Subject: PV-WAVE FAQ (2 of 2)
Posted by mgs on Mon, 26 Feb 1996 08:00:00 GMT
View Forum Message <> Reply to Message

Following is part 2 of the updated version of the PV-WAVE FAQ. Please send info about corrections, requests, etc., to mgs@visdata.com.

TECHNICAL QUESTIONS:

T01. Why doesn't polycontour fill open contours??

In PV-WAVE 6.0 there is a new procedure called CONTOURFILL, which does a similar job to POLYCONTOUR, except it does fill open contours. It also uses a better polygon sorting algorithm (POLYCONTOUR does not fill some contours correctly. For example, an elevation map of a volcano would not have the internal crater colored properly, but CONTOURFILL will handle it properly). Also, there is a new user library routine called FILL_CONTOUR, which does the work of calling CONTOUR and CONTOURFILL for you.

If you do not have access to PV-WAVE 6.0, the following information is applicable.

This problem is described in the POLYCONTOUR manual page.

RESTRICTIONS:

This routine will NOT draw open contours. To eliminate open contours in your dataset, surround the original array with a 1-element border on all sides. The border should be set to a value less than or equal to the minimum data array value.

For example, if A is an (N,M) array enter:

```
B = REPLICATE(MIN(A), N+2, M+2)  ;Make background
B(1,1) = A                       ;Insert original data
CONTOUR, B, PATH=Filename ...    ;Create the contour file.
```

The following is from Ray Sterner at Johns Hopkins University:
Here is a very simple algorithm that might be a useful addition to the section of the FAQ on filled contours. It is for evenly spaced contours only.

Z is an array to be contoured,
CI is the desired contour interval,

C0 is the desired starting color index,
D is the desired step between colors.

T = fix(Z/CI)
M = T - smooth(T,3)
F = (C0 + T*D)*(1-M)

is an array with filled contours with the contours plotted with color 0. For contours of a different color simply add M*CC where CC is the desired contour color index.

T05. Is there on-line help for PV-WAVE?

WAVE> HELP [,'command']
Now brings up On-Line HELP on all platforms. This will be Windows help on Windows platfoms, and Bristol HyperHelp (with a similar look and feel) on others.

WAVE> HELP, /Documenatation
Brings up the (FrameViewer) on-line docuemntation (formerly started by just HELP). This is only available on selected platforms.

(NOTE - The selected platfoms are basically non-Digital platforms, FrameViewer is available on all Unix's except Digital Unix (Alpha) and will be available on Windows)

T06. I run PV-WAVE under X in SunOS 4.x, and after I logout, the screen becomes completely blank. Typing in login names and passwords blindly logs you in again with the correct colors. How to prevent this?

Put a call to clear_colormap in your .login file to be executed after OpenWindows start up.

T07. Sometimes my variables seem to disappear. Why is this?

Quoting the PV-WAVE User's Guide, page 10-8:

PV-WAVE users may find that all their variables have seemingly disappeared after an error occurs inside a procedure or function. The misunderstood subtlety is that after the error occurs, PV-WAVE's context is inside the called procedure, not in the main level. Typing RETALL or RETURN will make the lost variables reappear.

RETALL is best suited for use when an error is detected in a procedure and it is desired to return immediately to the main program level despite nested procedure calls. RETALL issues RETURN commands until the main program level is reached.

The HELP command can be used to see the current call stack (i.e., which program unit PV-WAVE is in and which program unit called it).

T08. Is there a major mode for editing PV-WAVE code in Emacs?

There is a PV-WAVE major mode for emacs that has many of the features that one would like in such a mode. It is available with the PV-WAVE distribution, and may be found in the $WAVE_DIR/lib/emacs directory. Alternately, it can also be obtained from the following URLs:

ftp://ftp.boulder.vni.com/PVI/emacs/wave-mode.shar
ftp://eos.crseo.ucsb.edu/pub/idl/wave-mode.shar

Chris Chase (chase@jackson.jhuapl.edu) has written idl.el for editing IDL code. He has also written idl-shell.el for running IDL as an inferior process under emacs. URLs for these files are:

ftp://eos.crseo.ucsb.edu/pub/idl/idl.el
ftp://eos.crseo.ucsb.edu/pub/idl/idl-shell.el
ftp://fermi.jhuapl.edu/pub/idl_emacs/idl.el
ftp://fermi.jhuapl.edu/pub/idl_emacs/idl-shell.el

Given the (remaining) strong similarity between IDL and PV-WAVE these ought to be useable.

See Appendix A02 for details on using URLs.

T11. Where are all the PV-WAVE routines and userlib procedures?

The "Kernal" C routines are not accessible, for proprietary reasons. The userlib, standard lib and widget procedures are in $WAVE_DIR/lib/ andd are disstributed on the PV-WAVE CDROM.The system variable !path also contains the directory names for all accessible PV-WAVE procedures. The most current entries to the User Lib are available through VNI's WWW homepage and their FTP site.

T12. Does anybody know how to put multiple image plots on one page in PostScript?

Because PostScript has scalable pixels, you must specify the xsize and ysize parameters, as well as the position parameter, in TV or TVSCL.

```
  ; Display four images in a 2x2 grid ; Assume

       data(x,y,4) = array containing the 4 images

set_plot, 'ps'          ;request PostScript output device, ...

;modify page size, orientation, etc. as desired ximsize =

       0.5*!d.x_size  ;define output image size yimsize = 0.5*!d.y_size

       ;note: 0.5 assumes 2x2 grid for i=0,3 do begin       ;display the

       4images, using i as position index tv, data(*,*,i), i,

       xsize=ximsize, ysize=yimsize endfor
```

T13. Does case matter in PV-WAVE?

No.

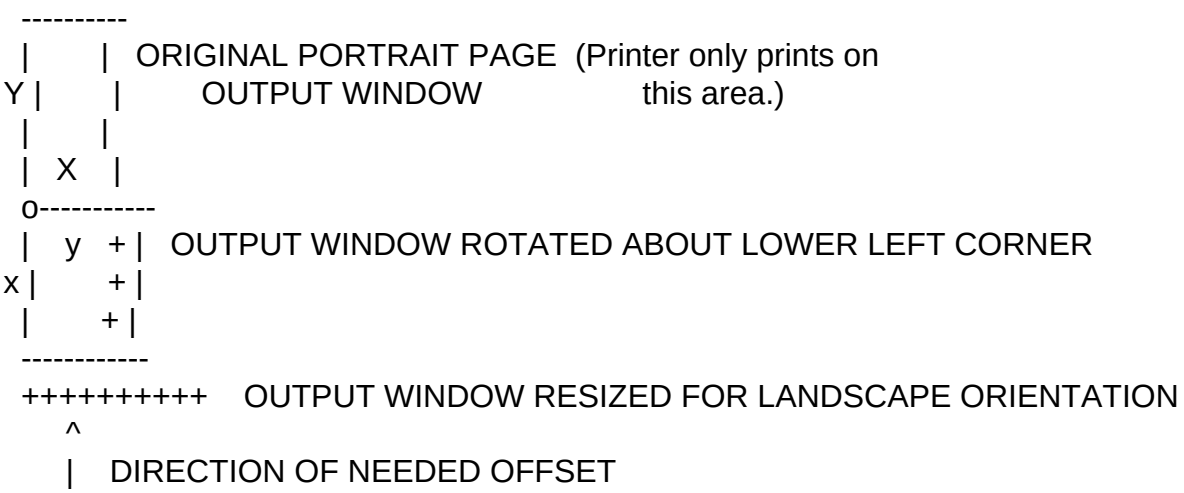
Compiled routines are case insensitive. The only catch is that, on Unix systems, when executing a script via the .RUN command, the file name argument must exactly match the file name as it appears on the disk. Once the routines in the script are compiled, their names can be written in any case. This is not a problem in operating systems such as VMS that do not distinguish case for file names. This is usually not a problem under Unix either since, by convention, most people use lower case file names.

T14. How do I set up PV-WAVE to get precise control over plot window and text positioning with either portrait or landscape page orientation on a PostScript or HP-GL printer?

(This answer only applies to PostScript and HP-GL printers -- other printers may differ in having the X and Y offsets measured from the upper left corner of the portrait page instead of the lower left corner.)

PV-WAVE uses portrait page orientation as a default. (The x axis is along the shorter dimension of the paper.) In portrait orientation the lower left corner of the page is the origin for the XOFFSET and YOFFSET page offsetting keywords of the DEVICE command that determine the origin (lower left corner) of the output window. (Normally one uses XOFFSET=0 and YOFFSET=0 for

portrait orientation.) Size of the output window is determined by the XSIZE and YSIZE keywords of the DEVICE command. The origin for graph positioning variables !P.POSITION and !P.REGION is the output window origin. X and Y coordinates for portrait page orientation are shown on the sketch below as upper case X and Y.



If device,/landscape is specified, then the output window is rotated 90 deg. clockwise about the lower left corner of the page. In this condition nothing will be plotted on the page, since the rotation has carried the output window entirely off the paper as shown in the sketch above. To correct this mismatch, the rotated output window must be offset. XOFFSET AND YOFFSET are still measured in the X and Y coordinates of the portrait page, but now represent the position of the lower left corner of the rotated (and resized) output window (marked by an o above) with respect to the lower left corner of the portrait (actual) page. Hence, one generally uses XOFFSET=0 and YOFFSET=long_dimension_of_page for landscape orientation.

In landscape orientation, the coordinates for graph positioning variables !P.POSITION and !P.REGION are the lower case x and y coordinates shown in the sketch above and having origin marked by the letter o. Position of output window origin o on the page is, of course, affected by the setting of XOFFSET and YOFFSET, as explained before. The XSIZE and YSIZE (output window size) keywords of the DEVICE command are also measured in the x and y directions when in landscape orientation. The resizing of the landscape page generally means interchanging the values of XSIZE and YSIZE appropriate for the portrait page.

Example of settings for a portrait page:

```
XPAGE=8.5 & YPAGE=11. & XOFFS=0. & YOFFS=0. ;Inches

DEVICE,/INCHES,XSIZE=XPAGE,YSIZE=YPAGE,XOFFSET=XOFFS,YOFFSET =YOFFS
```

Example of settings for a landscape page:

```
  XPAGE=11.  &  YPAGE=8.5  &  XOFFS=0.  &  YOFFS=XPAGE  ;Inches

   DEVICE,/LANDSCAPE,/INCHES,XSIZE=XPAGE,YSIZE=YPAGE,XOFFSET=XO FFS, $

      YOFFSET=YOFFS
```

Example of setting position and size of a plot window:

```
  X0=1.374  &  Y0=1.283  &  XLEN=3.622  &  YLEN=6.157  ;Inches

   !P.POSITION=[X0/XPAGE,Y0/YPAGE,(X0+XLEN)/XPAGE,(Y0+YLEN)/YPA GE]
```

Example of setting position and orientation of a text string:

```
  x0=.35  &  y0=.37  ;Inches

  xyouts,x0/xpage,y0/ypage,!stime,orient=90,/normal     ;Date, time
```

T15. I get the error message "Code Area Full". What do I do?

PV-WAVE sets aside a certain amount of memory area for compiling programs.
The current code and area sizes can be seen with the INFO command, e.g.

```
 Wave> info % At $MAIN$  . Code area

       used: 0% (0/16384),  Symbol area used: 0% (2/4096)
```

These sizes can be increased with the .SIZE command. Quoting the PV-WAVE
User's Manual, section "Using Executive Commands" in Chapter 2:

> The .SIZE command resizes the code area and data area. These
> memory areas are used when PV-WAVE commands are compiled. The code
> area holds internal instruction codes that the compiler generates.
> The data area, also used by the compiler, contains variable name,
> common block, and keyword information for each compiled function,
> procedure and main program.
>
> Resizing the code and data areas erases the currently compiled
> main program and all mail program variables.
>
> For example, to extend the code and data areas to 40000 and 10000
> bytes respectively:
>
>   .SIZE 40000 10000

The upper limit for both code_size and data_size is over 2 billion bytes.

Getting "Code Area Full" is often an indication that the routine is large, and would benefit by decomposition into sub-procedures/functions. It's better to avoid use of .SIZE because your code will always work on other systems where the users don't use a large .SIZE setting.

T16. Sometimes I get the following error message:

% Unable to allocate memory: to make array. not enough core

RSI and VNI support replies:

The circumstances described happen when memory becomes fragmented. Unfortunately, there is nothing you can do except use less memory in your application, or get more for the system to work with. There are Tips in the Tips Database regarding memory problems. Additionally, memory management is discussed in chapter 11 of the PV-WAVE Programmer's Guide.

T17. How can I set the cursor to a crosshair on my display?

Although this is very possible to do in PV-WAVE, a routine which accomplishes this has not been submitted to the FAQ maintainer for inclusion here. There are two known versions of this capability available for IDL which may be ported to PV-WAVE.

T18. How can I vectorize an equation of two different arrays?

From the user community:

I have two different arrays, (8) of float and (300,8) of float. I want to vectorize the equation and therefore I need to use both arrays in the same equation. For example :

newarray=cos(small_array)*sin(large_array)

where I want the data in small_array to be used over and over 300 times in this calculation.

From Dan Carr (dan@rsinc.com):

```
IDL> arr1 = Findgen(8)
IDL> arr2 =Findgen(300, 8)
```

IDL> newarr = (Replicate(1.0, 300) # Cos(arr1)) *Sin(arr2)

From Dave Landers (landers@boulder.vni.com)

to convert an array1(M) to array2(n,M) :
array2 = array1(Lindgen(n,M) / n)
or array2 =replicate(1,n) # array1
to convert an array1(M) to array2(M,n) :
array2 = array1(Lindgen(M,n) MOD M)
or array2 = array1 # replicate(1,n)

T19. How can I get PV-WAVE and MacX to work without crashing?

Using MacX v1.2 and PV-WAVE cause the Mac to crash quite often. This happens especially during allocation of color resources or display windows. You can get around the problem by downgrading to MacX v1.1.7 (apparently Apple will supply this if you can prove to them that you rightfully own v1.2), or upgrading to MacX 1.5, available from Apple Computer, or Age Logic.

Another solution is to purchase White Pine's eXodus software. Although eXodus is not without its share of problems, it does run PV-WAVE quite well. White Pine can be contacted at:

White Pine Software
40 Simon St. Suite 201
Nashua, NH 03060-3043
Tel: 603-886-9050
Fax: 603-886-9051
http://www.wpine.com/

Yet another solution is to use Tenon's Power MachTen for PowerMacs or MachTen and X-Windows package for 68K Macs. The FAQ maintainer has tried both Tenon packages, and the non-PowerMac Apple and White Pine packages for extended periods of time and recommends Tenon. On a Mac IIfx, MachTen provides excellent response (2-5x faster) compared to both MacX and eXodus. A direct comparison between PowerMac versions of the programs was not made. The limiting factor is the cost of the software. MacX and eXodus can be found for around $250, Power MachTen (X-Windows Server included) retails for $695, and MachTen and X-Windows retail for $1045. Substantial discounts (up to 70% off) have been offered in the past during MacWorld shows. Keep in mind that you are getting a full UNIX implementation as opposed to an X-Windows emulator for this price. Tenon can be contacted at:

Tenon Intersystems
1123 Chapala Street
Santa Barbara, CA 93101

Tel: (805)963-6983
Fax: (805)962-8202
http://www.tenon.com/

T20. How can I determine if a variable is defined?

It is often useful to determine if a PV-WAVE variable is defined. The is easily done using the n_elements function which returns 0 if the given variable is undefined. This is especially useful for setting defaults for keyword parameters. Here are a couple examples:
if n_elements(start) eq 0 then start=0
if n_elements(dir) eq 0 then cd, current=dir

T21. Why should KEYWORD_SET not be used to check if a variable is defined?

In version 6.0 of PV-WAVE there is an additional function, PARAM_PRESENT, that tests for the existance of a parameter passed to a procedure or function. PARAM_PRESENT tests if the user supplied something, regardless if that paramter's variable is defined or not.

PARAM_PRESENT compliments the KEYWORD_SET and N_ELEMENTS functions. PARAM_PRESENT lets you distinguish between the two cases in which KEYWORD_SET
returns FALSE, and the two cases when N_ELEMENTS returns zero (0).

Some possible cases are:

Parameter Supplied?	Variable Defined?	Value	N_ELEMENTS()	KEYWORD_SET()	PARAM_PRESENT()
No	-	-	0	0	0
Yes	No	-	0	0	1
Yes	Yes	0	1	0	1
Yes	Yes	Non-Zero	1	1	1

If you are not using version 6.0, the following is applicable:

> From a comp.lang.idl-pvwave post by William Thompson:

The IDL function KEYWORD_SET() is only designed to be used with logical variables, i.e. those which can be either True (usually signalled with the value 1) or False (0). It has the property that if a variable is undefined, then it returns False, so people often make the mistake of using it to test whether a variable is defined or not.

To test whether a variable is defined or not, use N_ELEMENTS() instead. This

will return 0 if a variable is undefined, or some positive number otherwise.
Only use KEYWORD_SET for truly Boolean (True/False) variables.

T22. What is the undocumented routine TVRDC?

> From a comp.lang.idl-pvwave post by William Thompson:

The reason that TVRDC is undocumented is because it's not needed any more. It doesn't do anything that CURSOR doesn't do. TVRDC is only retained for compatibility with older programs.

In the old days, before X-windows, CURSOR was used to read coordinates off of line graphics terminals, and TVRDC was used to read coordinates off of image display devices. With the advent of IDL 2.0 [and PV-WAVE 2.0 - MGS], the distinctions between different kinds of graphics devices were mostly removed, and both of these functions were merged into CURSOR.

T27. Why is memory not released back to the operating system after an array is deleted?

By Eric Korpela of Berkeley

This is a result of IDL being written in C and using the C library functions (malloc and free) for memory allocation. In most C libraries, memory that is freed is NOT returned to the operating system. The C program retains this memory and will reuse it for future calls to malloc (assuming that the new allocation will fit in the freed block).

Another way of considering it is in terms of how memory allocation is done under UNIX. New memory is allocated using brk() or sbrk() which control the size of the data segment. These routines are called by malloc().

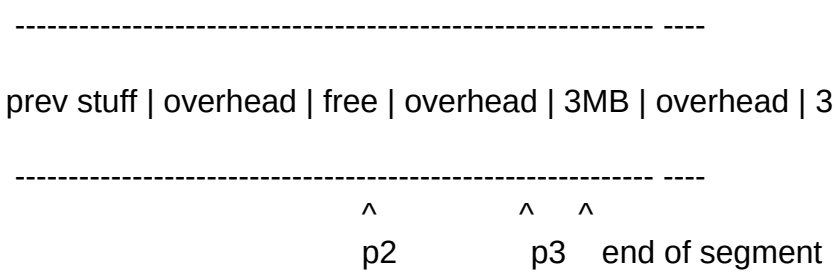
Suppose you allocate 3 1 MB regions of memory under C.

```
char *p1=(char *)malloc(3*1024*1024);
char *p2=(char *)malloc(3*1024*1024);
char *p3=(char *)malloc(3*1024*1024);
```

Here's what your data segment would look like assuming malloc had to call sbrk().

```
------------------------------------------------------------ ---
prev stuff | overhead | 3MB | overhead | 3MB | overhead | 3MB |
------------------------------------------------------------ ---
             ^          ^          ^   ^
             p1         p2         p3   end of segment.
```

Now we free(p1).



Notice that the free memory is still in the data segment. If free had called brk to reduce the size of the segment, the 3MB pointed to my p3 would be outside the data segment! SIGSEGV city! If free had moved the allocated memory to lower addresses so the segment size could be reduced without losing data, then p2 and p3 would point to invalid addresses, and we'd be forced to use handles rather than pointers and call GetPointerFromHandle() every time we wanted to access the memory. Ick! Just like Windows!

APPENDIX

A01. Disclaimer:

I do not work for VNI and I am in no way answerable to them. Questions and answers in this document are culled from the user community with some input from VNI for clarity. No warranty, express or implied exists regarding this document. Permission to copy all or part of this work is granted, provided that the copies are not made or distributed for resale.

A02. Obtaining the latest PV-WAVE FAQ

The current PV-WAVE FAQ may be accessed at fermi.jhuapl.edu [128.244.147.18] in directory www/s1r/idl/idl_faq

HTML version: pvwave_faq.html (compressed: pvwave_faq.html.Z)
Plain Text: pvwave_faq.txt (compressed: pvwave_faq.txt.Z)

The URL (Uniform Resource Locator) for this file is:
ftp://fermi.jhuapl.edu/www/s1r/idl/idl_faq/pvwave_faq.html

How to interpret the URL:

Using a WWW (World Wide Web) Browser, for example mosaic:
mosaic ftp://fermi.jhuapl.edu/www/s1r/idl/idl_faq/pvwave_faq.html

Save the file using the Save as ... option.

Using anonymous ftp:
ftp fermi.jhuapl.edu
Login: anonymous
Password: enter your email address
cd www/s1r/idl/idl_faq
get file
bye

The IDL FAQ is located at
ftp://fermi.jhuapl.edu/www/s1r/idl/idl_faq/idl_faq.html

Additions and Corrections

Send additions and corrections to:

Mike Schienle
mgs@visdata.com

Mike Schienle created the PV-WAVE FAQ based on the previously existing IDL FAQ as of 1995 May 12. As of 1994 Oct 27 Ray Sterner has been maintaining the IDL FAQ. Mike Schienle was handling the IDL FAQ previously, and Patrick Ryan before him.

A03. Many thanks to the following for their contributions

black@breeze.rsre.mod.uk (John Black)
claflin@itsa.ucsf.edu (Scott Claflin)
edelsohn@npac.syr.edu (David Edelsohn)
fireman@iuegtc.DNET.NASA.GOV (Gwyn Fireman)
gurman@umbra.gsfc.nasa.gov (Joseph B. Gurman)
jdlb@kukui.IFA.Hawaii.Edu (JF Pitot de La Beaujardiere)
kashyap@oddjob.uchicago.edu (Vinay Kashyap)
mayor@vaxine.larc.nasa.gov (Shane Mayor)
oet@maz.sma.ch (Thomas Oettli)
rmmoss@Texaco.COM (Robert M. Moss)
sterne@dublin.llnl.gov (Philip Sterne)
thompson@serts.gsfc.nasa.gov (William Thompson)
valenti@soleil.Berkeley.EDU (Jeff Valenti)
sterner@tesla.jhuapl.edu (Ray Sterner)
joel@stars.gsfc.nasa.gov (Joel Parker)
landers@boulder.vni.com (David Landers)
dan@rsinc.com (Dan Carr)
denisef@rsinc.com (Denise Fields)

Previous PV-WAVE FAQ maintainers:
None.

Previous IDL FAQ maintainers:
Ray Sterner: 10/28/94 to present
Mike Schienle: 12/01/93 to 10/27/94
Patrick Ryan: Created the IDL FAQ

A04. PV-WAVE FAQ Versions History

1.1 (2/22/96)
Added additional comments throughout due to a 7 month hiatus from updating the FAQ.

1.0 (7/25/95)
First PV-WAVE FAQ.

This version of the PV-WAVE FAQ was derived from the IDL FAQ. There will be some effort to keep the two FAQ's similar and to cover topics in both FAQs which are relevant to both languages.

The End

--
Mike Schienle
Custom Data Visualizationsmgs@visdata.com
mgs@visdata.com