GPR_DISP

Version: 2.11.06.01
Last revision date: 11-6-2001
Description: GPR_DISP displays one or more ground penetrating radar (GPR) files to the CRT as an 8-bit gray-scale image or as wiggle traces and optionally saves the display to disk as an Encapsulated PostScript (EPS) file or as a PCX graphics file. The primary purpose is to visualize GPR data before and after manipulation and to produce graphics files that are suitable for printing and publication. The best print quality will be with EPS files. These files, however, are not readily read by graphics or paint programs, except for Corel PhotoPaint. EPS files are gray scale only. PCX files are a copy of the screen image - B&W or color. See details below. Press the Esc key to end the program after the image is displayed.

The input to GPR_DISP.EXE is a "CMD" file, an ASCII text file containing keywords (or commands or parameters) describing how to display the radar data. An optional "override" or "global" command file may be specified on the DOS command line that will take precedence over values found in the other command files.

The GPR data can be read from disk using the following formats:
- GSSI SIR-2, SIR-2000, and SIR-10 binary "DZT" files,
- Sensors and Software pulseEKKO "DT1" and "HD" files, or
- Society of Exploration Geophysicists SEG-Y formatted files.

If the storage format does not conform to any of the above or GPR_DISP has trouble reading the file correctly, there are options for the user to supply the required information.

A message file called GPR_DISP.LOG is opened when the program starts. It is located either in the directory where the program was called from or in the root directory of drive C. In multitasking environments, this may prevent more than one session of the program from executing in the same directory. The log file may contain more information regarding the failure or success of GPR_DISP as it executes. Sessions are appended at the end of the log file.

IMAGE/DATA PROCESSING

When the GPR data are read into this program, they are immediately converted to 8-bit unsigned integers (values 0 to 255). This may cause some unexpected results if the GPR data are 16 bit but have a low dynamic range. Sixteen-bit numbers are divided by 256 to get 8-bit numbers. If the data amplitude range is some low multiple of 256 only a few values will survive the conversion. In cases when the data range is less than 256 then 1 or no values may survive. The program GPR_PROC can change the data amplitudes before calling GPR_DISP.

The CRT image presents the GPR data in one of five modes: trace/time, time/time, distance/time, distance/distance, or trace/sample.
- Trace/time images simply display the data as adjacent traces (horizontal axis is trace number and vertical axis is sample time in nanoseconds, ns).
- Trace/sample images display the data as adjacent traces, except the vertical axis is the sample number (rather than sample time).
- Time/time images display the traces as equally located in time horizontally (seconds) and vertically (ns).

- Distance/time images place the GPR traces in the appropriate location along the horizontal axis (in meters), with no elevation correction. The vertical axis is sample time, or travel-time, in ns.
- Distance/distance images have been geometrically adjusted so traces are placed "correctly" in 2-D space with a (multi-)layer time-to-depth conversion applied (X- and Z-axes are in meters).

Some fundamental image processing operations are available to refine the presentation of the data.

- Point processes (using look-up tables):
  - image contrast stretching (either of data or using look-up table) to enhance subtle features
  - image contrast compression to enhance strong reflectors
  - local contrast stretching (the endpoints of the gray scale remain the same but the rate of change of the middle portion is either greater or lessor than 1 to 1)
  - image brightening or darkening
  - negative of image
  - EPC recorder-style images (white is at the midpoint value and black occurs at both endpoints of the 8-bit range, with a gray-scale between)

- Geometric processes
  - scaling (with interpolation or elimination if necessary)
  - mirroring (data from disk may be displayed in reverse order and/or "upside down")

Some Hilbert transform processes are available. The instantaneous amplitude and instantaneous power can be calculated and displayed as calculated or wrapped by an envelope.

"Background" removal is available. An average radar trace is calculated by adding all traces in the "window" together, sample by sample, and then dividing each sample by the number of traces. This average trace is then normalized to have zero as the midpoint and is subtracted from each trace in the "window". This operation may not be appropriate for some data sets, especially those with few traces (a few hundred or less) or with outstanding "horizontal" features.

Data amplitude gain modification is available. Gain may be removed or added or both. The user supplies a set of gain values in decibels as 20 x log(ratio), where log is to base 10. For example, to multiply data by 1000, a decibel value of 20 x log(1000) or 60 is used. To decrease data by 10 (i.e. multiply by 0.10), a decibel value of 20 times -1 or -20 is used. A decibel value of 96 is approximately equivalent to a ratio of 65535:1 (or 2^16, the dynamic range of 16-bit data).

For example, to multiply all amplitudes by 2 use the following keywords.

```
change_gain = "TRUE"
num_gain_on = 2
gain_on[]   = 6 6     ; 20 x log(2) = 6
```

For users of GPR_PROC and GPRSLICE, GPR_DISP allows you to visualize data manipulations before creating a new data file or slicing a group of files.

For velocity analysis, a small-spherical-object reflection hyperbola can be overlaid on the image. Antenna separation, object horizontal location, depth, and radius, and the GPR wave velocity must be supplied. The GPR velocity is in m/ns. If you know the RDP, then take the square root of the RDP and divide that into 0.2998 m/ns (the velocity of light in a vacuum) to get the velocity.

For example, a RDP of 5.5 is equivalent to a velocity of 0.128 m/ns [0.2998 / SQRT(5.5)]. A velocity of 0.10 m/ns is equivalent to a RDP of 8.9 [SQR(0.2998/0.10)]

GRAPHICS FILES

Images can be stored optionally to disk using the Encapsulated PostScript level 2 storage format, in either landscape or portrait paper orientation, or using the PCX graphics file format as black and white or color images. Once the data are displayed to the screen, pressing the following keys can change the color/gray scale.
   G (gray scale - the default),
   S (spectrum),
   R (reverse color/gray scale),
   1 (custom color palette 1),
   2 (custom color palette 2),
   3 (custom color palette 3).

Only PCX files will preserve the color scale. EPS files are gray scale only. PCX files are a direct copy of the pixels shown on the screen - image or wiggle trace. The quality of the letters/numbers will not be as nice as with EPS output using the Hershey fonts. Make sure that the root directory on the C: drive and the drive you are running GPR_DISP from has a directory named “hershey” and that it contains the font files. These files are available with this report.

NOTE: To obtain a PCX file of only the GPR data, set the viewport parameters to the full limits of the screen and do not add axes, title, or labels.
    SET: vx1 = 0.0   vx2 = 133.333   vy1 = 0.0   vy2 = 100.0

THE KEYWORDS

Following is the list of keywords and their default values. The documentation format is:
“KEYWORD: **keyword** = default value”.
Look at GPR_DISP.CMD for the exact format for setting keywords.

############ NOTE #############
The file GPR_DISP.CMD has most comments stripped out, and GPR_DISP.CM_ has all comments removed. Currently there are about 180 keywords or commands. There is only one keyword that is required to be in the keyword file. It is "dat_infilename". If the rest of the keywords are missing or assigned default values, the GPR data will be displayed on the CRT as trace number versus sample time. There will be no title or axes, no manipulation of the data, and no output graphics file. Once you create a few CMD files, setting them up won't seem so daunting. Yes, we're working on an interactive graphical interface for our GPR programs.
################################

*************************** PROGRAM CONTROL *******************************
These keywords are used only from the FIRST command file. Press the Esc key to end the program after the image is displayed.

KEYWORD: **batch** = "FALSE"
Place program in batch mode (no pauses) if "TRUE". If set to “FALSE”, the program will normally pause at certain points and before ending.

KEYWORD: **display_all** = "FALSE"
Set to "TRUE" to display keyword values for all command files, otherwise only values for the first command file are displayed

KEYWORD: **display_none** = "FALSE"
Set to "TRUE" to suppress displaying keyword values when program starts up.

********************* TO DISPLAY MULTIPLE GPR FILES ***********************
This program can display more than one GPR file on the CRT. If there is only one GPR data set to display leave the next keyword assigned to an empty string or remove the equal sign. If there is another GPR file to display, then insert the name of another command file between the quotes. For a third, etc., place the third command file name after this keyword in the second command file. In this way, command files are chained together. Be sure to assign viewport coordinates in each command file as appropriate. Also, be sure to leave this keyword blank (or delete the equal sign) in the last command file!

KEYWORD: **next_cmd_filename** = ""

********************* SPECIFICATION OF INPUT DATA ************************
One data file can be input for each command file. This is the only keyword that is required to be in the command file. The data storage format is determined by inspecting the file. If the program cannot recognize a flavor of the three formats below then an error message may be issued.

Recognized storage formats are:
  DZT - GSSI SIR-2, SIR-2000, and SIR-10
  DT1 - Sensors & Software pulseEKKO
  SGY - SEG SEG-Y

DT1 and HD files are assumed paired, i.e. both have the same filename with different extensions. So, if a data file with a ".DT1" extension is specified, the ".HD" filename will be assumed. Only DT1/HD files must have those filename extensions. GSSI and SEG-Y files can have any extension.

KEYWORD: **dat_infilename** = ""

RAMAC and user-defined data files can be read by assigning correct values to the next five keywords.

########### NOTE ############
IF the GPR format DOES NOT CONFORM to any of the above formats then the next six parameters (other_format, file_header_bytes, trace_header_bytes, samples_per_trace, total_time, and input_datatype) MUST be specified. Otherwise, IGNORE THEM. If you want to convert the storage format then GPR_CONV is the program to use. GPR_INFO will report this basic information for recognized storage formats.
##############################

KEYWORD: **other_format** = "FALSE"
Replace with "TRUE" if you want to use the next five parameters to specify the input format.

KEYWORD: **file_header_bytes** = 0
Replace with number of bytes in the file header. PulseEKKO data files do not have a file header - the information is held in another file with a .HD extension. GSSI files have either a 512-byte (old style) or 1024-byte (current style) header. However, DZT files can have up to 4 file headers - one for each channel. SEG-Y files have a 3600-byte header. RAMAC data files have no file header.

KEYWORD: **trace_header_bytes** = 0
Replace with number of bytes in each trace header. For pulseEKKO files, a 128-byte header precedes each GPR trace. For GSSI files, no header precedes each trace, but the first 2 samples (not necessarily bytes) are reserved. SEG-Y files have a 240-byte trace header. RAMAC data files have no trace headers.

KEYWORD: **samples_per_trace** = 0
Replace with the number of samples per trace. For pulseEKKO data, the number of samples per trace is recorded in the HD file (NUMBER OF PTS/TRC). For GSSI data, the number of samples per trace is a power of 2, from 128 to 2048, typically 256, 512, or 1024. The information is recorded in the .DZT file header in the rh_nsamp field. For RAMAC files, the RAD text file records the number of samples. For SEG-Y files, look in the comment area of the file header.

KEYWORD: **total_time** = 0
Replace with total number of nanoseconds per trace. For pulseEKKO data, look at the "TOTAL TIME WINDOW" field in the .HD file. For GSSI data the value is recorded in the file header. For SEG-Y files, look in the comment area of the file header. For RAMAC files, the TIMEWINDOW parameter records the time per trace in microseconds (multiply by 1000 to get ns).

KEYWORD: **input_datatype** = 0
This defines the type of input data element. Replace with one of the following element types:

```
 1 for 1-byte signed characters
-1 for 1-byte unsigned characters            (GSSI)
 2 for 2-byte signed short integers          (pulseEKKO, RAMAC, SEG-Y)
-2 for 2-byte unsigned short integers        (GSSI)
-5 for 2-byte unsigned short integers, but only first 12-bits used
 3 for 4-byte signed long integers           (SEG-Y)
-3 for 4-byte unsigned long integers
-6 for 4-byte unsigned long integers, but only first 24-bits used
 4 for 4-byte floats                         (SEG-Y)
 8 for 8-byte doubles
```

For example: 8-bit GSSI data are unsigned characters (values from 0 to 255), use -1 for input_datatype. Use –2 for 16-bit GSSI data (values from 0 to 65535). PulseEKKO and RAMAC data are typically 16-bit signed integers (values from -32768 to 32767), use 2 for input_datatype. For SEG-Y data, the input_datatype can be 2 (signed short integers), 3 (signed long integers), or 4 (4-byte floating point reals). Data types are stored in the file header of DZT and SGY files. PulseEKKO and RAMAC do not record the data type.

If the data element type is floating point (input_datatype equal to 4 or 8), then the next 2 parameters must be assigned values in order to control the re-scaling of the data to 8-bit unsigned integers (for the image). For other data types, these 2 values are optional;

KEYWORD: **max_data_val** = 0.0
The maximum value to use for scaling.

KEYWORD: **min_data_val** = 0.0
The minimum value to use for scaling.

KEYWORD: **row_by_row** = "FALSE"
This keyword allows for non-standard data. Set to "TRUE" for data stored "row-by-row" like a screen image. NOTE: samples_per_trace must now be the number of rows in the file. For normal and standard GPR data leave as "FALSE".

******************* SPECIFYING OPTIONAL INPUT FILES **********************

These are additional input ASCII data file names that may or may not be required, depending on other options.

KEYWORD: **mrk_infilename** = ""

KEYWORD: **xyz_infilename** = ""

MRK and XYZ files are used when displaying the traces using spatial coordinates. They contain the number of sets stated on the first file record with the sets listed on following records.

Example MRK file containing marked trace locations:

```
 3
 104
 256
 897
```

Example XYZ file containing X, Y, and Z locations of the marked traces:

```
 3
 10.0  10.0  293.456
 20.0  10.0  294.567
 30.0  10.0  295.678
```

KEYWORD: **lbl_infilename** = ""

LBL files look like C code and allow vector graphics and characters to be overlaid onto the screen image. You can look at example files to see how this works. Here is the list of recognized HPGL commands: viewport, window, pen, ldir, csize, lorg, plot, label, hpgl_select_font, frame, clip, unclip, linetype, circle, arc, wedge, and ellipse. The default color palette is a 256-shade gray scale. A pen color of 0 (black) or 255 (white) is selected based on the background color before the LBL file is read. (The background color is black for screen-only output and white if hardcopy output is requested.) Also the current file viewport() and window() values are set before and after the LBL file is read. Vector graphics only can be supplied in the LBL files and "C" statements such as for, while, if, etc. are not recognized. Vector graphics may be drawn inside or outside of a viewport (if unclip(); is in the file) and within the GPR image.

************************* SPECIFYING HARDCOPY FILES ***********************

To save the screen image to disk, Then place filenames within the parentheses after either, or both, of the next two keywords. If these keywords are left defined as blank strings or the equal sign is missing, then output is to CRT only.

Encapsulate Postscript output should deliver publication quality images. The CRT screen shows what the graphics file image will look like (within the resolution limits of the CRT). If no EPS output file is defined, then the CRT image will have a black background. If a PostScript file is defined, then the background will be white. I have found the most reliable translator of these EPS files is Corel Photo-Paint. Corel Draw, and many others graphics viewers/translators do not read these EPS files correctly. Often some sort of “title” page is shown with no image. Once the EPS file is imported into Photo-Paint with the correct settings (256-gray scale and appropriate resolution), then it can be output in another graphics format (such as BMP or JPEG).

PCX files will look exactly like the screen – gray or color. The number of pixels in the file is the same as the number of screen pixels. If a color palette is selected after the image is displayed (by pressing the 1, 2, or 3 keys), it will be saved with the image.

NOTE: These parameters are used only from the FIRST command file.

KEYWORD: **pcx_outfilename** = ""

KEYWORD: **eps_outfilename** = ""

**************** SELECTING CHANNEL, TRACES, AND SAMPLES ******************
This group determines which channel, traces, and samples to use from a file. If "first_samp" or "last_samp" are not specified or are both 0, then, all trace samples will be used OR they will be determined from the input data/info files. If "lock_first_samp" is "TRUE" then the data/info file cannot override "first_samp" (otherwise, first_samp WILL BE determined from the data/info file if possible).

KEYWORD: **channel** = 0
This is the channel number in multi-channel data sets INDEXED FROM 0; so 0 is first channel, 1 is second channel, etc. This keyword only applies to GSSI DZT files.

KEYWORD: **skip_traces** = 0
If >0, then the number of traces to skip for every one read. For example, if equal to 1, then every other trace is read; if equal to 3, then every fourth trace read. This keyword may have to be assigned for large files (greater than several thousand traces). The CRT screen can display a maximum of 1024 traces.

NOTE: next 3 keywords are active for all coordinate modes specified below.

KEYWORD: **lock_first_samp** = "FALSE"

KEYWORD: **first_samp** = 0
First_samp is interpreted either as "time zero" or "ground surface", depending on the value of coord_mode below.

############ NOTE #############
First_samp specifies the first sample to use from each trace to construct the image of the radar data. It CANNOT be negative. If the data have a start time, or transmit time-zero, before the first sample (as can occur with wide antenna separations) you can use GPR_PROC and its "slide_samp" keyword to move the samples "downward" so that zero time is at the first sample.

Alternatively, the window can be sized (vx1 and vx2) larger than the data range and the left and right axis ranges changed to reflect the actual time zero (laxis_min, laxis_max, raxis_min, raxis_max).

For example, assume the time range is 20 ns and zero time is 5 ns before the first sample is recorded. Set "first_samp" equal to zero (the first sample in the trace) and the user coordinates for the window at "top" equal to -5 and "bottom" equal to 20. The data will be displayed correctly in this window. Now change the endpoints of the left axis to "laxis_min" = 25 and laxis_max = 0 (remember "laxis_tick_int" will be negative). The vertical range of the window is the still the same.
################################

KEYWORD: **last_samp** = 0

*********************** ELIMINATING UNWANTED TRACES ***********************

If there are traces in the data file which you do not want to display, then use num_bad and bad_traces[] to list them.

KEYWORD: **num_bad** = 0

If greater than 0, then the traces listed in bad_traces[] will NOT be displayed.

KEYWORD: **bad_traces[]**

If num_bad is greater than 0, then a set of trace numbers indexed from 0. Add an equal sign after the brackets then list the trace numbers separated by spaces.

*********************** SPECIFYING COORDINATE MODE ************************

This group determines what the coordinate system will be in the data window on the screen and how to place the GPR data into that system. On the CRT screen, a rectangular area called a "viewport" is designated (see below), and the GPR data are displayed within that area. User coordinates are assigned to the viewport (see window coordinates below). GPR data that are within the window coordinates are displayed on the CRT.

KEYWORD: **coord_mode** = 1

This keyword places the data into the viewport using the following coordinate systems.

coord_mode	horizontal axis	vertical axis	description
0	-	-	invalid mode
1	trace number	sample time (ns)	"raw" traces, default
2	distance (m)	distance (m)	geometrically corrected
3	time (sec)	time (ns)	stationary antenna
4	distance (m)	time (ns)	horizontal rubbersheeting, no topographic correction
5	trace number	sample number	"really raw" data, sample rate unknown

These modes are for 2-D displays. Distance units are meters. Time units are nanoseconds (vertically) or seconds (horizontally).

NOTE: first_samp, last_samp, and lock_first_samp are active for all modes.

********************************* MODE 1 **********************************

For coord_mode equal to 1, the next keywords are used
first_trace, last_trace, first_samp_time, last_samp_time.
These parameters should be assigned if defaults are not appropriate.

KEYWORD: **first_trace** = 0

The first trace to use from a file. If 0 then first trace in the file is used

KEYWORD: **last_trace** = 0

The last trace to use from a file. If 0 then last trace in the file is used.

KEYWORD: **first_samp_time** = "INVALID_VALUE"

First sample time {in nanoseconds) to display. If equal to "INVALID_VALUE", then it is determined from first_samp

KEYWORD: **last_samp_time** = "INVALID_VALUE"
Last sample time {in nanoseconds) to display; if "INVALID_VALUE", then it is determined from last_samp.

NOTE: first_samp and last_samp take priority over first_samp_time and last_samp_time.

********************************* MODE 2 **********************************
For coord_mode equal to 2, the next keywords are used:
horiz_mode, horiz_start, horiz_stop, horiz_mode, num_layers,
layer_rdp[], layer_mode[], layer_val[].

These parameters should be assigned if defaults are not appropriate. If horiz_start or horiz_stop are not specified, or are both equal to "INVALID_VALUE", then all traces will be used from a file and these values will be calculated.

NOTE: MRK and XYZ files must be supplied.

KEYWORD: **horiz_mode** = "T"
This determines what coordinates are used for the horizontal direction. Default is "T". Either a numeric value or a string can be assigned to the keyword. Either a numeric value or a string can be assigned to the keyword. Choices are:
"X" or 1 to use X-coordinates
"Y" or 2 to use Y-coordinates
"T" or 3 to use traverse distance coordinates, sqrt(X*X + Y*Y)
NOTE: When traverse distance is selected, "0.0" is at the first tick mark regardless of orientation of the profile with the coordinate axes. To reverse the display, place 0.0 at the right side of the window and the distance on the left side. (See left, right, top, and bottom in data window section below.)

KEYWORD: **horiz_start** = "INVALID_VALUE"
Based on "horiz_mode", it is the location to start getting traces from a file.
NOTE: If horiz_start is set equal to horiz_stop or left undefined, then the data limits are used.

KEYWORD: **horiz_stop** = "INVALID_VALUE"
Based on "horiz_mode", it is the location to stop getting traces from a file.
NOTE: If horiz_start is set equal to horiz_stop or left undefined, then the data limits are used.

KEYWORD: **num_layers** = 0
If greater than 0, the number of layers to use for the time-to-depth conversion

KEYWORD: **layer_rdp[]**
If num_layers is greater than 0, this is the list of relative dielectric permittivities for each layer. Time-to-depth conversion uses the velocity in meters per second.

KEYWORD: **layer_mode[]**
If num_layers is greater than 1, this determines how the BOTTOM of a layer is calculated. Layers can have different modes. This value is assigned 3 if num_layers equal to 1. Either a numeric value or a string can be assigned to the keyword. Choices are:
"D" or 1 if the layer bottom is a constant distance from the surface
"E" or 2 if the layer bottom is at a constant elevation
"I" or 3 if the layer bottom is at infinite depth (lowest or single layer)

KEYWORD: **layer_val[]**
If num_layers is greater than 0, this is the depth or elevation corresponding to the selected layer mode, in user units. This value is assigned "infinity" if num_layers is equal to 1.

******************************** MODE 3 *********************************
For coord_mode equal to 3, the next keywords are used:
trace_per_sec, first_trace_time, last_trace_time, first_samp_time,
last_samp_time.
These parameters should be assigned if defaults are not appropriate.

KEYWORD: **trace_per_sec** = 0.0
This is the number of traces that were recorded per second. If equal to 0.0, then the value will be determined from the data or info file.
NOTE: For DZT files you may have to factor in any stacking that was done at record time. This program does NOT determine the stack from the data.

KEYWORD: **first_trace_time** = "INVALID_VALUE"
Earliest time (in seconds) to display from a file. If equal to "INVALID_VALUE", then first trace is used.

KEYWORD: **last_trace_time** = "INVALID_VALUE"
Latest time (in seconds) to display from a file. If equal to "INVALID_VALUE", then last trace used.

KEYWORD: **first_samp_time** = "INVALID_VALUE"
First trace sample time {in nanoseconds) to display. If equal to "INVALID_VALUE", then determined from first_samp.

KEYWORD: **last_samp_time** = "INVALID_VALUE"
Last trace sample time {in nanoseconds) to display; if "INVALID_VALUE", then determined from last_samp.

NOTE: first_samp and last_samp take priority over first_samp_time and last_samp_time. However, first_trace and last_trace can be determined from first_trace_time and last_trace_time.

******************************** MODE 4 *********************************
For coord_mode equal to 4, the next keywords are used:
horiz_mode, horiz_start, horiz_stop, first_samp_time, last_samp_time.
If horiz_start or horiz_stop are not specified, or are both equal to "INVALID_VALUE", then all traces will be used from a file and these values will be calculated.

NOTE: MRK and XYZ files must be supplied.

KEYWORD: **horiz_mode** = "T"
This determines what coordinates are used for the horizontal direction. Default is "T". Either a numeric value or a string can be assigned to the keyword. Choices are:
"X" or 1 to use X-coordinates
"Y" or 2 to use Y-coordinates
"T" or 3 to use traverse distance coordinates, sqrt(X*X + Y*Y)
NOTE: When traverse distance is selected, "0.0" is at the first tick mark regardless of orientation of the profile with the coordinate axes. To reverse the display, place 0.0 at the right side of the window and the distance on the left side. (See left, right, top, and bottom in data window section below.)

KEYWORD: **horiz_start** = "INVALID_VALUE"
Based on "horiz_mode", it is the location to start getting traces from a file.

KEYWORD: **horiz_stop** = "INVALID_VALUE"
Based on "horiz_mode", it is the location to stop getting traces from a file.

KEYWORD: **first_samp_time** = "INVALID_VALUE"
First trace sample time {in nanoseconds) to display. If equal to "INVALID_VALUE", then it is determined from first_samp

KEYWORD: **last_samp_time** = "INVALID_VALUE"
Last trace sample time {in nanoseconds) to display; if "INVALID_VALUE", then it is determined from last_samp.

NOTE: first_samp and last_samp take priority over first_samp_time and last_samp_time.

********************************* MODE 5 **********************************
For coord_mode equal to 5, the next keywords are used:
  first_trace and last_trace.
These parameters should be assigned if defaults are not appropriate.

KEYWORD: **first_trace** = 0
The first trace to use from a file, if 0 then first trace in the file is used;

KEYWORD: **last_trace** = 0
The last trace to use from a file, if 0 then last trace in the file is used;

************************ CRT GRAPHICS DISPLAY MODE ************************
This parameter controls the graphics display mode.

THIS VALUE SHOULD BE LEFT EQUAL TO 0 UNLESS A PARTICULAR MODE AND RESOLUTION ARE DESIRED OR REQUIRED. For example, the graphics card on some portable computers can be switched to 1024x768 mode for external monitors, but the portable's screen is capable of only 800x600 mode.

The program will automatically select a VESA VGA mode that it recognizes. If no high-resolution mode is found, then the low-resolution mode 20, available for all VGA cards, is selected.
Recognized video modes:

X		Y		colors	IBM	ATI VGA Wonder	VESA SVGA	Tseng	Paradise	HGSC
1024	x	768	x	256	8514		261	56		1024
800	x	600	x	256		99	259	48		800
640	x	480	x	256		98	257	46	95	640
320	x	240	x	256	20	20	20	20	20	20

```
    HGSC = Hercules Graphics Station Card
    Tseng = cards with ET4000 chip set
```

NOTE: This parameter is used only from the FIRST command file.

KEYWORD: **video_mode** = 0
This default value, 0, causes the program to search for highest-resolution mode.

******************** PLACING DATA WINDOW ON THE CRT ***********************
This group determines where the main data window is placed on the monitor screen. The graphics library assigns (0.0,0.0) to the lower left screen corner and (133.333,100.000) to the upper right screen corner. In landscape mode (portrait set equal to 0 below), the range for vx1 and vx2 is 0.0 to 133.333. In portrait mode, range is 0.0 to 75.0, and if vx2 > 75.0 it will be reduced to 75.0. The range for vy1 and vy2 is 0.0 to 100.0 in both portrait and landscape orientations. NOTE: care must be taken in specifying these values for multiple GPR data sets. Results may be unexpected if viewports overlap! On the other hand you may want to overlap windows to superimpose data. Default values are shown. These allow room for the axes labels on the outside of the data window rectangle. These values are honored for all coordinate modes except coord_mode equal to 2. For coord_mode equal to 2, these values specify the maximum size of the data window. One direction, horizontal or vertical, may be shrunk to maintain geometric correctness or vertical exaggeration. The default values for vx1, vx2, vy1, and vy2 allow room outside the window for axis numbers and labels and a title. However, the entire CRT screen can be used.

NOTE: A line is drawn around the viewport.

For EPS (PostScript) files, a margin of approximately one-inch is built in. Everything on the CRT screen will be placed in a rectangle that is at least one inch from the paper edges. This feature is not adjustable at this time.

```
                           CRT SCREEN
  |----------------------------------------------------------|
  | Landscape mode (11 x 8.5)                (133.333,100.0) |
  |                                                          |
  |                                                          |
  |                                                          |
  |             vy2 |-----------------------|                |
  |                 |                       |                |
  |                 |                       |                |
  |                 |                       |                |
  |                 |                       |                |
  |                 |                       |                |
  |                 |                       |                |
  |             vy1 |-----------------------|                |
  |                vx1                     vx2               |
  |                                                          |
  |                                                          |
  |                                                          |
  | (0.0,0.0)                                                |
  |----------------------------------------------------------|
```

KEYWORD: **vx1** = 10.0

KEYWORD: **vx2** = 123.333

KEYWORD: **vy1** = 10.0

KEYWORD: **vy2** = 90.0

KEYWORD: **vert_exag** = 1.0
The vertical exaggeration factor. This only affects the viewport coordinates when coord_mode is equal to 2.

**************** ASSIGNING USER UNITS TO THE DATA WINDOW ******************
This group determines the user-unit limits of the data window. These values must be coordinated with the selected coordinate mode (see coord_mode keyword). If not specified, then limits are determined from "coord_mode" and the data. Even though the selection of samples, traces, and times above must have the first values less than or equal to last values, these 4 parameters specify the coordinate system of the window, and will let you display data "backwards" or "upside-down", or both. The window limits can be a subset, superset, or partial set of the GPR coordinate limits. For example, if coord_mode = 2 and the corrected data will be displayed starting at the 10 m location to the 65 m location and elevation is about 200 m and the corrected data cover about 3 m depth, then the following might be suitable values: left = 10, right = 65, bottom = 197, top = 201. These limits can be larger or smaller than the portion of the data that has been selected to be used by this program. Data will either be clipped at the window edges or surrounded by a blank area.

############ NOTE #############
If a framed viewport appears but GPR data do not appear in it, then one or more of these values is probably incorrect (also check coord_mode).
###############################

KEYWORD: **left** = "INVALID_VALUE"

KEYWORD: **right** = "INVALID_VALUE"

KEYWORD: **bottom** = "INVALID_VALUE"

KEYWORD: **top** = "INVALID_VALUE"

********************** DATA AMPLITUDE PROCESSING ***************************
This group manipulates the data amplitudes directly before they are displayed. The GPR data are converted to unsigned 8-bit bytes first (range 0 to 255). The order shown here is also the order applied in the program. These may take a while as the actual data values are changed. These changes apply to both image and wiggle trace displays.

KEYWORD: **change_gain** = "FALSE"
Change the range gain of the data if "TRUE". The next four keywords are in effect only if this one is "TRUE".

KEYWORD: **num_gain_off** = 0
If greater than or equal to 2, then number of breakpoints for gain to be removed. If equal to 0, then no gain is removal.

KEYWORD: **gain_off[]**
This is the list of floating point values for the gain that will be removed. NOTE: These values are in decibels, dB! For example, to multiply data by 1000, a decibel value of 60 (i.e. 20 * log(1000)) is used. To decrease data amplitudes by 10 (i.e. multiply by 0.10), a decibel value of -20 (i.e. 20 * -1) is used. S&S DT1 files often do not have gain applied. GSSI DZT files usually have gain applied and the values can be known by inspecting the file header with programs such as dzt_rhdr.exe.

For example:
change_gain = "TRUE"
num_gain_off = 3
gain_off[] = 10 20 43

KEYWORD: **num_gain_on** = 0
If greater than or equal to 2, then number of breakpoints for gain to be added. If equal to 0, then no gain is added. This function can be used to multiply all amplitudes by 2, for example, using the following.

```
For example:
change_gain = "TRUE"
num_gain_on = 2
gain_on[]   = 6 6     ; 20 x log(2) = 6
```

KEYWORD: **gain_on[]**
This is the list of floating point values for the gain that will be removed.

KEYWORD: **background** = "FALSE"
If set to TRUE, then a "background" trace is removed from the image. The background trace is the average trace determined by adding all traces together and dividing by the number of traces (stacking). The stacking process enhances coherent signal and reduces randomly varying signal (noise). In this case, the coherent signal is the horizontal banding often seen in GPR data (system noise) and the randomly varying signal is the received radar signal from the subsurface. The appearance of the data is often improved by removing the horizontal banding. Caution must be used, however, with small data sets (less than a few hundred traces) or data that has strong natural horizontal reflectors.

KEYWORD: **abs_val** = "FALSE"
If set to “TRUE” then the amplitude values (at this stage of any other manipulation) will be converted to their absolute value. The mean of the data type (128) is subtracted first and the negative values are converted to positive; then multiplied by 2 to keep them within the gray-scale range of 0 to 255. This option produces results similar to "inst_amp".

KEYWORD: **square** = "FALSE"
If set to “TRUE” then the amplitude values (at this stage of any other manipulation) will be converted to their squared value. The mean of the data type (128) is subtracted first and the values are squared; then divided by 64 to keep them within the gray-scale range of 0 to 255. This option enhances strong features in the data and produces results similar to "inst_pow".

KEYWORD: **inst_amp** = “FALSE”
If set to “TRUE” then the amplitude values (at this stage of any other manipulation) will be converted to instantaneous amplitudes. An analytic function is constructed using the original trace as the real component and its Hilbert transform as the imaginary component. The modulus of the complex function (the square root of the sum of the squares of the real and imaginary components) is called the instantaneous amplitude of the function. For GPR data it measures the reflectivity strength, reducing the appearance of random signal in the data.

KEYWORD: **inst_pow** = "FALSE"
If set to “TRUE” then the amplitude values (at this stage of any other manipulation) will be converted to instantaneous power, or energy. An analytic function is constructed using the original trace as the real component and its Hilbert transform as the imaginary component. The square of the modulus of the complex function (the square root of the sum of the squares of the real and imaginary components) is used. For GPR data it measures the total energy of the GPR signal at an instant in time. The effect on the appearance of the data is similar to converting to instantaneous amplitude, but noise is reduced even further.

############ NOTE #############
Only one, inst_amp or inst_pow, will be used, the first one that is found set to "TRUE".
################################

KEYWORD: **envelope** = “FALSE”
If set to “TRUE” and inst_amp = “TRUE” then an envelope connecting the peaks of the instantaneous amplitude or power will be displayed.

KEYWORD: **stretch** = 0
The stretch technique enhances contrasts in the data. It must be a whole number from 1 to 99. The default value zero indicates no stretching. A histogram is constructed counting the number of times each amplitude (0 to 255) occurs in the entire GPR file. The most commonly found value (the mode) will usually be about 128 (the middle value of the GPR trace). The stretch keyword specifies a value that is a percent of the count found at the mode. Histogram stretching will work properly only if the data are mono-modal with a small standard of deviation (i.e. values are clustered about one central value -- which is of course why we would want to enhance the contrasts). The following diagram (while not technically correct) gives an idea of the effect.

```
                            +   +  +
                        +       *       +
  stretched data -- +          * *           +
        |     +               *   *             +
        | +                  *     *                 +
    +           original -- *       *                     +
+                 data     *         * - stretch /100         +
*    *  *   *   *   *   * *           * *   *   *   *   *  *  *
--------------------------------+-------------------------------
0                              128                           255 (amplitude)
```

************************* LOOK-UP TABLE PROCESSING ************************
This group affects how the GPR data appear in the window. The GPR data are converted to unsigned 8-bit bytes (range 0 to 255) for display. The order shown here is also the order applied in the program. These gray-scale modifiers affect only the look-up table. The look-up table is used for both image and wiggle trace displays. The gray scale palette can be converted to a color palette after the image is displayed by pressing the keys: 1, 2, 3, S, or R. Only a PCX graphics file retains the colors. PostScript files are gray shades only.

KEYWORD: **range** = 255
If greater than 0, then this keywords expands or contracts the gray scale. Set to 0 to use the EPC-style display (see below). If the range is set between 0 and 256, then results are similar to invoking "stretch", that is, contrasts are enhanced. If range is set greater than 255, then contrasts are reduced. For output to most printers, a value of 128 or less is suggested, because most printers are capable of only 120 or so shades of gray.

KEYWORD: **local_offset** = 0.0
If this keyword is greater than 0 then contrasts in the image can be enhanced or reduced. This is the half-width of the local contrast window around the middle value of 127.5. Use 0 for no local stretch. Valid range is 0.0 to 127.5. The endpoint values do not change, that is, 0 is still 0 (black) and 255 is still 255 (white).

KEYWORD: **local_change** = 0.0
If local_stretch is > 0, then this is the amount to add to the high end of the local contrast window and subtract from the low end. Use 0 for no local stretch. A positive value will increase contrasts; a negative value will decrease contrasts. In the example below, contrasts near are enhanced.

```
[new
scale]  contrast window       * -- original table values (*)
255.0       |---|       +  *
|                   +   *
|               +    *   --+
|                 *        |-- local change
127.5         +*         --+
|           *
|        *  +
|     * + -- locally stretched values (+)
 0 *
  0.0 ------ 127.5 --------- 255.0 [original scale]
```

KEYWORD: **brightness** = 0
This keyword will brighten or darken the image by adding this value to each table entry. The valid range is an integer between –128 to 128.

KEYWORD: **negative** = "FALSE"
If this keyword is set to “TRUE” then the gray scale is reversed from black-to-white (0 to 255) to white-to-black (255 to 0).

********************* EPC-STYLE GRAY SCALE DISPLAY ************************
The following group controls the appearance of an EPC graphic recorder style display. In this type of display, the midpoint amplitude of the trace is white; the endpoints of the trace amplitude range (0 and 255) are black, with a gradual darkening in between.

NOTE: The range keyword must be set to 0.

KEYWORD: **epc_threshold** = 5
This is the distance along the scale from the midpoint to start darkening; that is the band of white at midpoint is wider the larger epc_threshold is. Range is 0 to 127.

KEYWORD: **epc_contrast** = 255
This is the value along the scale for the "black" endpoint of the gray scale. Range is 1 to 255.

KEYWORD: **epc_sign** = 0
If greater than 0, then all values below midpoint are white. If less than 0, then all values above midpoint are white.

KEYWORD: **epc_gain** = 2.0
This keyword multiplies contrast; that is "black" occurs closer to midpoint.

******************** ADDING MARKED TRACE LOCATIONS ************************
This group only affects the appearance of the gray-scale image.

KEYWORD: **show_markers** = 0
If this keyword is not equal to 0, then the location of marked traces are shown in the image. Marked traces are determined from a MRK file. The length of the mark is the absolute value of this keyword in screen

pixels (NOT user units. If greater than 0, then marks are black. If less than 0, marks are white. This option not available if coord_mode is equal to 2.

KEYWORD: **show_marker_width** = 1
If show_markers is not equal to 0, then this is the width of the marker line in screen pixels.

******************** WIGGLE TRACE DISPLAY OF DATA ***********************
This group effects how the data appear in the window. The GPR data are displayed as wiggle traces if display_wiggle is set to "TRUE". Selected traces may also be superimposed on a gray-scale image of the GPR data (display_wiggle is "FALSE") if num_wiggles is a value greater than 0 and wiggle_traceval[] is assigned values.

IMPORTANT! If display_wiggle is set to "true": The quality of this display mode depends heavily on how many traces are displayed in the viewport/window and the width of the trace. It is suggested that only 1 to 2 traces be displayed for each viewport unit (see vx1 and vx2 commands above). Use skip_traces to adjust the number of traces displayed. Use wiggle_width to adjust trace width (distance between peaks) – I suggest starting with 10. The info screen displayed at the start of GPR_DISP shows the total number traces in the file and the number to be displayed (as num_cols).

KEYWORD: **display_wiggle** = "FALSE"
If "TRUE" then the data are displayed as wiggle traces. If "FALSE", the data are displayed as an image.

KEYWORD: **num_wiggles** = 0
If this keyword is greater than 0 and display_wiggle is "FALSE", then this is the number traces to display as wiggles superimposed on the gray-scale image. Traces are listed after the next keyword.

KEYWORD: **wiggle_traceval[]**
If num_wiggles is greater than 0 and display_wiggle is "FALSE", then this keyword defines the locations of traces to be superimposed on the image as wiggles. NOTE: The values depend on the coord_mode selected. For coord_mode equal to 1 or 5, then the trace number is used. For coord_mode equal to 2 or 4, the distance is used. For coord_mode equal to 3, then the trace time is used.

KEYWORD: **wiggle_color** = -1
This is the gray level to assign to the wiggle traces from 0 (black) to 255 (white). The default is -1, and the shade will be selected based on whether hardcopy output is requested or not. Hardcopy output has a white background so wiggles will be black. If wiggle color is not appearing as desired (sometimes it is white on a white background for hardcopy output), set wiggle_color to the desired value.

KEYWORD: **wiggle_fill** = 0
If this keyword is less than 0, then the negative values of the wiggle traces will be shaded. If greater than 0, then the positive sides are filled. If set to 0 (default), neither side is shaded.

KEYWORD: **wiggle_width** = 10.0
This is the width of the wiggle trace as a percentage of the viewport/window width. Range is 0.001 to 100.0.

KEYWORD: **wiggle_clip** = 100.0
This is the clip limit as a percentage of wiggle_width. There is no clipping if equal to 100.0. The clip will be at half of width if set to 50. Range is 0.001 to 100.0.

************** OVERLAYING A HYPERBOLA FOR VELOCITY ANALYSIS **************

Because the GPR antenna radiation pattern extends in front of and behind the antennas (E-field oriented perpendicular to the tow direction), objects are detected before and after the antennas pass over them. The reflection pattern is in the shape of half of a hyperbola. These keywords overlay a reflection hyperbola on the image and a cross noting where the object would be in distance-time space.

KEYWORD: **plot_hyperbola** = "FALSE"
Set this keyword to "TRUE" to overlay the velocity hyperbola on the image.

KEYWORD: **ant_sep** = 0
This is the distance between the centers of the two antennas in meters. Value must be greater than or equal to zero

KEYWORD: **obj_depth** = 0
This is the depth in meters to the center of the object. The object is assumed to have a circular cross-section in the direction of the antenna tow direction.

KEYWORD: **obj_radius** = 0
This is the radius in meters of the circular object. The length of the object is assumed large (such as a long rod), and perpendicular to the antenna tow direction, to maximize reflection of the radar waves, which have their electric field polarized also perpendicular to the tow direction.

KEYWORD: **obj_loc** = 0
This is the location of the center object in user coordinates in whatever horizontal coordinates are used to display the image. For example, if the image has horizontal values of 0 m and 20 m at the left and right edges respectively, then the obj_loc should be in the range of 0 to 20 m. However, the object can lie outside this range. The object is assumed to be in the plane of the image. That is it cannot be off the path that the antenna was towed on.

KEYWORD: **hyp_vel** = 0
This is the average velocity of the radar waves in the medium above the object, in meters per nanosecond. Values must be in the range 0.01 to 0.30 m/ns. If the value is outside of this range, no hyperbola will be plotted.

KEYWORD: **hyp_color** = 0
This is the pen color to use to plot the hyperbola and the cross representing the object's center. The valid range is 0 (black) to 255 (white).

***************** MODIFYING AND CONTROLLING EPS OUTPUT *******************

A filename must be defined for the “eps_outfilename” keyword for keywords in this section to have any effect. These keywords do not affect the image itself. These are options to orient the image with respect to the paper (see discussion of vx1 and vx2 above for portrait mode) and to add page numbers and figure captions to the EPS image.

NOTE: “portrait and “use_11x17” are used only from the FIRST command file.

KEYWORD: **portrait** = "FALSE"
Set this keyword to "TRUE" to get Encapsulated PostScript output in portrait orientation. In portrait mode, the bottom of the page is the short edge. The default is landscape orientation (bottom of page is long edge).

KEYWORD: **use_11x17** = "FALSE"
Use 11 x 17-inch paper if "TRUE". 8.5 x 11 in. paper is the default.

The next keywords allow the user to include a "figure caption" on the printed EPS page. The caption can appear on either the short or long edge of the paper (regardless of whether the image is being printed in portrait or landscape mode). Available fonts depend on the PostScript printer being used. The standard character sets found on typical PostScript printers include: Times-Roman, Times-Italic, Times-Bold, Times-BoldItalic, Helvetica, Helvetica-Oblique, Helvetica-Bold, Helvetica-BoldOblique, Courier, Courier-Oblique, Courier-Bold, and Courier-BoldOblique.

NOTE: You specify the number of inches from the left and bottom sides of paper for the LAST line of the caption; hold the paper in front of you oriented so that the edge the caption is to be printed on is at the bottom. There is no check to see if a printed line is going to run off the page to the "right", so you must be sure to limit the length of lines appropriately. Also, proportional fonts may cause unexpected effects as far as lining up text. Here is an example from Steve Duke's MS Thesis (Colo. School of Mines):

```
caption = "Figure 3-6  900 MHz pulse signatures from zero to one"
          "            wavelength separation in air at normal incidence."
```

A maximum of 2000 characters may given, however, for now the image may cover upper lines if more than 2 lines are specified.

NOTE: These values are only used from the FIRST command file. These characters appear ONLY in the output file and NOT on the CRT.

KEYWORD: **eps_select_font** = "Times-Roman"

KEYWORD: **caption_font_size** = 12
This is the point size (72 points per inch).

KEYWORD: **caption_left_edge** = 1.25
Inches from "left" side of paper.

KEYWORD: **caption_bottom_edge** = 1.00
Inches from "bottom" side of paper.

KEYWORD: **caption_long_edge** = "FALSE"
Set to "TRUE" to print on long edge, otherwise caption is on short edge.

KEYWORD: **caption** = ""
"" indicates no caption.

The next parameters allow the user to include a page number on the figure. The page number is always printed with the page in portrait mode. You specify the distance in inches from the left and bottom edges of the page to the lower left side of the page number. The page number must be a string to allow for such cases as 'ix', 'V', 'A1', 'B-2', etc. A maximum of 200 characters may given. 'pagenum' can also be used to add running headers or footers to the figure.

NOTE: These values are only used from the FIRST command file. These characters appear only in the output file and NOT on the CRT.

KEYWORD: **pagenum_font_size** = 12
This is the point size (72 points per inch)

KEYWORD: **pagenum_left_edge** = 7.25
Inches from "left" side of paper

KEYWORD: **pagenum_bottom_edge** = 10.00
Inches from "bottom" side of paper

KEYWORD: **pagenum** = ""
"" indicates no page number

******************* SELECTING FONTS FOR CRT AND EPS ***********************
This keyword determines the HERSHEY font for graphics text (axes labels, etc.) Possible fonts are: cyrillic, gothengl, gothgerm, gothital, grkcmplx, grkcmpsm, grksmplx, itlcmplx, itlcmpsm, itltrplx, romcmplx, romcmpsm, romduplx, romsmplx, romtrplx, scrcmplx, scrsmplx, eqncmplx, eqntrplx, and eqnsmplx.

NOTE: There must be a root directory on your computer called HERSHEY. Available fonts (included with this software) are found there. If the Hershey fonts are missing, a default (and crude) HPGL font is used.

KEYWORD: **hpgl_select_font** = "romtrplx"

****************** OVERLAYING GRID LINES ON THE DATA *******************
This group controls the overlaying of horizontal or vertical lines on the data within the window. Start and stop will default to limits of the window. Lines will ONLY be drawn if add_hlines and/or add_vlines are set to TRUE and ...int is assigned a value. Lines will not be drawn outside of the data window. Defaults are to not draw lines. A line style may also be selected (styles are defined in hpgl.h). A number from 0 to 7 can be assigned (0 is default). The style represents how "bits" are turned on along a line when drawn.

```
  0 = 0xFFFFFFFF (solid)
  1 = 0x88888888 (dotted)
  2 = 0xF0F0F0F0 (short dashes)
  3 = 0xFFF0FFF0 (long dashes)
  4 = 0x7FF27FF2 (long dash, dot)
  5 = 0xFFF6FFF6 (long dash, short dash)
  6 = 0x7F367F36 (long dash, 2 short dashes)
  7 = 0xF666F666 (short dash, 3 short dashes)
```

KEYWORD: **add_hlines** = "FALSE"

KEYWORD: **hline_start** = “INVALID_VALUE”
Start horizontal lines at this "vertical" value, in user units.

KEYWORD: **hline_int** = “INVALID_VALUE”
This is the interval, in user units, between horizontal lines.

KEYWORD: **hline_stop** = “INVALID_VALUE”
Stop horizontal lines at this "vertical" value, in user units.

KEYWORD: **hline_style** = 0
This is the horizontal line style.

KEYWORD: **add_vlines** = "FALSE"

KEYWORD: **vline_start** = “INVALID_VALUE”
Start vertical lines at this "horizontal" value, in user units.

KEYWORD: **vline_int** = “INVALID_VALUE”
This is the interval, in user units, between vertical lines.

KEYWORD: **vline_stop** = “INVALID_VALUE”
Stop vertical lines at this "horizontal" value, in user units.

KEYWORD: **vline_style** = 0

******************************** TITLE ***********************************
The title appears above the top edge of the image. This may also be specified in a LBL file and placed appropriately.

taxis_..." parameters below can control size and placement if defined here.

KEYWORD: **title** = ""

************** CONTROLLING AXIS TICKS, ANNOTATION AND LABELS **************
This group controls the annotation of axes around the data window. These keywords cause the most problems and confusion when using GPR_DISP. Incorrect selections will leave the window without axes ticks and labels or produce unexpected results. In versions older than 2.0, the program is will actually quit working or "lock" the operating system. I have built in checks to prevent this. If you cannot produce the axes you want, please look at the example data and command files included with this report.

..._min and ..._max of axes will default to appropriate left, right, bottom, top values defined above. Annotation of axes can be forced to be different than the data window values by assigning values to ..._min and ..._max.

Only tick marks within the limits of the data window will be displayed, so ...tick_start and ...tick_int can be outside of expected data window limits. If ...tick_start and ...tick_int are not given then GPR_DISP can determine where axis tick marks should go.

...ano_left and ...ano_right default to the significant digits in the tick mark annotation but can be fixed by specifying values for them.

Values for ...title_size and ...ano_size below are actually multipliers that may vary depending on the size of the data window. Experiment to find the best size.

Axis titles are drawn only if tick marks are displayed (that is ...tick_ano must be set to “TRUE”).

There can be 3 different tick lengths. Labeled ticks (determined from ..._tick_start and ..._tick_skip) are the longest. Second order ticks (determined from ..._tick_start and ..._tick_mid_skip) are the second longest and not annotated. The rest (determined from ..._tick_start and ..._tick_int) are the shortest and not annotated.

############ NOTE #############
If you just set ...tick_show to 1 or –1 and ..._tick_ano to "TRUE" the program will label the axis using default values.
###############################

LEFT (vertical) axis display/annotation controls.

KEYWORD: **laxis_title** = ""
If characters are inserted between the quote marks then this title will be displayed to the left of the window and rotated counterclockwise 90 degrees.

KEYWORD: **laxis_title_offset** = 0.0
This is the number of character widths to move the title away from (greater than 0) or toward (less than 0) the left window edge from default location. This value can often be left at 0.0.

KEYWORD: **laxis_title_size** = 0.85
This is the relative size of the characters in the title.

KEYWORD: **laxis_max** = "INVALID_VALUE"
This is the value at the TOP of the left vertical axis. This value will default to the value of the "top" keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **laxis_min** = "INVALID_VALUE"
This is the value at the BOTTOM of the left vertical axis. This value will default to the value of the "bottom" keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **laxis_tick_show** = 0
This keyword determines of tick marks are drawn along the left edge of the data window. Use 0 to not draw tick marks. Use 1 to draw outside of window. Use -1 to draw on inside of window. Only ticks in the range laxis_tick_abs_min to laxis_tick_abs_max will be shown.

KEYWORD: **laxis_tick_start** = "INVALID_VALUE"
Start adding ticks to the axis at this value (towards or below bottom of axis). If left unassigned, it will default to the value of laxis_min.

KEYWORD: **laxis_tick_int** = "INVALID_VALUE"
This is the interval between (shortest) tick marks. If the left axis is, for example, travel time in ns and the data window is 20 ns wide, then the following keyword values will place a short tick every 1 ns along the axis (20 at the bottom and 0 at the top): laxis_tick_start = 20 and laxis_tick_int = -1. If left unassigned, the default value will be calculated.

KEYWORD: **laxis_tick_abs_min** = "INVALID_VALUE"
This keyword prevents any tick marks from being displayed below this value. It defaults to the value of laxis_min.

KEYWORD: **laxis_tick_abs_max** = "INVALID_VALUE"
This keyword prevents any ticks from being displayed above this value. It defaults to the value of laxis_max.

KEYWORD: **laxis_tick_num** = 9
If laxis_tick_start and laxis_tick_int are not set to valid values, then the program will try to label the axis with this many tick marks by default. This keyword is only active if laxis_tick_show is not equal to 0.

KEYWORD: **laxis_tick_ano** = "FALSE"
This keyword will label the tick marks with numbers if set to "TRUE". This also lengthens the labeled tick marks to their longest setting. Tick marks can be shown but left unlabeled.

KEYWORD: **laxis_tick_skip** = 0
This is the number of (shortest) tick marks to skip between annotated ones. For example, if this keyword is set to 1 then every other tick mark is labeled. Some annotations may be skipped if the numbers are too close together.

KEYWORD: **laxis_tick_mid_skip** = -1
This is the number of (shortest) tick marks to skip between the 2nd-order (medium-length) tick marks. These tick marks are not annotated. If set to -1, then no 2nd order ticks.

KEYWORD: **laxis_ano_left** = 0
This keyword selects the number of places (up to 6) to left of the decimal point for floating point numbers. If left at the default value of 0, then the numbers will displayed in the simplest fashion.

KEYWORD: **laxis_ano_right** = -1
This keyword selects the number of places (up to 6) to right of the decimal point for floating point numbers. If left at the default value of -1, then the numbers will displayed in the simplest fashion. A value of 0 forces the truncation of real numbers to integer numbers.

KEYWORD: **laxis_ano_size** = 0.80
This is the relative size of the numbers annotating the tick marks.

--
RIGHT axis display/annotation controls

KEYWORD: **raxis_title** = ""
If characters are inserted between the quote marks then this title will be displayed to the right of the window and rotated counterclockwise 90 degrees.

KEYWORD: **raxis_title_offset** = 0.0
This is the number of character widths to move the title away from (greater than 0) or toward (less than 0) the right window edge from default location. This value can often be left at 0.0.

KEYWORD: **raxis_title_size** = 0.85
This is the relative size of the characters in the title.

KEYWORD: **raxis_max** = "INVALID_VALUE"
This is the value at the TOP of the right vertical axis. This value will default to the value of the "top" keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **raxis_min** = “INVALID_VALUE”
This is the value at the BOTTOM of the right vertical axis. This value will default to the value of the “bottom” keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **raxis_tick_show** = 0
This keyword determines of tick marks are drawn along the right edge of the data window. Use 0 to not draw tick marks. Use 1 to draw outside of window. Use -1 to draw on inside of window. Only ticks in the range raxis_tick_abs_min to raxis_tick_abs_max will be shown.

KEYWORD: **raxis_tick_start** = “INVALID_VALUE”r
Start adding ticks to the axis at this value (towards or below bottom of axis). If left unassigned, it will default to the value of raxis_min.

KEYWORD: **raxis_tick_int** = “INVALID_VALUE”
This is the interval between (shortest) tick marks. If the right axis is, for example, travel time in ns and the data window is 20 ns wide, then the following keyword values will place a short tick every 1 ns along the axis (20 at the bottom and 0 at the top): raxis_tick_start = 20 and raxis_tick_int = -1. If left unassigned, the default value will be calculated.

KEYWORD: **raxis_tick_abs_min** = “INVALID_VALUE”
This keyword prevents any tick marks from being displayed below this value. It defaults to the value of raxis_min.

KEYWORD: **raxis_tick_abs_max** = “INVALID_VALUE”
This keyword prevents any ticks from being displayed above this value. It defaults to the value of raxis_max.

KEYWORD: **raxis_tick_num** = 9
If raxis_tick_start and raxis_tick_int are not set to valid values, then the program will try to label the axis with this many tick marks by default. This keyword is only active if raxis_tick_show is not equal to 0.

KEYWORD: **raxis_tick_ano** = "FALSE"
This keyword will label the tick marks with numbers if set to “TRUE”. This also lengthens the labeled tick marks to their longest setting. Tick marks can be shown but left unlabeled.

KEYWORD: **raxis_tick_skip** = 0
This is the number of (shortest) tick marks to skip between annotated ones. For example, if this keyword is set to 1 then every other tick mark is labeled. Some annotations may be skipped if the numbers are too close together.

KEYWORD: **raxis_tick_mid_skip** = -1
This is the number of (shortest) tick marks to skip between the 2nd-order (medium-length) tick marks. These tick marks are not annotated. If set to -1, then no 2nd order ticks.

KEYWORD: **raxis_ano_left** = 0
This keyword selects the number of places (up to 6) to left of the decimal point for floating point numbers. If left at the default value of 0, then the numbers will displayed in the simplest fashion.

KEYWORD: **raxis_ano_right** = -1
This keyword selects the number of places (up to 6) to right of the decimal point for floating point numbers. If left at the default value of -1, then the numbers will displayed in the simplest fashion. A value of 0 forces the truncation of real numbers to integer numbers.

KEYWORD: **raxis_ano_size** = 0.80
This is the relative size of the numbers annotating the tick marks.

--
TOP axis display/annotation controls

KEYWORD: **taxis_title** = ""
If characters are inserted between the quote marks then this title will be displayed above the window.

KEYWORD: **taxis_title_offset** = 0.0
This is the number of character widths to move the title away from (greater than 0) or toward (less than 0) the top window edge from default location. This value can often be left at 0.0.

KEYWORD: **taxis_title_size** = 0.85
This is the relative size of the characters in the title.

KEYWORD: **taxis_max** = “INVALID_VALUE”
This is the value at the RIGHT of the top horizontal axis. This value will default to the value of the “right” keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **taxis_min** = “INVALID_VALUE”
This is the value at the LEFT of the top horizontal axis. This value will default to the value of the “left” keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **taxis_tick_show** = 0
This keyword determines of tick marks are drawn along the top edge of the data window. Use 0 to not draw tick marks. Use 1 to draw outside of window. Use -1 to draw on inside of window. Only ticks in the range taxis_tick_abs_min to taxis_tick_abs_max will be shown.

KEYWORD: **taxis_tick_start** = “INVALID_VALUE”r
Start adding ticks to the axis at this value (towards or to the left of the top axis). If left unassigned, it will default to the value of taxis_min.

KEYWORD: **taxis_tick_int** = “INVALID_VALUE”
This is the interval between (shortest) tick marks. If the top axis is, for example, trace number there are 151 traces in the data window, then the following keyword values will place a short tick every 10 ns along the axis (0 at the left and 150 at the right): taxis_tick_start = 0 and taxis_tick_int = 10. If left unassigned, the default value will be calculated.

KEYWORD: **taxis_tick_abs_min** = “INVALID_VALUE”
This keyword prevents any tick marks from being displayed to the left of this value. It defaults to the value of taxis_min.

KEYWORD: **taxis_tick_abs_max** = “INVALID_VALUE”
This keyword prevents any ticks from being displayed to the right of this value. It defaults to the value of taxis_max.

KEYWORD: **taxis_tick_num** = 9
If taxis_tick_start and taxis_tick_int are not set to valid values, then the program will try to label the axis with this many tick marks by default. This keyword is only active if taxis_tick_show is not equal to 0.

KEYWORD: **taxis_tick_ano** = "FALSE"
This keyword will label the tick marks with numbers if set to “TRUE”. This also lengthens the labeled tick marks to their longest setting. Tick marks can be shown but left unlabeled.

KEYWORD: **taxis_tick_skip** = 0
This is the number of (shortest) tick marks to skip between annotated ones. For example, if this keyword is set to 1 then every other tick mark is labeled. Some annotations may be skipped if the numbers are too close together.

KEYWORD: **taxis_tick_mid_skip** = -1
This is the number of (shortest) tick marks to skip between the 2nd-order (medium-length) tick marks. These tick marks are not annotated. If set to -1, then no 2nd order ticks.

KEYWORD: **taxis_ano_left** = 0
This keyword selects the number of places (up to 6) to left of the decimal point for floating point numbers. If left at the default value of 0, then the numbers will displayed in the simplest fashion.

KEYWORD: **taxis_ano_right** = -1
This keyword selects the number of places (up to 6) to right of the decimal point for floating point numbers. If left at the default value of -1, then the numbers will displayed in the simplest fashion. A value of 0 forces the truncation of real numbers to integer numbers.

KEYWORD: **taxis_ano_size** = 0.80
This is the relative size of the numbers annotating the tick marks.

--
BOTTOM axis display/annotation controls

KEYWORD: **baxis_title** = ""
If characters are inserted between the quote marks then this title will be displayed below the window.

KEYWORD: **baxis_title_offset** = 0.0
This is the number of character widths to move the title away from (greater than 0) or toward (less than 0) the bottom window edge from default location. This value can often be left at 0.0.

KEYWORD: **baxis_title_size** = 0.85
This is the relative size of the characters in the title.

KEYWORD: **baxis_max** = “INVALID_VALUE”
This is the value at the RIGHT of the bottom horizontal axis. This value will default to the value of the “right” keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **baxis_min** = “INVALID_VALUE”
This is the value at the LEFT of the bottom horizontal axis. This value will default to the value of the “left” keyword. You can force this value to be different from, and possibly unrelated to, the actual data window limits. This keyword can usually be left unassigned.

KEYWORD: **baxis_tick_show** = 0
This keyword determines of tick marks are drawn along the bottom edge of the data window. Use 0 to not draw tick marks. Use 1 to draw outside of window. Use -1 to draw on inside of window. Only ticks in the range baxis_tick_abs_min to baxis_tick_abs_max will be shown.

KEYWORD: **baxis_tick_start** = “INVALID_VALUE”r
Start adding ticks to the axis at this value (towards or to the left of the bottom axis). If left unassigned, it will default to the value of baxis_min.

KEYWORD: **baxis_tick_int** = “INVALID_VALUE”
This is the interval between (shortest) tick marks. If the bottom axis is, for example, trace number there are 151 traces in the data window, then the following keyword values will place a short tick every 10 ns along the axis (0 at the left and 150 at the right): baxis_tick_start = 0 and baxis_tick_int = 10. If left unassigned, the default value will be calculated.

KEYWORD: **baxis_tick_abs_min** = “INVALID_VALUE”
This keyword prevents any tick marks from being displayed to the left of this value. It defaults to the value of baxis_min.

KEYWORD: **baxis_tick_abs_max** = “INVALID_VALUE”
This keyword prevents any ticks from being displayed to the right of this value. It defaults to the value of baxis_max.

KEYWORD: **baxis_tick_num** = 9
If baxis_tick_start and baxis_tick_int are not set to valid values, then the program will try to label the axis with this many tick marks by default. This keyword is only active if baxis_tick_show is not equal to 0.

KEYWORD: **baxis_tick_ano** = "FALSE"
This keyword will label the tick marks with numbers if set to “TRUE”. This also lengthens the labeled tick marks to their longest setting. Tick marks can be shown but left unlabeled.

KEYWORD: **baxis_tick_skip** = 0
This is the number of (shortest) tick marks to skip between annotated ones. For example, if this keyword is set to 1 then every other tick mark is labeled. Some annotations may be skipped if the numbers are too close together.

KEYWORD: **baxis_tick_mid_skip** = -1
This is the number of (shortest) tick marks to skip between the 2nd-order (medium-length) tick marks. These tick marks are not annotated. If set to -1, then no 2nd order ticks.

KEYWORD: **baxis_ano_left** = 0
This keyword selects the number of places (up to 6) to left of the decimal point for floating point numbers. If left at the default value of 0, then the numbers will displayed in the simplest fashion.

KEYWORD: **baxis_ano_right** = -1
This keyword selects the number of places (up to 6) to right of the decimal point for floating point numbers. If left at the default value of -1, then the numbers will displayed in the simplest fashion. A value of 0 forces the truncation of real numbers to integer numbers.

KEYWORD: **baxis_ano_size** = 0.80
This is the relative size of the numbers annotating the tick marks.

Usage: `GPR_DISP cmd_filename [override_cmd_filename]`
Required command line arguments:
cmd_filename - The name of the keyword file.
Optional command line arguments (do not include brackets):
override_cmd_filename - The name of a keyword file that can override some keywords in the main keyword file.
Examples:

```
gpr_disp dfile1.cmd
gpr_disp dfile1.cmd doption1.cmd
```