depth 4, etc... Whenever segments have the same depth, the result may be unpredictable; however, normally most recently created segment appears in front of older ones.

To save unecessary refresh of the drawing, the depth attribute is not used when a segment is created: the segment appears on top of the existing ones at this time. The depth attributes becomes effective after a `ZOOM` action, or for hardcopies. **GreG** incorporates a default handling of depths when segment are created in the following way:

- Normal (vector) segments are created with depth = 1
- Bitmaps are created with depth = 2
- Filled areas are created with depth = 1

This allow bitmap images to be plotted in the background, with all standard graphic overlaid on top of them without user intervention.

This behaviour is often sufficient. When not, the user can modify the depth of any graphic segment using the command `CHANGE DEPTH`. See the internal help for details.

5 GREG1 Language Internal Help

We give here a *fac simile* of the internal `HELP` file for `GREG1` (one dimensional plots). In case of trouble, the user should refer directly to the internal help which is always accurately kept up to date.

5.1 AXIS

```
    [GREG1\]AXIS NAME [A1 A2] [/TICK Orien Small  Big]  [/LOCATION  X  Y
Len] [/LABEL Value [ONLY]] [/[NO]LOG] [/ABSOLUTE] [/UNIT Valid_Unit]

Makes an axis labelled from A1 to A2 at location X, Y, length Len.   The
current ORIENTATION determines the angle of the axis.

The axis  scale  is  LINEAR  or  LOGARITHMIC  according  to  the  option
/[NO]LOG.

The tick orientation "Orien" may be IN or OUT or NONE  with  respect  to
the  BOX.  Tick spacing can be defined by Small and Big. The orientation
NONE cancels both ticks and labels. Labels for other "Orien"  are  drawn
according  to  "Value",  which may be N[one], P[arallel] or O[rthogonal]
(to the axis). If option /LABEL has the second argument ONLY,  only  the
labels of the axis are written, not the axis nor the ticks. This feature
allows use of different pens for the labels and the axis.

AXIS provides default values for all these parameters according  to  the
axis NAME (XLow, XUp, YLeft, YRight) and the current values used by BOX.

AXIS creates a graphic segment named "AXIS".
```

5.1.1 AXIS /ABSOLUTE

```
    The option /ABSOLUTE is used to write (RA,DEC) labels for EQUATORIAL
system  or  (L,B)  coordinates  for GALACTIC system when a projection is
active, i.e., both a PROJECTION and a SYSTEM are defined.  RA (or L)  is
used  for  X-like  axis,  and DEC (or B) for Y-like axis. Note that this
should be used only for small fields (size less than 0.2  Radians),  and
must  be  used  only  with  the  axes  at  the default position (i.e. no
/LOCATION option). When SYSTEM is defined but the projection is NONE  or
has  never  been  defined,  the  /ABSOLUTE  option supports large fields
(since PROJECTION NONE makes a non-conformal mapping  anyway).  The  SET
SEXAGESIMAL  command (see help) permits some mastering on how UNKNOWN or
GALACTIC ABSOLUTE labels are written.

See HELP TICKSPACE for more  information  on  how  to  master  the  Tick
spacing with the /ABSOLUTE or the /UNIT options.
```

5.1.2 AXIS /UNIT

The option /UNIT is used to convert the axis coordinates to the desired unit, M for arc minutes, S for arc seconds, R for Radians. This option is valid only if a PROJECTION and a SYSTEM (EQUATORIAL or GALACTIC) are both defined. See HELP TICKSPACE for more information on how to master the Tick spacing with the /ABSOLUTE or the /UNIT options.

5.2 BOX

```
[GREG1\]BOX [Arg1  [Arg2  [Arg3]]]  [/ABSOLUTE]  [/UNIT  valid_unit]
[/LABEL Pen] [/FILL pen0m]
```

BOX puts axes around the plot region, labelling the lower and left. The first two arguments are used to modify the labelling of the lower and left axes respectively. They may take the values P for Parallel labels (default for X axis), O for Orthogonal labels (default for Y axis) or N for No labels. A third argument can be specified to indicate that the ticks are to be IN or OUT of the box, or not drawn at all (NONE). BOX creates a segment named "BOX".

The option /LABEL is used to specify a different pen to use for axis labelling. Creates a supplemetary graphic segment "LABEL".

The option "/FILL pen_num" will fill the inside of the box with the color of the pen pen_num. The last 16 pens (numbered 8-23) can receive arbitrary colors with the command LUT /PEN. Creates a supplementary graphic segment "FILL".

BOX N N N makes a rectangle with the current pen, even if the pen is dashed.

5.2.1 BOX /ABSOLUTE

The option /ABSOLUTE is used to write (RA,DEC) labels for EQUATORIAL system or (L,B) coordinates for GALACTIC system when a projection is active, i.e., both a PROJECTION and a SYSTEM are defined. RA (or L) is used for X-like axis, and DEC (or B) for Y-like axis. Note that this should be used only for small fields (size less than 0.2 Radians), and must be used only with the axes at the default position (i.e. no /LOCATION option). When SYSTEM is defined but the projection is NONE or has never been defined, the /ABSOLUTE option supports large fields (since PROJECTION NONE makes a non-conformal mapping anyway). The SET SEXAGESIMAL command (see help) permits some mastering on how UNKNOWN or GALACTIC ABSOLUTE labels are written.

See HELP TICKSPACE for more information on how to master the Tick spacing with the /ABSOLUTE or the /UNIT options.

5.2.2 BOX /UNIT

The option /UNIT is used to convert the axis coordinates to the desired unit, M for arc minutes, S for arc seconds, R for Radians. This option is valid only if a PROJECTION and a SYSTEM (EQUATORIAL or GALACTIC) are both defined. See HELP TICKSPACE for more information on how to master the Tick spacing with the /ABSOLUTE or the /UNIT options.

5.3 COLUMN

```
[GREG1\]COLUMN [X Nx] [Y Ny] [Z Nz] [/FILE File_Name] [/LINES L1 L2]
[/TABLE Table_Name] [/COMMENT "Separator"]
```

This command will read X data from column Nx of the current file, Y data from column Ny and Z data from column Nz of the current (or specified) input file or binary table. Any combination of any of three X,Y,Z items is valid in any order, provided the column number immediately follows the column descriptor. The X and Y arrays contain respectively the X and Y coordinates of points to be plotted. The Z array is an additional buffer which is used for ERRORBARS, for RANDOM maps and so on.

5.3.1 COLUMN /FILE

```
[GREG1\]COLUMN /FILE File_Name
```

Indicates in which formatted (list-directed free format) file the data are to be read. If not given, the last input file or table is used. There is an implementation dependent limit (usually 50000) on the number of lines that can be read by one COLUMN command in a formatted file. To go over that limit, you can convert the formatted file to a table (recommanded, for efficiency), or work by pieces using /LINES.

5.3.2 COLUMN /LINES

```
[GREG1\]COLUMN /LINES L1 L2
```

limits the range of lines read in the input file or table. It is useful to avoid non-data lines, or to read by pieces a formatted file which is too long for the column buffers. L1 and L2 are reinitialized for each new input file or table. L2 defaults to end of file if not specified.

5.3.3 COLUMN /TABLE

```
[GREG1\]COLUMN /TABLE Table_Name
```

```
Indicates that the columns are to be read in a  "Table"  at  the  GILDAS
format.  If  not  given,  the  last input file or table is used. "Table"
format is typically 100 times more efficient  used  than  the  formatted
file format.
```

5.3.4 COLUMN /COMMENT

```
[GREG1\]COLUMN /COMMENT "Separator"
```

```
Specify which character is used to indicate the beginning of  a  comment
line  in  the  input  file.  "Separator"  is a single character; typical
values  are  "!",  ";",  "#",  although  any  single  character  can  be
specified. The default is set by command SET COMMENT (default "!").
```

5.4 CONNECT

```
[GREG1\]CONNECT [Array_X Array_Y] [/BLANKING Bval Eval] [/FILL]
```

```
CONNECT draws line segments connecting the coordinates read by COLUMN  X
and  Y,  or stored in the variables Array_X Array_Y.  The current pencil
with its dashed pattern is used.

CONNECT uses the current blanking values, which can be overridden by the
/BLANKING  option.  Hence,  it does not connect with data points so that
abs(Y(i)-Bval) <= Eval. Eval negative means no blanking value, and  Bval
and Eval values default to those specified in SET BLANKING.
```

5.4.1 CONNECT /FILL

```
[GREG1\]CONNECT [Array_X Array_Y] [/FILL]
```

```
The /FILL option indicates to  fill  the  connected  line.  Blanking  is
irrelevant.  The  connected line is closed by linking together the first
and last point, then filled with the colour of the current pen. Use with
caution on complex shapes.
```

5.5 CURVE

```
[GREG1\]CURVE  [Array_X  Array_Y]  [/ACCURACY  A]  [/VARIABLE  V
[Array_V]] [/PERIODIC] [/BLANKING Bval Eval]
```

```
Builds the spline interpolation of the (x,y) pairs read  by  COLUMN,  or
stored  in  the  variables Array_X Array_Y, using the selected algorithm
and options. The current pen with its dashed pattern is used.  Presently
the  only  algorithm  available is CUBIC_SPLINE. By default, the command
assumes that a curve Y=f(X) is to be plotted ; this can be changed using
options to specify the kind of curve to be plotted.
```

5.5.1 CURVE /ACCURACY

```
  [GREG1\]CURVE /ACCURACY A

Controls the accuracy of the spline interpolation. The argument  is  the
desired accuracy (in paper units, i.e. centimeters) and should be set to
the plotter resolution. The default  value  is  defined  using  the  SET
ACCURACY command
```

5.5.2 CURVE /VARIABLE

```
  [GREG1\]CURVE /VARIABLE V [Array_V]

Selects the the interpolation variable V, which may be
  - X                 To plot a function Y=F(X) (Default)
  - Y                 To plot a function X=F(Y)
  - Z  [Array_V]      To plot a curve with an  explicit  parametrization
    using  the  Z array, or the array Array_V if this second argument is
    present.
  - POLYGONAL_LENGTH  To plot a curve using the polygonal length of  the
    curve as parametrization. It is fast and of general use.
  - CURVILINEAR_LENGTH Refined and slow version of  the  parametrization
    in POLYGONAL_LENGTH.
  - NUMBERING          The parameter is the numbering of points.  It  is
    only    appropriate    if    the    numbering    reflects    a  reasonable
    parametrization of the curve.
```

5.5.3 CURVE /PERIODIC

```
    [GREG1\]CURVE /PERIODIC

Specifies that the curve is periodic (and  currently  closed  also).  If
/VARIABLE  X,  it  expects  that  y(1)=y(NXY)  ;  if  /VARIABLE  Y, that
x(1)=x(NXY) ; and in the case of a parametric  curve,  that  x(1)=x(NXY)
and y(1)=y(NXY)
```

5.5.4 CURVE /BLANKING

```
[GREG1\]CURVE /BLANKING Bval Eval
```

Does not uses data points such that abs(Y(i)-Bval) < Eval. These points act as separators. Eval negative means no blanking value. Bval and Eval values default to those specified in SET BLANKING.

5.6 DRAW

```
[GREG1\]DRAW Argument Xc Yc [/CLIP] [/BOX N] [/CHARACTER N] [/USER
[code]]
```

This is a powerful command which allows you to make drawings without data either by specifying coordinates (Xc,Yc) and action desired (Argument), or by interactive use of a cursor.

The cursor is obtained if Xc and Yc are omitted, and the appropriate action is then specified by keys typed at the keyboard: R is for RELOCATE, L for LINE, M for MARKER, A for ARROW, T for TEXT. In addition, C works like T, but prompts for the centering parameter in addition to the text. D can be used to DELETE the last operation, and has no action on the operations which were plotted previously to the current DRAW command. (Presently D has no visible action. Use ZOOM REFRESH after.) E exits from the cursor routine. ^C will exit AND generate an error, which can be trapped (useful to regain control inside of forever SIC loops calling the interactive cursor repeatedly).

Other keys gives you the cursor coordinates (in the current angular unit when a projection is defined), except that keys '0' '1' '2' and '3' temporarily (i.e., as long one does not exit from cursor mode) changes the current angular unit in which the cursor coordinates are returned.

For X11 displays, you may use the mouse buttons for the commands R (left mouse button), L (middle mouse button), and E (right mouse button).

The complete command is written to the LOG_FILE so that you can replay it easily. The option allows you to supersede the default value of the coordinate system defined by SET COORDINATE or to keep clipping. The code suboption in the /USER option, if present, can be any of 'SECONDS', 'MINUTES', 'DEGREES', 'RADIANS' (in which case the Xc and Yc coordinates are taken to be in the corresponding unit) or 'ABSOLUTE' (in which case Xc and Yc are taken as absolute coordinates, usually exprimed in sexagesimal format). The presence of the code suboption is valid only if a PROJECTION and a SYSTEM (EQUATORIAL or GALACTIC) are both defined.

When using the explicit form, Xc and Yc may be equal to * to use the current pen position.

Any "realistic" combination of literal values, 0-d and 1-dimensional variables is accepted for that command. For example,
DRAW MARKER A B /USER MINUTES
will plot markers at each location given by the two arrays A and B (provided they are of the same size). A and B must be CHARACTER arrays in the case where the "/USER ABSOLUTE" option is used.

5.6.1 DRAW /BOX

```
[GREG1\]DRAW [Action Xc Yc] /BOX N
```

Specifies that coordinates Xc Yc are physical units relative to point number N of the current BOX. The 9 most remarkable points of the box (corners, middle of the sides, and box center) are numbered according to a VT100 numeric keypad notation (see HELP GREG1\LABEL /CENTERING).

5.6.2 DRAW /CHARACTER

```
[GREG1\]DRAW [Action Xc Yc] /CHARACTER N
```

Specifies that coordinates X Y are units of character size, and relative to point number N of the current BOX. The 9 most remarkable points of the box (corners, middle of the sides, and box center) are numbered according to a VT100 numeric keypad notation (see HELP GREG1\LABEL /CENTERING)..

5.6.3 DRAW /USER

```
[GREG1\]DRAW [Action Xc Yc] /USER
```

Specifies that coordinates are user coordinates.

5.6.4 DRAW /CLIP

```
[GREG1\]DRAW [Action Xc Yc] /CLIP
```

Keeps clipping in the box for all drawing actions (except for labels which are never clipped).

5.6.5 DRAW RELOCATE

```
[GREG1\]DRAW RELOCATE Xc Yc
```

Move pen up to given location (Xc,Yc). Cursor form R.

5.6.6 DRAW LINE

```
[GREG1\]DRAW LINE Xc Yc
```

Move pen down to given location (Xc,Yc). The line will be clipped in the box only if the /CLIP option is present. Cursor form L.

5.6.7 DRAW MARKER

```
[GREG1\]DRAW MARKER Xc Yc
```

Draw a graphic marker at given location (Xc,Yc). Current marker style and size is used. Cursor form M.

5.6.8 DRAW ARROW

```
[GREG1\]DRAW ARROW Xc Yc
```

Draw an arrow from current location to given location. The arrow size is governed by EXPAND and MARKER size. Cursor form A.

5.6.9 DRAW TEXT

```
[GREG1\]DRAW TEXT Xc Yc "Text" [N [Orien]]
```

Draw a text a location (Xc,Yc) according to the centering parameter N, or to the current centering parameter if N is not specified. Cursor form T, or C to be prompted for the centering. The orientation angle of the text (in degrees, counterclockwise) can be specified after the centering parameter; otherwise the current orientation defined by SET ORIENTATION is used.

5.6.10 DRAW VALUE

```
[GREG1\]DRAW VALUE Xc Yc
```

Inquires the map value at the location (Xc,Yc). A regular grid array must be present to use this, (Xc,Yc) must be within the Regular Grid array limits; otherwise the command is ignored. Cursor form V.

5.7 ERRORBAR

```
[GREG1\]ERRORBAR Argument [Array_X Array_Y Array_Z [Orientation]]
```

This command draws error bars on all (X,Y) points (read by COLUMN X and Y, or specified in Array_X Array_Y) of length read by COLUMN Z or specified by Array_Z. Argument may be +X X -X +Y Y or -Y. Symmetric errorbars are drawn if you do not specify the sign, while only the errorbar of the given sign is drawn if you do so. This allows independent errorbars in opposite directions. The size of the line at the end of the errorbar is the Marker size and can therefore be adjusted with SET MARKER.

Errorbars with arbitrary orientation can be obtained by specifying code +O, O or -O. The orientation (in degrees) is then either read from a fourth array if one is specified on the command line, or otherwise taken from the current marker orientation (as specified in SET ORIENTATION).

5.8 HISTOGRAM

[GREG1\]HISTOGRAM [Array_X Array_Y] [/BASE [Ybase]] [/BLANKING Bval Eval] [/FILL [pen_num]]

Connects the coordinates read by COLUMN X and Y, or stored in the variables Array_X Array_Y as a histogram. The /BASE option specifies a "true" histogram, connected to a base level (Default Ybase=0). The /BLANKING option will supersede the current blanking values (as defined by SET BLANKING command) so that data points with abs(Y(i)-Bval) < Eval are not plotted. Eval negative means no blanking value. Filled Histogram is created by the options /FILL + /BASE.

5.8.1 HISTOGRAM /FILL

[GREG1\]HISTOGRAM [Array_X Array_Y] [/BASE Ybase] /FILL [pen_num]

The /FILL option indicates to fill the histogram with the colour of the current pen. Blanking is handled as for other histograms. Usually needs to be done in conjunction with the BASE option to properly fill the histogram. Pen_num is the number (0-23) of the pen used to fill. The colors of the last 16 pens (8-23) can be changed by the user (See HELP LUT /PEN).

5.9 LABEL

[GREG1\]LABEL "Character String" [Orientation] [/X] [/Y] [/APPEND] [/CENTERING N]

Writes the string Character String (included between double quotes) at current pen position or location specified by option, and with current or requested centering. The label is written according to the current or specified Orientation and EXPAND factors. Within a string, the character "\" is an escape character and causes the following action :

```
\\X - set mode X
 \X - set mode X for next char
 \N - Default character set
 \1 - Simplex character set
 \2 - Duplex character set
 \R - Roman font
 \G - Greek font
 \S - Script font
 \I - toggle italics
 \U - superscript
 \D - subscript
 \B - backspace over previous char
```

5.9.1 LABEL /X

```
[GREG1\]LABEL /X
```

Put the label centered below the X axis made by BOX.

5.9.2 LABEL /Y

```
[GREG1\]LABEL /Y
```

Put the label centered left of the Y axis made by BOX.

5.9.3 LABEL /APPEND

```
[GREG1\]LABEL /APPEND
```

Append the label at current pen location. This corresponds to /CENTERING 6.

5.9.4 LABEL /CENTERING

```
[GREG1\]LABEL /CENTERING N
```

```
The label is oriented with respect to the current location according  to
the argument N which can be 1 - 9 for :
                        right    center     left       justified
        label above       7         8         9
              centered    4         5         6
              below       1         2         3
These conventions follow the standard numeric keypads notation.
```

5.10 Language

```
                    GREG1\ Language Summary
AXIS            : Draws an axis according to its name
```

```
BOX            : Makes a box labelled according to LIMITS and TICKSPACE.
COLUMN         : Reads the data file
CONNECT        : Connects (x,y) pairs with line segments
CURVE          : Connects (x,y) pairs with a spline interpolation
DRAW           : Calls the interactive cursor or execute detailed command
ERRORBAR       : Draws error bars on (X,Y) pairs
HISTOGRAM      : Connects (x,y) pairs as an histogram
LABEL          : Writes a string...  according to options given
LIMITS         : Sets the limits of the plot (no args for auto)
PENCIL         : Select and defines the pen attributes
POINTS         : Draws markers at the (X,Y) pairs
RULE           : Makes a grid by joining axis tickmarks.
SET            : Modifies some basic parameter
SHOW           : Shows basic parameters
SORT           : Sorts the current X-Y-Z buffers according to one.
TICKSPACE      : Sets tick intervals for BOX or AXIS
VALUES         : Write the Z values at the (X,Y) positions
```

5.10.1 Language NEWS

```
Oct 1995: Separate text and marker orientation. Text  orientation  can
  now  also be given in command LABEL and DRAW TEXT.  See DRAW, LABEL,
  SET MARKER, SET ORIENTATION.
Jul  1995:  RGDATA  enhanced:  /SUBSET  option  even  with  /VARIABLE,
  simple-minded creating of a grid from X,Y,Z variables, etc...
Jun 1995: "true" astronomical labelling. See BOX and SET  SEXAGESIMAL.
  WEDGE command enhanced.
Mar 1995: Pie charts possibility with ELLIPSE /ARC.
Oct 1994: Support array variables  as  X  \&  Y  coordinates  in  DRAW
  command.   CHARACTER array variables are also supported in DRAW TEXT
  command.  Supports CHARACTER arrays for  coordinates  in  the  /USER
  ABSO option.  See DRAW.
Sep 1994: Option /FILL to command  BOX  to  have  really  nice  colour
  plots.  See BOX.
```

5.11 LIMITS

```
    [GREG1\]LIMITS [X1  X2  Y1  Y2  [Unit]]  [/XLOG]  [/YLOG]  [/RGDATA]
[/BLANKING Bval Eval] [/REVERSE [X] [Y]] [/VARIABLES Array_X Array_Y]

Sets the coordinates of  the plot  region.   The  limits  are  the  user
coordinates   of   the   BOX   corners,  and  in  conjunction  with  the
BOX_LOCATION, specifies the conversion formula  between  USER  and  PLOT
coordinates.  If LIMITS  has  no arguments,  it will  computes automatic
limits.  The data from  the most recent COLUMN X and Y are  used  to set
the limits, except if the  /RGDATA  option is present, in which case the
last read regular grid map is used.
```

```
         Some special characters are allowed instead of numerical values
for the limits :
      *   Compute automatic limit for this argument
      <   Compute automatic limit for this argument, and take
          the minimum of this value and the precedent limit.
      >   Same as above, but take the maximum.
      =   Keep the precedent value for this limit.

An additional argument can be used when a projection and  a  system  are
defined. It may take values
  - RADIAN : The values are assumed to be normal  projected  coordinates
    from the projection center.
  - SECOND, MINUTE or  DEGREE  :  The  values  are  converted  from  the
    specified angular unit to radians. They still represent offsets from
    the projection center.
  - ABSOLUTE : The values are assumed to be absolute coordinates on  the
    sphere. APPROXIMATE corresponding projected coordinates are computed
    from  these  values  (the  projected  coordinates  are   necessarily
    approximate  since  in  general  e.g.  the upper left and lower left
    corner of the box correspond to different absolute  Right  ascension
    for Equatorial system).
The default value is the current angle unit as defined by SET ANGLE.
```

5.11.1 LIMITS /BLANKING

```
    [GREG1\]LIMITS [Args . . .] /BLANKING Bval Eval

Does not uses data points such that abs(Y(i)-Bval) < Eval. These  points
act as separators.  Eval negative means no blanking value. Bval and Eval
values default to those specified in SET BLANKING. Cannot  be  use  with
/RGDATA.
```

5.11.2 LIMITS /REVERSE

```
    [GREG1\]LIMITS [Args . . .] /REVERSE [X] [Y]

Exchange the left and right limits for X axis,  and/or  top  and  bottom
limits for Y axis.
```

5.11.3 LIMITS /RGDATA

```
    [GREG1\]LIMITS [Args . . .] /RGDATA

Specifies that the current Regular Grid array should be used to  compute
the  (non-specified)  limits,  instead  of  the  current  column arrays.
Logarithmic  conversion  formulae  are  not  allowed   in   this   case.
The /BLANKING and /VARIABLES options are incompatible with /RGDATA.
```

5.11.4 LIMITS /XLOG

```
[GREG1\]LIMITS [Args . . .] /XLOG
```

Specifies that the X conversion formula should be logarithmic. Cannot be used with the /RGDATA option.

5.11.5 LIMITS /YLOG

```
[GREG1\]LIMITS [Args . . .] /YLOG
```

Specifies that the Y conversion formula should be logarithmic. Cannot be used with the /RGDATA option.

5.11.6 LIMITS /VARIABLES

```
[GREG1\]LIMITS [Args . . .] /VARIABLES Array_X Array_Y
```

Use SIC variables Array_X Array_Y to compute the limits, instead of the X and Y buffers. Cannot be used with the /RGDATA option.

5.12 PENCIL

```
[GREG1\]PENCIL [N] [/COLOUR C] [/DASHED D] [/WEIGHT W] [/DEFAULT]
```

Selects virtual pen N and defines its attributes in terms of colour (if supported by the device), dashed pattern and weight (thickness). Up to 16 pens can be defined (0-15). Most plotting commands uses the current pen. The major exception is RGMAP. Some commands also ignore the dashed patterns, in particular all labels and the BOX command.

- The dashed pattern D must be in range 1 to 7, 1 means solid lines.
- The weight factor W must be in range 1 to 5.
- The colour index C is an integer in the range 0-23. The first 8 colors are
  FIXED, the last 16 are user-definable. See the topic COLOR for more information on pen colors.

PENCIL N /DEFAULT reset default attributes to Pen number N. PENCIL /DEFAULT reset default attributes for all Pens, as defined at program initialization.

5.12.1 PENCIL COLORS

There are 24 colors available for PENS. Colors 0 and 7 are SPECIAL, colors 1 to 6 are FIXED (named colors or their equivalent shade of grey on grayscale terminals). The last 16 are user-defined.

Color 0 (the default for the default PEN 0) is ALWAYS the foreground color (i.e., always visible whatever the background color of the graphic window is), and becomes BLACK on the paper hardcopy. Color 7 is ALWAYS the background color, useful for text written on an grey-scale image for example.

Colors 1 to 6 are defined as follows for a standard display with sufficient colors available in its colormap:
    "red","green","blue","cyan","yellow" and "magenta"
For grey-scale displays, it becomes:
    "grey42","grey70","grey14","grey28","grey56" and "grey84"
This may vary depending on the number of available colors on your display.

The last 16 colors are user-definable (see HELP LUT) and correspond to the 16 'FILL PENS' used as fill colors in various commands, like RGMAP /GREY or POLYGON /FILL.

5.13 POINTS

    [GREG1\]POINTS [Array_X Array_Y] [/BLANKING Bval Eval] [/SIZE Z_Value [Array_Z [Exponent]]] [/MARKER Array_Nsides Array_Style]

Makes points of the current style (MARKER), size (MARKER and EXPAND), and orientation (ORIENTATION) at the coordinates read by COLUMN, or in the variables Array_X Array_Y.

With the option /SIZE, followed by a real value Z_Value, POINTS will draw markers of sizes governed by the values of the Z array (or of the array Array_Z if specified). Points with Z value equal to Z_Value will have a marker of size the current marker size. All other markers will have an AREA proportional to the Z value at each point (i.e. marker radius at X(i) Y(i) proportional to the square root of Z(i); this power law exponent can be changed by specifying an additional argument to option /SIZE). If Z_Value is equal to zero, the program uses the maximum absolute value of the Z array instead.

The /BLANKING option can be used to override the current blanking behaviour, as defined by SET BLANKING command.

The /MARKER option can be used to specify a different marker sidedness and style for each data point.

5.14 RULE

    [GREG1\]RULE [X] [Y] [/MAJOR] [/MINOR]

Without an argument, or with both X and Y arguments, RULE makes a grid within the BOX using the current pen. Major or minor ticks or both are selected according to the options. With the argument X or Y, the grid is restricted to X or Y ticks respectively. The /MAJOR and /MINOR options can be combined in the same command.

5.14.1 RULE /MAJOR

```
[GREG1\]RULE [X] [Y] /MAJOR
```

Makes a grid at major ticks (This is the default if no option is given).

5.14.2 RULE /MINOR

```
[GREG1\]RULE [X] [Y] /MINOR
```

Makes a grid at minor ticks.

5.15 SET

```
[GREG1\]SET Argument [Value [...]] [/DEFAULT]
```

Specify or reset some default value for a parameter used by GreG. These values may sometimes be overridden for a single command by the appropriate option (e.g. /BLANKING and SET BLANKING). SET Argument /DEFAULT reset the default values for the given key.

SET /DEFAULT reset all default values of SET including the FONT and the PLOT_PAGE ; thus the plot cleared. Use PENCIL /DEFAULT to reset the PENCIL definitions, and TICKSPACE 0 0 0 0 to reset tick spacing parameters.

5.15.1 SET ANGLE_UNIT

```
[GREG1\]SET ANGLE_UNIT D
```

Sets the angle unit for the projection limits. This will become effective only when a projection is effective, and only affects the way limits are given and labels written (not internally stored). D may have the values SECOND, MINUTE, DEGREE, RADIAN

5.15.2 SET BLANKING

```
[GREG1\]SET BLANKING Bval Eval
```

Define a blanking value Bval with tolerance Eval. Default is no blanking

```
(i.e.  Eval  negative).  The  blanking  is  used by most curve plotting
commands such as CURVE, CONNECT, POINTS  to  ignore  data  points  which
verify  ABS(Y-Bval)  less  or  equal to Eval. The blanking value is also
used for contours (RGMAP), to avoid plotting contours  in  pixels  whose
value  verify  the  same  relation  as above. And finally, some commands
require a blanking value to attribute when they are unable to  define  a
real value ; this include RANDOM and MASK.
```

5.15.3 SET BOX_LOCATION

```
    [GREG1\]SET BOX_LOCATION Argument_List

Define the position of the box.  The  argument  list  specifies  various
options for the box position :
  - PORTRAIT     Define a default Portrait oriented box
  - LANDSCAPE    Define a default Landscape oriented box
  - SQUARE       Define a square box
  - MATCH [Gx1 Gy1 [Ratio]]   Define a box which matches  the  ratio  of
    the  physical units. By default, the bottom left corner is placed at
    the default position for the current plot page, but can be moved  by
    specifying arguments Gx1 and Gy1. The box size is adjusted such that
    the box occupies as much space as possible within  the  default  box
    position  for  the  current  plot  page type. Ratio is an additional
    arguments which indicates a scaling factor for the Y axis.
WARNING : The box is computed from the  current  limits,  and  will  not
    change if you computes new limits
  - Gx1 Gx2 Gy1 Gy2  Defines  the  positions  of  all  box  corners  in
    physical units.
The default is LANDSCAPE, PORTRAIT or the default VIEWPORT according  to
the current PLOT_PAGE type.
```

5.15.4 SET CENTERING

```
    [GREG1\]SET CENTERING N

Sets the centering option. Default is 0, which means automatic centering
according to the position relative to the box.
```

5.15.5 SET CHARACTER

```
    [GREG1\]SET CHARACTER S

Sets the character size to S physical units. Default is 0.6 .

2 COMMENT
    [GREG1\]SET COMMENT "Separator"
```

Specify which character is used to indicate the beginning of a comment line in formatted input files for command COLUMN. "Separator" is a single character; typical values are "!", ";", "#", although any single character can be specified. The default is "!".

5.15.6 SET COORDINATES

```
[GREG1\]SET COORDINATES [System [N]]
```

Sets the coordinate system for DRAW. The system may be BOX, CHARACTER or USER. If BOX or CHARACTER, N can be used to specify the point of the box to which the units refer. Nine points are available, following a standard VT100 keypad disposition (see HELP GREG1\LABEL /CENTERING). N may be 0. In this case, the nearest point will be used when making annotations with the cursor, and point 1 will be used in explicit form. The default is BOX 0.

5.15.7 SET DECIMAL

```
[GREG1\]SET DECIMAL [X] [Y]
```

Specify that labelling of X or Y axis is with a decimal field (default). Used to revert from sexagesimal labelling (See HELP SET SEXAGESIMAL).

5.15.8 SET EXPAND

```
[GREG1\]SET EXPAND E
```

Expands all characters or markers by a factor E. Default is 1.

5.15.9 SET FONT

```
[GREG1\]SET FONT F
```

Defines the character font used. F may be SIMPLEX (default) or DUPLEX, which is nicer but substantially slower, and should be reserved to publication quality plots. Default is SIMPLEX.

5.15.10 SET MARKER

```
[GREG1\]SET MARKER Nsides Mstyle Size Orien
```

Sets the marker type to an polygon of Nsides, of style Mstyle (0-3) and size Size physical units.

```
    Nsides=0,1 Dots ; Nsides=2 Bar ; Nsides=3 Triangle ...
    Mstyle=0 Convex polygon  ;  Mstyle=1  Vertex  connected  ;  Mstyle=2
    Starred polygon, and Mstyle=3 Filled polygon (Only for Nsides >= 3)

The default is Nsides=0 Mstyle=0  Size=0.  and  Orientation  0  (Points,
optimised  algorithm).  The Orientation is used to plot the marker (e.g.
Horizontal bar for MARKER 2 0 * 0.0, and vertical bar  for  same  marker
MARJER 2 0 * 90.0).

A * instead of a numerical value indicate to keep the last value.
```

5.15.11 SET ORIENTATION

```
    [GREG1\]SET ORIENTATION T [M]

Set orientation of text to T degrees and that of markers to M degrees (M
=   T  if  not  specified).  The  text  orientation  also  controls  the
orientation of axes. Default is 0.0 (horizontal).

Text orientation can also be controlled by command DRAW TEXT, and Marker
orientation by command SET MARKER.
```

5.15.12 SET PLOT_PAGE

```
    [GREG1\]SET PLOT_PAGE X Y

Sets the physical  size  of  plot  page.  One  can  use  "PORTRAIT"  and
"LANDSCAPE"  for  A4  format  with adequate orientation, and square will
give a 30 by 30 cm plot page. Default is LANDSCAPE (A4 30 by 21 cm).
```

5.15.13 SET SEXAGESIMAL

```
    [GREG1\]SET SEXAGESIMAL [X [Nx]] [Y [[Ny]]

Set X axis labelling in sexagesimal notation with Nx decimals (resp. Y).
The  default,  (Nx=1, Ny=0 and no sexagesimal notation) is nearly always
perfect for everybody.

So, this command is used for two restricted purposes:
   - to enable sexagesimal notation of axis when a projection is in use,
    the  system  is  GALACTIC  and the /ABSOLUTE option is required (for
    EQUATORIAL systems, the sexagesimal notation is always used in  this
    case,   but   the   default   for   GALACTIC  system  is  a  decimal
    representation of degrees).
   - to specify the number of decimals  when  using  ABSOLUTE  labelling
    (see BOX /ABSOLUTE or AXIS /ABSOLUTE).
```

Note that, when a realistic tick spacing is required by the user or (the default) GreG computes this spacing automatically, labelled tickmarks will fall on rounded values, so no decimals will appear (unless the field of view is smaller than a few arc seconds).

5.15.14 SET SYSTEM

```
[GREG1\]SET SYSTEM Name
```

Specify the coordinate system used in case of projection. The name can be UNKNOWN, GALACTIC or EQUATORIAL. The SYSTEM affects the projection definitions (in particular the GRID command) and coordinates sent back by the cursor (different formats are used for each system).

5.15.15 SET TICKSIZE

```
[GREG1\]SET TICKSIZE S
```

Sets the tick size to S physical units (Default 0.3).

5.15.16 SET VIEWPORT

```
[GREG1\]SET VIEWPORT Px1 Px2 Py1 Py2
```

Define the position of the box in terms of fractions of the plot page size in X and Y. Default is 0.150 0.925 0.125 0.925 .

5.16 SHOW

```
[GREG1\]SHOW Item
```

- SHOW <SET_key> lists the current value of the associated parameter(s).
- SHOW ALL lists the values of all the SET parameters.
- SHOW PEN lists the current pen definitions (use the PENCIL command itself for informations on the active pen alone).
- SHOW PROJECTION lists the current projection used.
- SHOW [WHERE or LIMITS] [DEGREE,MINUTE,SECOND or RADIAN] shows the current pen location and plot region limits in User and Plot coordinates. The second argument modifies the angular units in which the user's values are displayed, when a projection is enabled of course.
- SHOW Without arguments is a synonym of SHOW WHERE

5.17 SORT

```
[GREG1\]SORT Key
```

Sorts the current X-Y-Z buffers according to the one given as Key name, which will be in ascending order after the sorting. It may be useful if you want to plot a curve, but have data disordered.

5.18 TICKSPACE

```
[GREG1\]TICKSPACE SmallX BigX SmallY BigY
```

Sets tick intervals (in User coordinates) for BOX or default intervals for AXIS. SmallX refers to the interval between small tick marks on the X axis, BigX refers to the interval between large ticks, etc... If Big is 0, the axis routine will supply its own Big intervals according to the label limits and use your Small value if non zero. To restore fully automatic ticking, set Small and Big equal to 0. To suppress small ticks, specify equal values for Small and Big.

The Small and Big values are meaningless for LOGarithmic axes, which uses their own conventions. These conventions are not quite perfect, and if less than one decade is plotted, odds are good that you will not get any labelling !...

When using the /ABSOLUTE or /UNIT option for axis-drawing commands, one has to keep in mind that Small and Big (if not 0) should be expressed:

- in seconds of arc for declination or galactic (sexagesimal notation) axis, /ABSOLUTE option (one can usually multiply by 3600 the desired tick spacing)
- in seconds of time for right ascension axis, /ABSOLUTE option (multiply by 12)
- in seconds, minutes or degrees when option /UNIT [SECOND, MINUTE or DEGREE] is used.

Thus in the case of a user-defined tick spacing, TICKSPACE must be redefined before using another option.

5.19 VALUES

```
[GREG1\]VALUES [Array_X Array_Y Array_Z]
```

Writes (in the shortest possible format) the values of the Z array at the (X,Y) positions in the box. The current character size, expansion factor and fonts are used. Only points inside the current box are written, but the labels may still span over the box itself. Arrays may be specified explicitly.

6 GREG2 Language Internal Help

6.1 CONVERT

[GREG2\]CONVERT A0 D0 [Angle] [/TYPE Ptype] [/UNIT Unit] [/SYSTEM System] [/VAR Varx Vary]

This command assumes that the X,Y buffers (or the 1-D variables Varx and Vary if option /VAR is used) contain data in the specified projection (or in absolute spherical coordinates if Ptype is NONE) in the given angular unit (Degree Minute Seconds or the default Radian), in the specified System (GALACTIC or EQUATORIAL, default current system) and converts them to the current system, as defined by SET SYSTEM, and projection, as defined by command PROJECTION. Note that the projected coordinates are always natural units (i.e. the equivalent of Radians after projection). There is no provision for "hidden" parts which are treated exactly as visible ones.

A0 D0 are Hours and Degrees if the input system is EQUATORIAL, Degrees otherwise.

CAUTION: when using variables, the Varx and Vary arrays MUST BE REAL*8, and WRITEABLE (see HELP DEFINE).

6.2 ELLIPSE

[GREG2\]ELLIPSE Major [Minor [Position_Angle]] [/BOX X Y] [/USER X Y [code]] [/ARC Theta_min Theta_max] [/FILL [Colour]]

This command draws at current pen position (or at position (X,Y) if option /BOX or /USER is present) an ellipse of given semi-major axis, semi-minor axis, (default Minor = Major) and PositionAngle (default 0 degrees).

Major, Minor, X and Y are in User Coordinates, but Position_Angle is ALWAYS in DEGREES, TRIGONOMETRIC ORIENTATION. Major, Minor, X and Y, can be 1-D arrays of any size, provided the sizes are the same. Position_Angle can be indifferently an array, a number, or omitted.

The code suboption in the /USER option, if present, can be any of 'SECONDS', 'MINUTES', 'DEGREES', 'RADIANS' (in which case the X and Y coordinates are taken to be in the corresponding unit) or 'ABSOLUTE' (in which case X and Y are taken as absolute coordinates, usually exprimed in sexagesimal format). The presence of the code suboption is valid only if a PROJECTION and a SYSTEM (EQUATORIAL or GALACTIC) are both defined.

A portion of the ellipse can be drawn by specifying the boundary angles in option /ARC (in degrees, trigonometric sense from the position angle). Not compatible with the array mode.

Option /FILL [Colour] will fill the ellipse(s), with the current or specified colour. If the /ARC option is also present, the filled area is the corresponding pie part, not the subtended arc.

This command is useful for plotting beam sizes, incertainty ellipses, pie charts, etc...

6.3 EXTREMA

```
[GREG2\]EXTREMA [/BLANKING Bval Eval] [/PLOT]  [/COMPUTE]
```

Computes and shows the minimum and maximum values of the Regular Grid array: shows XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX. A useful command for deciding where to set contour levels, or what size to give to the BOX if you want an exact aspect ratio between two coordinates. The current blanking value is used, but can be overridden using option /BLANKING. Option /PLOT specifies that instead of computing global extrema LOCAL extrema values should be computed, and written at appropriate places on the plot. This is a very good way to avoid ambiguities in contour map. A shortest format is used for writing the values, and the character size controls the label size. If Option /COMPUTE is not present, the extrema are only computed if not already known (from a previous call to EXTREMA, or from an image header). /COMPUTE forces computation from the data array. It should be used if for any reason the known extrema are incorrect (invalid image header, killed bad pixel, change of blanking value...).

6.4 GRID

```
[GREG2\]GRID [Astep [Dstep]] [/ALTERNATE]
```

Plots a grid of meridian and parallels according to the current projection and system. Astep is the grid step for meridian in degrees (Default 15), Dstep for Parallels (Default Rstep). If one step is 0, this part of the grid is not plotted. If the /ALTERNATE option is present, the grid will be a grid in the alternate coordinate system (Equatorial or Galactic). This does not currently work for projection type AITOFF and RADIO.

6.5 LEVELS

```
[GREG2\]LEVELS C1 C2 TO C3 BY C4 ...
```

sets the contour levels to be C1, C2, C2+C4, C2+2*C4, ..C3, and so on, using the same list format as SIC\ command FOR. The contour levels are used to make contour plots of the Regular Grid array using command

RGMAP. Up to 30 contours can be given at a time. Note that it is more efficient to use a single RGMAP command with many levels than many RGMAP commands with a single level each.

LEVELS without arguments will display the contour levels, and LEVELS NONE reset the level list.

6.6 MASK

```
[GREG2\]MASK [IN] [OUT] [/BLANKING Bval]
```

masks all pixels of the Regular Grid array lying inside (if argument is IN) or outside (if argument is OUT or by DEFAULT) the current polygon (see command POLYGON) by attributing them a blanking value. This value is defined by command SET BLANKING, or specified using the option /BLANKING Bval.

6.7 MEAN

```
[GREG2\]MEAN [/CLIP Nsigma]
```

computes crude statistics on the part of the Regular Grid array lying inside the current polygon. Blanking values as defined in SET BLANKING are ignored. The results are displayed (some of them according to the current ANGLEUNIT), and stored in the variables POLY$SUM and POLY$AREA.

If option /CLIP is present, iterative kappa sigma rejection is used to reject outlier pixels.

6.8 PERSPECTIVE

```
[GREG2\]PERSPECTIVE [Altitude Azimuth] [/FACTOR Fv Ov] [/LOWER]
[/LINES] [/STEP nsx nsy] [/AXES Xmin Xmax Ymin Ymax] [/CENTER]
[/VERTICAL Vmin Vmax]
```

Makes a parallel projective view of the Regular Grid array, with hidden lines removed. Altitude and Azimuth are the values of the corresponding viewing angles in Degrees. Default values are 45 and 45 at the beginning, and then the last values given. The upper part is drawn, unless the /LOWER option is present.

Automatic scaling to match approximately the current box is used, but it is only a rough guess and may fail if extrema of the Regular Grid occurs near the edges. Fine tuning of the plot scale can be obtained using the /FACTOR option. It controls scaling of the Regular Grid values by the factor Fv (default 1) and an offset Ov which is in physical units along the Y direction of the plot.

The Regular Grid array is plotted according to the orientation implied by command LIMITS, but the whole array is plotted. The /STEP option may be used to indicate to take only 1 pixel over Nsx ones for the X axis (resp. Nsy and Y). The /AXES option plots (X,Y) axes. The /VERTICAL option controls the max, min of plotted perpsective and plots a Z axis. The /CENTER option centers (?) the perspective. BLANKED PIXELS are forced to the minimum value of the map, or the Vmin value when the/VERTICAL option is in use.

6.9 PLOT

[GREG2\]PLOT [Varname] [/SCALING Type Low [/BLANKING Bval Eval] [/POSITION Posx1 Posx2 Posy1 Posy2] [/LIMITS Userx1 Userx2 Usery1 Usery2] [/VISIBLE]

PLOT will put a pixel image (pixels contained in the variable Varname) inside the current BOX. Varname is a 2-D SIC variable name that defaults to the RG variable if ommitted. Subsets like A[4] where A is a 3-D sic variable are not yet allowed. How to create and handle images in programs calling the GREG2\ language (including GREG) is described in the subtopic IMAGE_HOWTO.

This command is normally used in the following sequence:

```
Greg> SET BOX LOCATION Gx1 Gx2 Gy1 Gy2  ! Define box position
Greg> RGDATA Varname /VARIABLE          ! Put image in the RG variable,
                                        ! read associated header variables
Greg> LIMITS/RGDATA                     ! Computes associated limits
Greg> PLOT                              ! Plot the array in the box
Greg> BOX                               ! Draws the box
```

6.9.1 PLOT /BLANKING

[GREG2\]PLOT /BLANKING Bval Eval

Blanking value (Bval) and tolerance (Eval). Default values are those defined by SET BLANKING command (q.v.)

6.9.2 PLOT /LIMITS

[GREG2\]PLOT /LIMITS Userx1 Userx2 Usery1 Usery2

User coordinates corresponding to the image boundary. The default is those of the plotted image (when the image has any), or the current user limits if not.

6.9.3 PLOT /POSITION

```
[GREG2\]PLOT /POSITION Posx1 Posx2 Posy1 Posy2
```

Box location in physical units (default is the current GreG box location)

6.9.4 PLOT /SCALING

```
[GREG2\]PLOT /SCALING Type [Low_cut High_cut]
```

Defines the type of transfer function to be used. Type can be LIN(EAR), LOG(ARITHMIC) of EQU(ALIZATION), and the Low_cut and High_cut are the lowest and highest represented values respectively. The default ''cuts'' are the image extrema. The current cuts are stored in variable LCUT and HCUT. The current scaling (as an integer) is stored in the variable SCALING. The LOG transfer function computes the logarithm of the pixels values, so does not like negative or null pixels. The EQUalization scaling tries its best to put the same number of pixels in each slot of the color table, thus increasing immensely the visibility of low-contrast images. This scaling mode has problem (and warns user) when a large number of pixels have exactly the same value (i.e., no contrast at all). Beware that most "real" images are obtained with receivers working with a small number (11 to 16) bits, thus restricting the range of available Real numbers in the data.

The SIC variables EQUAL_NLEV and EQUAL_LEV contain the number of different levels computed in EQUALIZATION mode, and the values of these levels. Useful to know which contour levels to draw, and necessary for the WEDGE command.

6.9.5 PLOT /VISIBLE

```
[GREG2\]PLOT /VISIBLE
```

Specify that the image is to be always drawn, even when totally masked by an overlapping image. This allows special features like movies when typing ZOOM REFRESH.

6.9.6 PLOT IMAGE_HOWTO

There exist a number of ways to get pixels into GREG2\.

- Read the X, Y, Z values of a Z=f(X,Y) "image", use RGDATA (if X and Y define a regularly sampled grid) or RANDOM (if X and Y are "random points") to make a Regular Grid Map (see help GREG1\COLUMN, GREG2\RGDATA and GREG2\RANDOM)

```
- Read a RGDATA-format file with the RGDATA command (see topic RGDATA)
- Define a 2-D SIC variable and fill it with  values,  or  connect  a  SIC
  variable  with  an  existing  GILDAS  DATA  FORMAT (GDF) image (See HELP
  SIC\DEFINE), or use the SIC\ACCEPT command to read formatted/unformatted
  streams  of  values  and  put  them  in  a  SIC  2-D  variable (see HELP
  SIC\ACCEPT).  Then give this 2-D variable to the internal  image  buffer
  by the RGDATA Varname/VARIABLE command.
- The command GRAPHIC\IMAGE of GRAPHIC,  which  is  a  superset  of  GREG,
  connects  directly an existing 2,3 or 4-D GILDA DATA FORMAT image in the
  GREG image buffer. Use the GRAPHIC program instead of GREG and  look  at
  HELP IMAGE.
- GRAPHIC enable the parallel run of image-processing  programs.  Use  the
  VECTOR\RUN  FITS_GILDAS command to convert a FITS image to a GDF (Gildas
  Data Format) image that will be  directly  readable  afterwards  by  the
  GRAPHIC program.
- The GFITS program reads FITS images and converts them to GDF format, and
  the reverse.
- "on-the-fly" automatic conversion from FITS to GDF format for SIC\DEFINE
  IMAGE and GRAPHIC\IMAGE commands.
```

6.10 POLYGON

```
    [GREG2\]POLYGON [(File)Name] [/PLOT] [/VARIABLE] [/FILL [Colour]]

If no argument and no option, it calls the interactive cursor to  define
the  polygon  summits.   Type any key to go to next summit, D to correct
the last one and type E to end the polygon definition. The last  polygon
side  will  then appear. The polygon definition may be aborted by typing
Q. For X11 displays, you may use the mouse  buttons  for  the  commands.
The  left  mouse  button draws a vertex, the middle mouse button deletes
the  last  vertex,  and  the  right  mouse  button  ends  the   polygon
definition.

The drawing of the polygon is not stored in the plot, and will disappear
if  you refresh the plot. The polygon values are stored internally; they
can be printed with the command GREG1\WRITE POLYGON, and they correspond
to  the  variables POLY$NXY (number of vertexes), POLY$X and POLY$Y (the
corresponding coordinates). The currently defined POLYGON is used by the
commands MEAN and MASK.

If a Name is given as argument, the cursor is not called but the  summit
coordinates are read from the corresponding file (X is in column 1, Y in
column 2), or corresponding SIC Variable if the option /VARIABLE is also
present.

If you specify the option /PLOT, the current polygon will be plotted  as
real  vectors  in the plot, and no longer as temporary vectors as above.
```

```
Option /FILL [Colour] will fill the current polygon, with the current or
specified colour.
```

6.11 PROJECTION

```
    [GREG2\]PROJECTION [A0 D0 [Angle]] [/TYPE Ptype]

Define a projection of the (celestial) sphere  from  point  of  (A0,D0),
(which  are  Longitude and Latitude respectively) of the specified type.
Angle is the angle between the Y axis and the North pole.  The  previous
values are kept if no argument is specified.  All angles are in degrees,
except if the SYSTEM is  EQUATORIAL  in  which  case  A0  is  the  right
ascension  and must be specified in hours. Formats like  -dd:mm:ss.s  or
  hh:mm.mmm  in sexagesimal notation up to the point field are  allowed.
After the point, decimal values are assumed.

When a projection is active, the User  coordinates  are  assumed  to  be
projected  coordinates  of  the  sphere,  and hence in the case of small
field of view where distortion are  negligible,  correspond  to  angular
offsets  MEASURED  IN  RADIANS.  The  field of view of the projection is
defined by command LIMITS. For convenience (?), LIMITS may be  specified
in  a  so-called  ANGLE_UNIT  which  can  be  set to Seconds, Minutes or
Degrees. Note that only the numbers you give are assumed to be  in  this
"ANGLE_UNIT" :  they  are immediately converted to the natural projected
unit (equivalent to Radian in case of small field) by  division  by  the
appropriate  scaling  factor.  That  is  all what this improperly named
ANGLE_UNIT means.

The TYPE can be
  - NONE             Disables the  projection  system.  User  coordinates
    then  loose  their interpretation in terms of projected coordinates.
    The ANGLE_UNIT is then totally ignored.
  - GNOMONIC         Radial projection on the tangent plane. Being R  and
    P the (angular) polar coordinates from the projection point (tangent
    point), the projected coordinates are given by X = Tan(R).Sin(P) and
    Y = Tan(R).Cos(P) .
  - ORTHOGRAPHIC   View  from  infinity.  X  =  Sin(R).Sin(P)  and  Y  =
    Sin(R).Cos(P)
  - AZIMUTHAL      Spherical offsets from the projection  center.   X  =
    R.Sin(P) and Y = R.Cos(P).
  - STEREOGRAPHIC  Uses Tan(R/2) instead of Tan(R),  and  is  thus  less
    distorted  than  the  Gnomonic projection. This is an inversion from
    the opposite pole.
  - LAMBERT        Equal  area   projection.   Projected   distance   is
    2*Sin(R)/Sqrt(2*(1+Cos(R)).
  - AITOFF         Equal area projection. Angle and D0 are ignored
  - RADIO          The standard  radio  astronomy  single  dish  mapping
```

"projection", in which X = (A-A0).COS(D0) and Y = D-D0. The Angle is obviously ignored.

6.12 RANDOM_MAP

[GREG2\]RANDOM_MAP [Arg1 [Arg2]] [/NEIGHBOURS Nb] [/TRIANGLES] [/EXTRAPOLATE] [/BLANKING Bval]

This command is a general purpose interpolation task which uses data from the X, Y and Z arrays to fill in a Regular Grid array representing the same surface Z=f(X,Y) by an interpolation process. X, Y and Z are arrays defined by the COLUMN command (see HELP COLUMN). It triangulates the data for interpolation. If a triangulation already exists, and only the Z array has been changed, RANDOM_MAP does not make a new triangulation.

Usually, the user-provided Z=f(X,Y) surface is REGULARLY SAMPLED, even if NOT COMPLETELY SAMPLED. In this case, one SHOULD USE directly the RGDATA command to create a Regular-grid map, since RANDOM is known to perform awkwardly with regularly sampled data.

The interpolated grid can be defined by the arguments. If Arg1 is "RGMAP", the interpolated grid will be exactly coincident with the current Regular Grid array, else Arg1 and Arg2 are used to indicate the number of pixels in X and Y directions and Arg2 = Arg1 if not specified.

Note that several hypothesis are made about the geometry of triangles in the plot page space (hence the geometry may be changed by changing the LIMIT or the BOX_LOCATION). You should thus use limits which gives a reasonable representation of what are the neighbours of a given data point. Moreover, this is an interpolation process. Hence if the basic sampling theory is not respected by your choice of input data points and output grid, spurious oscillations may occur in the resulting grid.

6.12.1 RANDOM_MAP /NEIGHBOURS

[GREG2\]RANDOM_MAP /NEIGHBOURS Nb

The argument Nb changes the number of points influencing the interpolated values. Default is 4 and values up to 8 may be used in some cases. Caution, it slows down the program, and may enhance oscillations in some cases.

6.12.2 RANDOM_MAP /TRIANGLES

[GREG2\]RANDOM_MAP /TRIANGLES

Plot the triangles selected by the program for interpolation. A single graphic segment will contain all triangles.

6.12.3 RANDOM_MAP /EXTRAPOLATE

```
[GREG2\]RANDOM_MAP /EXTRAPOLATE
```

Control wether extrapolation outside the convex hull of the triangles is to be done or not. The default is not, i.e. the command fills with the blanking value the exterior of the convex hull. Beware that extrapolation is always hazardous.

6.12.4 RANDOM_MAP /BLANKING

```
[GREG2\]RANDOM_MAP /BLANKING Bval
```

Specify the value Bval to be attributed for pixels lying outside the convex hull of triangles. The default value is that defined in the SET BLANKING command. This switch is mandatory if no default blanking value is preset and no extrapolation required.

6.13 RESAMPLE

```
[GREG2\]RESAMPLE Nx Ny [/X Xref Xval Xinc] [/Y Yref Yval Yinc]
[/BLANKING Bval]
```

This command resamples (by bilinear interpolation) the current regular grid array on a different grid. The number of pixels of the new grid must be specified. Unless the user specifies an explicit conversion formula in the /X and /Y options, the program adjusts it to match the current Regular Grid array. A blanking value MUST BE specified (either by default in SET BLANKING or using the option /BLANKING) because no extrapolation is performed. For the same reason, any output pixel's value that would be interpolated from at least one input blanked pixel, is itself blanked.

6.14 RGDATA

```
[GREG2\]RGDATA [Name|Array_X Array_Y Array_Z] [/SUBSET NXmin NYmin
NXmax NYmax] [/VARIABLE] [/FORMAT Fmt [code]] [/INCREMENT IncX IncY]
[/BLANKING Bval]
```

This command optionally creates, then loads a two-dimensional map into the Regular Grid array (to be contoured by RGMAP, plotted by PLOT...). This is THE basic command needed for subsequent 2-dimensional treatment. "Regular Grid" means that data is internally stored as a two-dimensional array, thus lies on a Regularly sampled Grid. The user-provided "pixel values" may come from various "descriptions" of such regularly sampled (but possibly uncompletely sampled) data:

```
    1) Three 1-dimensional arrays, say, X Y and Z, that contain  Z=f(X,Y)
     at regular intervals in X and Y.
    2) A preexisting internal 2-D array (possibly a section of a 3D or 4D
     array).
    3) An external, specially formatted, file.

For data NOT REGULARLY ("randomly") sampled,  use  the  GREG2\RANDOM_MAP
command  instead. The Regular Grid is itself a 2D SIC variable under the
name "RG". Once loaded, the RG can be saved on a file with  the  command
GREG2\WRITE.

The following command syntax is used to deduce  which  "description"  is
provided:
  - Without arguments following the command, or with 3 arguments, RGDATA
    will  try  to assemble Z=f(X,Y) either form the X,Y and Z arrays (as
    defined by command COLUMN) or from "Array_X Array_Y  Array_Z",  that
    should be 1D SIC variables previously created by command SIC\DEFINE.
    The option /INCREMENT is used to force the X and Y sampling interval
    to  (IncX,IncY)  in  case of roundoff errors. If a blanking value is
    currently defined, non-sampled pixels will be attributed this value,
    which  is  overriden by the Bval value of the /BLANKING option. Note
    that, if a blanking value is currently defined, pixels for  which  X
    or Y are in the blanking range are themselves blanked.
  - With ONE argument (Name) following the command, Name is taken as the
    name  of an existing SIC 2D variable if option /VARIABLE is present.
    Otherwise, Name is a  specially  formatted  file  (see  HELP  RGDATA
    FILE_FORMAT) from which the Regular Grid array must be read in.
A rectangular SUBSET of the array, defined by the pixel  coordinates  of
the  Bottom  Left  Corner  and Top Right Corner can be selected.  RGDATA
allocates virtual memory according to the map size, so that there is  no
limit  on the array dimensions. Beware however that handling maps larger
than 512 by 512 pixels may be slow...
```

6.14.1 RGDATA /FORMAT

```
    The /FORMAT option governs how the data part (pixel values)  of  the
RGDATA file must be read by GreG. fmt is a valid fortran format (default
is 10z8, the minimal way to encode real*4 data). code may be BY ,  R4   ,
R8   ,  I4   ,  I2,  depending  if the data written in the file are BYtes,
Real4, Real8, Integer4 or Integer2. (the values in  the  GreG  map  will
always be Reals*4 after the read is done).
```

6.14.2 RGDATA FILE_FORMAT

```
    The format for the input file consits of a four line header as such:
1st line : NX XREF XVAL XINC
2nd line : (any comment for first axis, could be blank)
3rd line : NY YREF YVAL YINC
4th line : (any comment for 2nd axis, could be blank)
followed by the data (NX times NY pixel values) in a format which can be
specified with the /FORMAT option when the default 10z8 is not used.
NX is the number of pixels in the 1st dimension of the map  (on  the  "X
axis"),  XREF  the  index  of  the pixel for which the actual X abscissa
value is XVAL, the increment between two pixels being XINC, and the same
for  the  y  axis.  Thus,  the  abcissa X of the pixel 'I' is just X=(I-
XREF)*XINC+XVAL
This format is very simple-minded (so that any inexperienced  programmer
can  use  it  in  less  than 5 minutes), but highly non efficient in I/O
processing.  A more elaborate data format ("IMAGES"),  specially  suited
for astronomical applications and large maps is available in the GRAPHIC
program (a superset of GreG), or directly using SIC  variables  (command
SIC\DEFINE)  and  the /VARIABLE option of this command.  One should read
once its data in RGDATA format in GreG, then convert it for  future  use
in IMAGE format with the command GREG2\WRITE IMAGE filename.

See also the IMAGE_HOWTO subtopic of the GREG2\PLOT command.
```

6.15 RGMAP

```
    [GREG2\]RGMAP [Quiet|Fast|chunksize] [/PERCENT  P1]  [/ABSOLUTE  A1]
[/BLANKING Bval Eval] [/GREY COLOUR N] [/KEEP File_Name [Blank]]

Draws (or Fills) isocontours of the Regular Grid array.  The  RG  is  an
internal   2D   "map"   created/loaded   by  various  commands  (RGDATA,
RANDOM_MAP, RESAMPLE, IMAGE), or loaded by a calling program. The levels
are  those specified by command LEVELS, but may be affected by a scaling
factor by option /PERCENT or /ABSOLUTE.

This command is normally used in the following sequence:
Greg> SET BOX LOCATION Gx1 Gx2 Gy1 Gy2  ! Define box position
Greg> RGDATA (options...)               ! Put image in the RG variable,
                                        ! read associated header variables
Greg> LIMITS/RGDATA                     ! Computes associated limits
Greg> PLOT                              ! Plot the array in the box
Greg> BOX                               ! Draws the box
Greg> LEVELS 0.1 to 0.9 by 0.1          ! define contour levels
Greg> RGMAP/PERCENT 1                    ! Draw contour levels  (in this
                                        ! case, as percentage of peak)

The "Quiet" mode does contouring silently. For  "Fast"  and  "chunksize"
```

modifiers, see subtopic MEMORY_OPTIONS

6.15.1 RGMAP MEMORY_OPTIONS

The program normally allocates less that 1/5 of the total number of pixels in the map to store the contour segments, and warns the user when that limit is reached. Besides, contouring may be slow on large maps. The modifier "Fast" cuts the map in smaller "chunks" and performs contouring on those submaps, without the 1/5 above limitation.

The Contour-filling mode (/GREY) can consume (temporary) a large amount of memory, so is always performed on subsets of the image ("chunks") when the image size is greater than a system-dependent limit (256*256 on UNIX systems). When the contours are too involved, an internal limit can nevertheless be reached. One can then retry the contour-filling by specifying a smaller "Chunksize" (in pixels). For example, to force contour-filling on subsets of 64 "lines" of a 512x512 pixel image, type RGMAP 64*512 /GREY

6.15.2 RGMAP /PERCENT

```
[GREG2\]RGMAP /PERCENT P1
```

Computes M=MAX(ABS(min),ABS(max)) of the data to be contoured. Contour levels selected with LEVELS are multiples of P1 percent of M. E.g. if P1 = 10 and LEVELS 1 2 5 9, the contour levels are 10, 20, 50 and 90% of the peak value.

6.15.3 RGMAP /ABSOLUTE

```
[GREG2\]RGMAP /ABSOLUTE A1
```

Contour levels are multiples of A1, e.g. for LEVELS 1 2 3, the contour used will be A1 2*A1 3*A1.

6.15.4 RGMAP /BLANKING

```
[GREG2\]RGMAP /BLANKING Bval Eval
```

Override the current blanking value as defined by command SET BLANKING. Contours are not drawn in pixels touching pixels with value V so that ABS(V-Bval) <= Eval . This is extremely useful for regular grids only partially filled as often occur in astronomy.

6.15.5 RGMAP /GREY

```
[GREG2\]RGMAP /GREY [start [step] [background]]
```

On graphic displays with area filling capabilities, fills contours with up to 16 different color pens (but the number of contours is not limited to 16). These pen colors may be changed in the same manner as for the pixels colors (see HELP LUT /PEN). "start" (default 0) and "step" (default 1) may be used to change the exploration of the 16-pen palette. "background" is the number of a color used to fill the map before contour-filling, thus providing a user-defined background for the contour-filled map.

In case of Filling problems, see the subtopic MEMORY_OPTIONS of this HELP.

6.15.6 RGMAP /KEEP

```
[GREG2\]RGMAP /KEEP File_Name [Blank]
```

Will keep a formatted copy of the coordinates of the contour points in the specified file. This file can later be used as an input file for COLUMN. X coordinates are in column 1 and Y coordinates in column 2. The Blank value is written as a separator value for later CONNECT/BLANKING Blank. Blank defaults to the SET BLANKING value if it exists, else Blank is mandatory.

6.15.7 RGMAP /PENS

```
[GREG2\]RGMAP /PENS [Pos Neg]
```

Selects the pens for positive and negative contours. Three modes of operation are possible :

- Without option /PENS, positive contour levels use pen 0 and negative contour levels use pen 15. Positive means that the value given in command LEVELS is positive, not the final contour level value which is affected by a possibly negative factor.
- With option /PENS, all contours use the current pen.
- With /PENS Pos Neg, positive contours use pen number Pos and negative contours use pen number Neg.

6.16 STRIP

```
[GREG2\]STRIP [X1 Y1 X2 Y2]
```

Will extract from the current Regular Grid array a strip going through points (X1,Y1) and (X2,Y2) in User coordinates. If no arguments are present, it will call the cursor to set the two points. Although the end points need not be on the map boundaries, STRIP will always produce a full strip across the map.

For strip parallel to the X or Y map axis, the nearest pixel is used to fill the values. For any other strip orientation, if the tangent of the angle is less than (NY-1)/(NX-1) (i.e. if you scan the X map axis faster than the Y map axis), the number of resulting points on the strip profile will be NX and the user coordinates along the strip will be the map X coordinate. For each pixel, interpolation along the Y axis is done using the three nearby Y pixels. A similar situation with X replacing Y occurs if the angle is larger.

The map user coordinates are put into the X buffer, and map values into the Y buffer. The strip can then be plotted using CONNECT, but remember to set decent limits before.

6.17 WEDGE

```
    [GREG2\]WEDGE [Top|Bottom|Left|Right [Size]]  [/SCALING  Lin|Log|Equ
Min Max] [/LEVEL [Label] ]
```

Will plot a WEDGE along the current BOX. WEDGES are used after a PLOT command to show the relationship between pixel hues and image values.

The default position is Right, i.e., on the right side of the box, but the other location and size modifiers are self-explanatory. The SCALING defaults to the current scaling of the last image PLOTted, but can be changed (as in command PLOT) with the /SCALING option.

The /LEVEL option will draw levels (as defined by the LEVEL command) at their appropriate location on the wedge. The modifier Label will label the wedge axis with the level values. This is the default (i.e., some levels should be defined) for the EQUALIZATION Mode, for which the relationship between pixel hues and image values is strongly non-linear.

6.18 WRITE

```
    [GREG2\]WRITE Item File_Name [/TABLE [X Nx] [Y Ny] [Z Nz]]
```

This command is used to saves some GreG internal buffers on an output file for later processing. The item specify what is to be saved, and the format of saving, and can be :

- POLYGON Saves the current polygon in formatted way on two columns easily readable by command COLUMN or POLYGON later. The default extension is .POL.
- RGDATA Saves the current Regular Grid array on a RGDATA-like

```
    output  file  (to  preserve  the result of a masking, or of a random
    mapping interpolation). The default extension is .DAT.
  - IMAGE      Same as above but uses the more  efficient  and  complete
    IMAGE data format of GILDAS. The default extension is .GDF.
  - COLUMN     Saves the current X Y Z buffers  on  a  formatted  output
    file (after a SORT or a STRIP for example). The default extension is
    .DAT.

[GREG2\]WRITE COLUMN File_Name [NEW]/TABLE [X Nx] [Y Ny] [Z Nz]

also saves the X Y Z buffers but on column  Nx,Ny,Nz  of  the  specified
output  table (GILDAS format). A new table is created if argument NEW is
specified, otherwise the table must already exists.  The  table  may  be
expanded  if  any  of Nx,Ny or Nz exceed the table column number ; exact
match on the number of lines is required. The  tables  can  be  read  by
command COLUMN/TABLE.
```

6.19 END

7 GTVL Language Internal Help

7.1 Language

```
                      GTVL\ Language Summary
CHANGE          : Change some attributs of the plot
CLEAR Arg       : Clear some (parts of) the plot or windows.
COMPRESS        : Remove invisible parts of the plot.
CREATE Arg      : Create directory, windows or LUTs
DEVICE          : Open the graphic device
DISPLAY         : List some parameters of the plot
EXPORT File     : Save part of the plot to metacode file, for further use
HARDCOPY        : Get a printed copy of the plot
IMPORT File     : Insert an external plot into the current one
LUT [arg]       : Change the color Look-Up-Table
REPLICATE       : Copy a directory to another place
ZOOM            : ZOOM or REFRESHES the plot (possibly in another window)
```

7.2 CHANGE

```
    [GTVL\]CHANGE Item SegName Value [Val2 [Val3]]

Change some attribute of the Graphic Library. Three  items  concern  the
current directory and active windows:
  - CHANGE DIRECTORY DirName [WinNum] change the  default  directory  to
    DirName,  and optionally the active window number to WinNum (instead
    of 0) within this directory.
  - CHANGE WINDOW WinNum [NOREFRESH|REFRESH] change  the  active  window
    number  to  WinNum,  and  optionally  change  its  automatic refresh
    behaviour.
  - CHANGE ZOOM NEW|CURRENT change the default ZOOM command behaviour.
  - CHANGE CLEAR  TREE|WHOLE  change  the  default  CLEAR  PLOT  command
    behaviour.

Two items are image segment attributes:
  - CHANGE  SCALING  SegName  LIN|LOG  [Lcut  [Hcut]]  will  change  the
    transfer function for the specified image segment.
  - CHANGE BLANKING SegName BLACK |  WHITE  |  Colour  will  change  the
    colour  of blanked pixels (if any) in the displayed image. Colour is
    a colour value as in PENCIL command (1-8).

Other items are general  segment  attributes:  WEIGHT,  DASHED,  COLOUR,
DEPTH, VISIBILITY (ON or OFF). For these items, if SegName is a segment,
only that segment is modified, but if it is a  directory,  all  segments
attached to this directory are modified accordingly.
```

7.2.1 CHANGE SEGMENT_NAMES

The 'SegName' used in all the CHANGE commands is formed by a name, given by the calling high-level program (usually a reminder of the command used), followed by an unique, incremental, number. To refer a particular segment, one uses the full name (as in CHANGE VISIBILITY CONTOUR:46 OFF). To refer to ALL the segments with the same name, omit the number (as in CHANGE VISIBILITY CONTOUR OFF). To refer to a range of segments, use a 'minus' sign in the range (as in CHANGE VISIBILITY CONTOUR:32-46 OFF). Segments names can be 'wildcarded' (an '*' can replace part of the beginning or the end of the segment name, as in CHANGE VISIBILITY *TOUR OFF).

Trick: A reverse range of segments (as in CLEAR SEGMENT 32-12) is valid. In the particular case of the above example, it is equivalent to the command CHANGE VISIBILITY 32-12 CLEAR wich in turns tries to delete (as opposed to change the visibility) the segments (which is possible if segment 32 is the last segment plotted).

7.3 CLEAR

```
CLEAR [Arg]
```

CLEAR withous arguments clears the current plot and ask the user for confirmation. In non interactive session, it clears the plot without prompting.

Please consult the following sections for the various CLEAR arguments:

7.3.1 CLEAR ALPHA

CLEAR ALPHA erases the alphanumeric screen alone, without affecting the plot on old graphic terminals. Under X-Window, the graphic screen created by the 'DEVICE' command is raised up.

7.3.2 CLEAR DIRECTORY

CLEAR DIRECTORY [DirName] makes the named directory (or by default the current directory) invisible. All the attached structure becomes invisible. Under X, the graphic screen is updated, elsewhere use the command ZOOM REFRESH to do it. Note that the directory is not destroyed: you can recover it by a CHANGE VISIBILITY command.

7.3.3 CLEAR GRAPHIC

CLEAR GRAPHIC lowers the graphic windows, popping up the alpha screen (X window only).

7.3.4 CLEAR PLOT

CLEAR PLOT destroys the current tree or the whole plot, and clears the graphic screen accordingly. The default behaviour can be controlled by command CHANGE CLEAR.

7.3.5 CLEAR SEGMENT

CLEAR SEGMENT destroys the last graphic segment, i.e. the part of the plot corresponding to the last graphic command. Can be used repetitively. The Graphic screen is updated under X-Window, elsewhere use command ZOOM REFRESH to do it.

CLEAR SEGMENT SegName makes the named segment invisible.

One can erase a range of graphic segments in a single command by typing a range of segment numbers, as in CLEAR SEGMENT 24-16. The use of reverse range (24-16) forces the graphic library to actually delete the segment, if possible, instead of making it invisible. See command GTVL\CHANGE for details.

7.3.6 CLEAR TREE

CLEAR TREE clears the current tree only, leaving all other parallel trees intact. All associated windows are destroyed, and the tree is is recreated empty, with a single attached window. If only one tree exist, a CLEAR WHOLE is performed.

7.3.7 CLEAR WHOLE

CLEAR WHOLE destroys the whole plot. On X-Window displays, all windows are destroyed, and a new window is recreated for the first tree. Other trees are destroyed.

7.3.8 CLEAR WINDOW

CLEAR WINDOW windownumber deletes the current graphic window (if possible). Only supported on multi-window displays (X-Window).

7.4 COMPRESS

```
[GTVL\]COMPRESS
```

Compress the internal metacode, by removing (definitely) all invisible segments and directories.

7.5 CREATE

```
[GTVL\]CREATE Argum Value
```

- CREATE DIRECTORY PathName
  Create a new directory. PathName syntax follows "Unix-like" pathnames, with the use of token '<' instead of '/'. Constructs like ..<DIR are valid. To create a new top directory (called e.g. NEWTOP), use the following syntax:
  ```
  CREATE DIRECTORY <NEWTOP
  ```
- CREATE LUT
  Store the current Look Up Table into the internal metacode. Stored LUT will be re-used for PostScript hardcopies, allowing to use more than one color palette on PostScript printout.
- CREATE PENLUT
  Store the current pencil colors into the internal metacode. Stored pencil LUTs will be re-used for PostScript hardcopies, allowing more than 16 filled colors on the same printout.
- CREATE WINDOW [BLACK|WHITE] [[NOREFRESH] [NAME Name] [/PIXEL Px Py]
  Create a new window with the corresponding attributs of background colour, automatic refresh behaviour and Name. Option /PIXEL specifies the window size (in pixels); if not present, the cursor is used to define the zooming area.

7.6 DEVICE

```
[GTVL\]DEVICE Dev_Type [Mode] [/OUTPUT Dev_Name]
```

Directs graphic output to the graphics device of the specified type and sets default values for the plot region location and device physical limits. The basic unit is 1 cm for hardcopies. All plots to interactive devices are made with some pre-computed factor to match the screen size to the Plot Page size, defined by command SET PLOT_PAGE (default size is an A4 format page, 30 by 21) .

Use DEVICE ? to get the list of recognized graphics devices or DEVICE type ? to get the list of modes on a given device.

/OUTPUT Dev_Name Allows you to direct the graphic output to something else than your own terminal (TT:). Note that some specific devices will not support this option: this is currently the case for X-Window devices.

Bitmap color is only available with DEVICE IMAGE on X-Window terminals.

7.7 DISPLAY

```
[GTVL\]DISPLAY Item [Argument [s]] [/OUTPUT]
```

```
List the internal parameters of the named item
  DISPLAY DIRECTORY  shows the working directory. Symbol  PWD  does  the
    same...
  DISPLAY SEGMENT  shows the attributes of segments attached to  current
    directory
  DISPLAY TREE      shows the directory tree from current point.
  DISPLAY  WINDOW  shows  the  list  of  windows  and  their  associated
    directories.
etc... for other stuff to be added. Used for debugging mostly.
```

7.8 EXPORT

```
[GTVL\]EXPORT FileName [DirName]
```

```
Export the current (or specified) directory tree to an external metacode
file  of  name FileName. The default extension is .meta, and these files
have a different format than the old .vec files. The sub-plot can  later
be  reincorporated  using  command  IMPORT.  This allows merging several
plots together. This command does an implicit COMPRESS before saving the
tree, so that invisible subtrees will be deleted.
```

7.9 HARDCOPY

```
[GTVL\]HARDCOPY [File] [/DEVICE FirstName SecondName]
[/PLOT Destination ] [/LANDSCAPE] [/PORTRAIT] [/EXACT]
```

```
Creates (and optionally plots) a "hardcopy"  of the current  plot.   The
plot  is not destroyed  and  can be appended later by new plot commands.
The default behaviour is to scale the GreG PLOT_PAGE (centimeters)  into
the  output  physical  page.   This  behaviour  can be overridden by the
/EXACT option,  and  in  a  restricted  manner  by  the  /LANDSCAPE  and
/PORTRAIT options.

The /DEVICE option first argument specifies which type of  plotter  will
be  used.  The same devices recognized by command DEVICE are allowed for
this option. PrimaryName usually indicate the graphic protocol supported
by the device, and SecondaryName a variant of this protocol. For example
               HARDCOPY File /DEVICE PS COLOR
is used to create a Postscript file with full color support.  The device
First  and  Second  names  are  defined  in the file "GAG_TERMINAL" that
describes, in binary format, the protocol and options  relative  to  all
the  devices  locally  supported.  Changes  to this file are done by the
system manager using the gtvdef program.
```

```
The /PLOT option is used to force immediate plotting and, optionaly,  to
control which plotter should be used. Plotter is the plotter name, which
defaults to the content of the logical name "GAG_PLOTTER".  If  no  file
name  was  specified  for  an  immediate  plotting, the metacode file is
deleted after plot completion, otherwise it is kept.

Default  type  and  destination  can  be  controlled  by  logical  names
GAG_HARDCOPY   and   GAG_PLOTTER,   in   the   user,   local  or  global
dictionnaries.
```

7.9.1 HARDCOPY /DEVICE

```
    The /DEVICE option first argument specifies which  type  of  plotter
will  be used. The same devices recognized by command DEVICE are allowed
for this option. PrimaryName indicates the graphic protocol supported by
the  device,  and  SecondaryName  a  variant of this protocol. Among the
available device types are the following
   PS FAST         PostScript with hardware pen thickness handling
                   Clipping of wide pens may be inaccurate.
   PS GREY         PostScript with fixed greyscale transfer function for
                   bitmap. Clipping is accurate.
   PS COLOR        PostScript with user controlled color transfer
                   function. A greyscale with equivalent luminosity is
                   is used on non-color printers.

   EPS FAST, EPS GREY, EPS COLOR
                   As PS, but for "Encapsulated PostScript", in which
                   the BoundingBox is computed from the plot boundaries
                   rather than from the Plot Page size.

   HPGL            HPGL language for HP printers.
```

7.9.2 HARDCOPY /LANDSCAPE

```
    The output plot will be in LANDSCAPE mode.  Not  available  on  all
types of hardcopy devices.
```

7.9.3 HARDCOPY /PORTRAIT

```
    The output plot will be in PORTRAIT mode. Not available on all types
of hardcopy devices.
```

7.9.4 HARDCOPY /EXACT

```
    The EXACT option is used to preserve the size of the plot (1  cm  in
GreG = 1 cm of the plotter). Available only on PostScript devices
```

7.10 IMPORT

```
[GTVL\]IMPORT File [DirName] [/PATH PathName]
```

```
Insert an external metacode file (type .meta, not the old .vec files) at
the specified PathName into the current plot. The directory name DirName
is derived from the filename by default.
```

7.11 LUT

```
[GTVL\]LUT Argum [/PEN]
```

```
Modify the bitmap or pen look up table. Option /PEN indicates to  modify
the  the  value  of  the  16 user-controlled pen colors (colors 8 to 23)
rather than the bitmap look-up table.

Possibles arguments are:
  - COLOR Load a default color table
  - WHITE Load a black and white color table  (White  background,  black
    sources)
  - BLACK Load a black and white color table  (Black  background,  white
    sources).
  - LUT Load the color table defined by an HSV color model. (HSV  stands
    for  HUE,  SATURATION and VALUE). There are three user-writeable SIC
    Variables of name HUE, SATURATION and VALUE, that  contain  the  HSV
    model  of  the  current LUT. These array's values can be manipulated
    using standard SIC mathematic functions to create new Lookup  Tables
    (see below).
  - Any other : "LUT Argum" will read Red Green  Blue  values  from  the
    formatted file named "Argum".

The default argument is LUT. A convenient value for the HUE array is
  LET HUE[I] = 256-2*I !a "classical" color table from blue  to  red  as
    used in AIPS for example.
    Color contours can be obtained using variations around the following
    command:
  LET HUE[I] = 16*INT((256-2*I)|16)
    and so on with the most funny functions you may invent.
CAUTION : the HUE is a real number between 0 and 360.0

Note that a similar behaviour is defined for the  pen  colors  with  the
arrays  P_HUE,  P_SATURATION  and P_VALUE (the colors of the 16 contour-
filling pens), and for the BLANKING  color  (the  color  used  to  paint
blanked  pixels  in  images),  with  the 3 reals B_HUE, B_SATURATION and
B_VALUE.
```

7.11.1 LUT /PEN

Same behaviour as LUT, but will update the colors of the 16 pens used for contour-filling.

7.12 REPLICATE

```
REPLICATE Dir1 Dir2
```

Copy Dir1 subtree to Dir2 subtree. Not very useful now, but will become interesting when shifting, stretching, rotations, and user coordinate changes will be allowed for all segments.

7.13 ZOOM

```
[GTVL\]ZOOM [Argument(s)] [/NEW] [/CURRENT] [/REGION]
```

```
ZOOM can be used to modify the viewing window of the current plot.  The
zooming does not affect the plot, but only the way it is displayed.
  - ZOOM X1 X2 Y1 Y2 Defines the viewing window by its corners
  - ZOOM OFF Turns off zooming, i.e. defines the viewing  window  to  be
    the plot page
  - ZOOM REFRESH Redraw the current plot (useful after CLEAR SEGMENT)

Without arguments ZOOM calls the cursor to define a new viewing  window.
This  new  window will be defined by its center (at the cursor position)
and a zooming factor (by steps of 1.414) :
   Z    Increase the zooming factor
   -    Decrease it
   B    Display the viewing window boundaries
These commands do not produce effective zooming, press the space bar
to do it. Additional commands :
   A    Clear alphanumeric screen
   O    Display the whole plot page (ZOOM OFF).
        <Space> returns to previous zooming factor.
   E    Exits from the routine.
For X11 displays, you may use the mouse buttons for the commands Z (left
mouse  button),  <Space>  (middle  mouse  button),  and  E  (right mouse
button).

Option /NEW and /CURRENT can be used to  specified  whether  the  zoomed
area appears in the special zoom window (/NEW) or in the original window
(/CURRENT). Default behaviour is specified by  CHANGE  ZOOM  NEW|CURRENT
command.
```

7.13.1 ZOOM /REGION

This option uses a different ZOOM cursor, a rectangular area drawn using the midle button of the mouse. Clicking outside the area with the middle button will resize the rectangle. By keeping the middle button pressed, the shape will follow the mouse. To drag the shape on another part of the screen, click with the middle button inside the shape and keep the button pressed while dragging the shape. Button 1 will perform the zoom (in an auxiliary zoom window), button 3 exits.

8 GreG Plot Library

Using **GreG** as a plot library is now described in the **GILDAS** programming guide.

9 GreG Character Set

This section includes a copy of the character set and a commented procedure to generate it. It is useful to give the correspondence between the keyboard and special letters in the **GreG**ś character set.

In the roman alphabet, all keyboard keys will give the corresponding ASCII characters with only 4 exceptions :

```
- The counter quote ‘ will give the double quote "
- The circumflex accent ˆ will give the degree sign
- The double quote cannot be used because of SIC character handling
- The backslash \ cannot be used because of GreG character handling
```

The Script character set has an exact correspondance for all letters and figures, and mathematical symbols corresponding to special characters.

The Greek character set has a rather convenient correspondance for all letters (this is not the alphabetic order which is highly non mnemotechnic). Planets symbols correspond to the figures (0 Sun, 1 Mercury and so on). Other astronomical symbols correspond to the special characters.

The procedure `FONT.GREG` (with the satellite procedure `F.GREG`) generates the three **GreG** alphabets in simplex and double quality and produce eight metacode files named `FONT1.VEC, ..., FONT8.VEC`. These files are stored in the directory `GAG_DEMO:`. You can plot them on paper using any of the metacode translators available on your site (`HPXY, LWXY, PLXY,` ...) or just invoke the `ZOOM` utility to have a glance at them on an interactive screen.

Satellite procedure `F.GREG` to generate the **GreG** fonts

This procedure `F.GREG` is called by the `FONT.GREG` procedure to generate the three alphabets in Simplex and Duplex quality :

```
!
! Draws a character in all GreG fonts and quality
! Parameter &1          Y position of character
! Parameter &2          Character to be drawn
!
DRAW TEXT  3 &1 "\\1&2"        ! Roman font, Simplex quality
DRAW TEXT  6 &1 "\\1\\G&2"     ! Greek font, Simplex quality
DRAW TEXT  9 &1 "\\1\\S&2"     ! Script font, Simplex quality
DRAW TEXT 13 &1 "\\2&2"        ! Roman font, Duplex quality
DRAW TEXT 16 &1 "\\2\\G&2"     ! Greek font, Duplex quality
DRAW TEXT 19 &1 "\\2\\S&2"     ! Script font, Duplex quality
```

Procedure FONT.GREG to generate the **GreG** fonts

```
!
! Generates the 6 alphabets of GreG fonts in 6 columns
!      1         2         3             4        5        6
!    Roman     Greek     Script        Roman    Greek    Script
!    Simplex   Simplex   Simplex       Duplex   Duplex   Duplex
!
SIC VERIFY ON
SET PLOT_PAGE PORTRAIT
SET BOX 0.5 20.5 1 28
SET CHARACTER 1.5
SET COORDINATE BOX
SYMBOL F "@ GAG_DEMO:F.GREG"
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 "   "     ! Space         Space                   Space
F 22 " ! "     ! !             Aries                   Existencial
F 20 " ‘ "     ! Double quote  Taurus                  Up-Down arrow
F 18 " # "     ! #             Gemini                  Included in
F 16 " $ "     ! $             Cancer                  Reunion
F 14 " % "     ! %             Leo                     Includes
F 12 " & "     ! &             Virgo                   Intersection
F 10 " ’ "     ! ’             Libra                   Belongs to
F  8 " ( "     ! (             Scorpius                Integral
F  6 " ) "     ! )             Sagittarius             Circular Integral
F  4 " * "     ! *             Capricornus             Multiply
F  2 " + "     ! +             Aquarius                Plus/Minus
HARDCOPY GAG_DEMO:FONT1.VEC
CLEAR PLOT
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " , "     ! ,             Pisces                  Divide
F 22 " - "     ! -             Space                   Minus/Plus
F 20 " . "     ! .             Square                  .
F 18 " / "     ! /             Space                   Square root
F 16 " 0 "     ! 0             Sun                     Zero
F 14 " 1 "     ! 1             Mercury
F 12 " 2 "     ! 2             Venus
F 10 " 3 "     ! 3             Earth
F  8 " 4 "     ! 4             Mars
F  6 " 5 "     ! 5             Jupiter
F  4 " 6 "     ! 6             Saturn
F  2 " 7 "     ! 7             Uranus
HARDCOPY GAG_DEMO:FONT2.VEC
CLEAR PLOT
```

```
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " 8 "     ! 8             Neptun
F 22 " 9 "     ! 9             Pluto
F 20 " : "     ! :             Moon quarter            Space
F 18 " ; "     ! ;             Comet                   Perpendicular
F 16 " < "     ! <             Eight branch star       Less or equal
F 14 " = "     ! =             Ascending node          Equivalent
F 12 " > "     ! >             Descending node         Greater or equal
F 10 " ? "     ! ?             Space                   Different
F  8 " @ "     ! @             Space                   Proportional
F  6 " A "     ! A             Alpha
F  4 " B "     ! B             Beta
F  2 " C "     ! C             Chi
HARDCOPY GAG_DEMO:FONT3.VEC
CLEAR PLOT
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " D "     ! D             Delta
F 22 " E "     ! E             Epsilon
F 20 " F "     ! F             Phi
F 18 " G "     ! G             Gamma
F 16 " H "     ! H             Eta
F 14 " I "     ! I             Iota
F 12 " J "     ! J             Nabla (Gradient)
F 10 " K "     ! K             Kappa
F  8 " L "     ! L             Lambda
F  6 " M "     ! M             Mu
F  4 " N "     ! N             Nu
F  2 " O "     ! O             Omicron
HARDCOPY GAG_DEMO:FONT4.VEC
CLEAR PLOT
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " P "     ! P             Pi
F 22 " Q "     ! Q             Theta
F 20 " R "     ! R             Rho
F 18 " S "     ! S             Sigma
F 16 " T "     ! T             Tau
F 14 " U "     ! U             Upsilon
F 12 " V "     ! V             Space
F 10 " W "     ! W             Omega
F  8 " X "     ! X             Xi
F  6 " Y "     ! Y             Psi
F  4 " Z "     ! Z             Zeta
F  2 " [ "     ! [             Dagger                  Approximately
HARDCOPY GAG_DEMO:FONT5.VEC
```

```
CLEAR PLOT
```

```
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " \ "     ! Unaccessible because this is the " escape" character
F 22 " ] "     ! ]             Double dagger           Approximately
F 20 " ^ "     ! Degree        Paragraph               Chemical eq.
F 18 " _ "     ! Space         Three dots              Left/Right arrow
F 16 " ` "     ! Double quote  Taurus                  Up/Down arrow
F 14 " a "     ! a             alpha
F 12 " b "     ! b             beta
F 10 " c "     ! c             chi
F  8 " d "     ! d             delta
F  6 " e "     ! e             epsilon
F  4 " f "     ! f             phi
F  2 " g "     ! g             gamma
HARDCOPY GAG_DEMO:FONT6.VEC
CLEAR PLOT
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " h "     ! h             eta
F 22 " i "     ! i             iota
F 20 " j "     ! j             Partial derivative
F 18 " k "     ! k             kappa
F 16 " l "     ! l             lambda
F 14 " m "     ! m             mu
F 12 " n "     ! n             nu
F 10 " o "     ! o             omicron
F  8 " p "     ! p             pi
F  6 " q "     ! q             theta
F  4 " r "     ! r             rho
F  2 " s "     ! s             sigma
HARDCOPY GAG_DEMO:FONT7.VEC
CLEAR PLOT
!
!    Keyboard    ROMAN FONT    GREEK FONT              SCRIPT FONT
F 24 " t "     ! t             tau
F 22 " u "     ! u             upsilon
F 20 " v "     ! v             Infinite
F 18 " w "     ! w             omega
F 16 " x "     ! x             xi
F 14 " y "     ! y             psi
F 12 " z "     ! z             zeta
F 10 " { "     ! {             Large {                 Up arrow
F  8 " | "     ! Vertical bar  Large integral          Right arrow
F  6 " } "     ! }             Large }                 Down arrow
F  4 " ~ "     ! ~             Large root              Left arrow
HARDCOPY GAG_DEMO:FONT8.VEC
EXIT
```

Index