Subject: IDL (Interactive Data Language) FAQ
Posted by pat on Tue, 27 Jul 1993 20:01:04 GMT
View Forum Message <> Reply to Message

Archive-name: idl-faq
Last-modified: 1993/06/17
Version: $Id: idl-faq,v 1.5 1993/06/17 19:34:13 pat Exp pat $

This is a list of Frequently Asked Questions about the Interactive Data Language or IDL. These questions pop up fairly regularly in the newsgroup comp.lang.idl-pvwave. This list is an attempt to cut down on net.traffic regarding commonly asked questions. Users are encouraged to read through this list before posting a query to the newsgroup.

[SPECIAL NOTE: As of this writing, the editor has not gotten his hands on IDL 3.1. Bug reports and questions about IDL 3.1 are especially welcome at this time.]

A Note from the editor about PV~WAVE:

I do not have any direct experience with PV~WAVE. As such, I will try to minimize comments which appear to favor one package or the other. I will, however, welcome concise descriptions of technical and functional differences between the two packages.

If you are viewing this text in a GNU Emacs Buffer, you can type "M-2 C-x $" to get an overview of just the questions. Then, when you want to look at the text of the answers, just type "C-x $".

The following questions are answered:

==

GENERAL:

G01. What is IDL?

IDL is the Interactive Data Language. It is a product of Research Systems, Inc. (RSI).

The following is quoted from the README file at rsinc.com:/pub/idl/README. As such, it describes IDL's capabilities in an understandably subjective manner. ;-)

{begin quote}
IDL, Interactive Data analysis Language, is a complete package for the interactive reduction, analysis, and visualization of scientific data and images. Optimized for the workstation environment, IDL integrates a responsive array oriented language with numerous data analysis methods and an extensive variety of two and three dimensional displays into a powerful tool for researchers.

IDL supports an extensive data import capability, publication quality hard copy output, and user-defined Motif graphical user interfaces.

Users can create complex visualizations in hours instead of weeks with the aid of IDL's high level capabilities and interactive environment.

IDL is useful in physics, astronomy, image and signal processing, mapping, medical imaging, statistics, and other technical disciplines requiring visualization of

large amounts of data.
{end quote}

Here is a short history of RSI:
[attributed to ali@rsinc.com (Ali Bahrami)]

{begin quote}
IDL is a product of Research Systems, Inc., founded in 1977
by David Stern. The origins of IDL were developed at the Laboratory
for Atmospheric and Space Physics (LASP) at the University
of Colorado. David was one of the people involved in efforts to
make computers easier to use for the physicists at the Lab.
The first program in the evolutionary chain to IDL was named Rufus
(named after Dave's dog). Rufus was a very simple vector oriented
calculator that ran on the PDP-12. It accepted 2 letter codes that
specified (1) An arithmetic operation (2) The input registers to
serve as operands, and (3) the destination register. The next version
was the Mars Mariner Spectrum Editor (MMED) which was a version
of Rufus that ran on the PDP-8.

The next program in this line was named SOL, and it also ran on
the PDP-8. Unlike its predecessors, SOL was a real computer language
with a real syntax (no more 2 letter codes). It was an APL influenced
array oriented language with some primitive graphics capabilities.
The resemblance to IDL was there, but very faintly.

In 1977, Dave left LASP to start Research Systems Inc. (RSI) with the
intention of building on the ideas contained in SOL. The initial
result of this endeavor was PDP-11 IDL, which was much more capable than
SOL. Graphics was usually done on Tektronix terminals and outboard
raster graphics displays. I used this version at LASP in 1981 on a
PDP11/34 under RSX-11M in 1981 (I worked as a student at LASP from
1981 to 1987). I didn't use it for very long though,
because 1981 was the year that Dave released the VAX/VMS version of
IDL. This version, which was written in VAX-11 MACRO and FORTRAN,
took advantage of the VAX virtual memory and 32-bit address space,
and was a huge step beyond the PDP-11 version. It used essentially
the same sort of graphics hardware as the PDP-11.

In 1987, Dave decided that Unix workstations were the direction in
which IDL should progress, but porting the current VAX IDL to Unix
didn't make much sense because of its MACRO and FORTRAN implementation.
I had just finished my Masters degree and was looking for work.
Dave hired me and together we wrote the current version of IDL for
Unix on the Sun 3 taking advantage of the re-write to extend and
improve the language. Since then, we've ported it to many Unix
machines and moved it back to VMS. RSI has many other employees now,
but our focus is still the continued development of IDL. Recently,

IDL was ported to PC class systems running Microsoft Windows.
{end quote}

G02. Where can I contact them?

Their address is:

Research Systems, Inc.
777 29th Street, Suite 300
Boulder, CO 80303
(303) 786-9900   (Voice)

Email:
info@rsinc.com or support@rsinc.com   # Internet
ORION::IDL   # SPAN

G03. How do I get IDL?

RSI's distribution scheme is unique in that all of the binaries and IDL you need are available via anonymous ftp. IDL binaries and code are available at these sites:

gateway.rsinc.com (192.5.156.17)
pub/idl
boulder.colorado.edu (128.138.240.1)
pub/idl
lumpi.informatik.uni-dortmund.de (129.217.36.140)
pub/idl

The README file describes which files are needed, how to unpack them, and how to install them.

If you install IDL without a valid license, you will get IDL's 7 minute demo mode. This mode is designed for users who are considering buying the package.

To actually get IDL running for good, you must pay for a license from RSI and follow their instructions. You will be asked to fill out a form with information unique to your machine. RSI will create a license key which the license manager program (lmgrd) reads to validate your license.

As of release 3.1, upgrades to IDL are no longer free. For details about upgrades and support contracts, contact RSI.

G04. What is the current version of IDL?

IDL 3.1.0 is the current version. It was released June 6, 1993.

There is no word about when future releases are expected.

G05. On what systems does IDL run?

The information below is from the file RELEASE_LEVEL located at rsinc.com:/pub/idl/RELEASE_LEVEL. Ths information applies to release 3.1.

{begin quote}
This release supports the following systems:

- Convex C2 and C3: ConvexOS 10.0.5.
- Data General Aviion: DG/UX 5.4.1 and later
- DEC ALPHA: OSF1 1.2.
- DOS based personal computers running Microsoft Windows 3.1 or later
- HP 9000: HP-UX 8.0 on Series 300, 400 and 700.
- IBM 6000: AIX 3.2.
- MIPS: Risc/OS 4.52B.
- Risc Ultrix: Ultrix 4.2.
- SGI: IRIX 4.0
- Sun 3: SunOS 4.1.1. Widgets are built using OpenWindows 2.0.
- Sun 4 (sparc):
    [SunOS 4.1.0] Widgets are built using OpenWindows 3.0.
    [SunOS 5.1 (Solaris 2.1)] Widgets are built using OpenWindows 3.1.
- VMS:
    [ALPHA] OpenVMS AXP 1.0.
    [VAX] VMS 5.1 and up. Standard X windows graphics are supported at all VMS versions. IDLwidgets are supported at VMS 5.4-3 at up *if* the host system has the DECWINDOWS-MOTIF package installed.

The following hardware/operating system combinations are no longer supported. The last release of these versions have been archived, and no future development for them will be done:

- Sun 3 running SunOS 3.5 (last release: 2.0.4).
- Sun 386i running SunOS 4.0.1 (last release: 2.0.4).
- Sun 3 and Sun 4 running SunOS 4.0 (last release: 2.2.1)
- HP 9000 series 300 and 400 running HP-UX 7.0 (last release: 2.2.2)
- VAX Ultrix (last release: 2.2.2)

{end quote}

G06. What is PV~WAVE and how is it related to IDL?

Around the time that the Unix version of IDL first became available (1988), Precision Visuals Inc. (PVI) entered into an agreement with RSI under which they enhanced and resold IDL under the name PV~WAVE. In September of 1990, they exercised an option in that agreement that resulted in the following:

- They received a copy of the IDL source code as it existed in September 1990 in return for a one-time payment to RSI.

- The connection between RSI and PVI was severed.

IDL and PV~WAVE are now on separate development tracks. Each company enhances, supports, and maintains its own product.

PVI has since merged with IMSL and is now Visual Numerics, Inc. (VNI).

G07. Are there anonymous FTP sites for IDL?

The sites below contain public domain IDL code.

JHU/APL IDL library
nansen.jhuapl.edu [128.244.147.14]
/pub/idl-pvwave/jhuapl

NASA IDL Astronomy User's Library
idlastro.gsfc.nasa.gov [128.183.57.82]
/

IUE RDAF library
iuesn1.gsfc.nasa.gov [128.183.57.16]
cetus.colorado.edu [128.138.238.151]
/pub

ICUR Spectral Analysis Software
ftp.astro.psu.edu [128.118.147.28]
/pub/nefftp/icur

IDL ROSAT software
rosserv.gsfc.nasa.gov [128.183.8.43]
/pub/IDL

IDLmeteo library
ftp.sma.ch (141.249.3.33)
/pub/idlmeteo

G08. How can I get help?

RSI has excellent telephone and email support. You can contact them at:

Voice:
(303) 786-9900

Fax:
(303) 786-9909

Email:
(Internet)
info@rsinc.com # general questions
support@rsinc.com # technical support

(SPAN)
ORION::IDL

Keep in mind, however, that RSI's tech support is for their paying customers, i.e. those with current support contracts.

G09. Why are there two newsgroups for IDL?

Unfortunately, there are two very different packages with the abbreviation "IDL". The newsgroup comp.lang.idl is for the Interface Description Language. The newsgroup for discussing issues related to RSI's IDL and VNI's PV~WAVE and IMSL/IDL is comp.lang.idl-pvwave.

G10. Does anyone at RSI read this group? Is anyone there listening?

[This question was included at the request of RSI. The answer was provided by Ali Bahrami. -pat]

{begin quote}
Yes, many of us do. We're naturally curious what people think of our product. We make notes about what people like and dislike and this influences our decisions.

However, you usually won't get a direct response from us from a posting to this group. There are many reasons for this. Here are a couple of the more important ones:

[] We believe that this group should belong solely to the user community, and should be free of vendor bias and marketing. It should be noted that both RSI and Visual Numerics (formerly PVI) have shown great restraint in this matter, and that this group is largely left to the actual users. (Long term readers will recall some notable exceptions to this, but in general it is true.)

It could be argued that as long as the topic stays technical, vendor postings are OK. The problem with this is that one persons technical posting is another's blatant product plug, and the line between them is not always obvious.

[] We provide support for our customers via the phone and email. If you would like an answer from us, you should call us directly. We have no objection to you sharing the information you get in this manner with the newsgroup as long as you quote us accurately and separate fact from conjecture.

In other words, you should view this newsgroup as a way to share questions and information with other users, not as a way to contact the vendor.

{end quote}

G11. When is the next version of IDL due out?

IDL 3.1 was release in June, 1993. There is no word as to when the next release is expected

==

TECHNICAL QUESTIONS:

Note for Sun users:

There is a mini FAQ for Sun IDL in $IDL_DIR/notes, and the following questions are answered in the file openwin3.doc:
(1) IDL complains about missing fonts when creating widgets.
(2) IDL complains about missing bitmap files when creating widgets.
(3) How to make IDL work with OpenWindows version 3.0.
(4) 3-D appearance for OpenLook IDLwidgets.
(5) Why does one of the widgets appear red?
(6) Why does pressing the left button while pointing at a pulldown menu button cause the menu to pop up with the pushpin in? Older versions of IDL didn't to that.
(7) The font used by the list widget is much larger than it used to be, and I don't like the result on my program.
(8) Why do I get "Cannot allocate colormap entry" errors, and why do they kill IDL?
(9) I've changed my .Xdefaults file, but nothing different happened. Why?

T01. Why doesn't polycontour fill open contours?

This problem is described in the POLYCONTOUR manual page.

{begin quote}
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.
```

{end quote}

[This problem was fixed in IDL 3.1.]

T02. How do I increase the number of commands stored in the history buffer?

The system variable !EDIT_INPUT controls command recall. By default, it is set to 1, causing the last 20 commands to be saved. If it is 0, no commands are saved. To save more than 20 commands, just put !EDIT_INPUT=50 (or other large number) in your startup file.

It is important to realize that IDL looks at the value of !EDIT_INPUT the first time it reads anything from the keyboard, and the size of the history buffer is fixed after that. Hence, the command must be in a startup file because entering it at the keyboard is too late.

T03. How do I get IDL to call routines in language X, running under system Y?

Jeff Valenti has written a sizable document about calling external FORTRAN routines from IDL. You can find it at jaameri.gsfc.nasa.gov:/pub/idl/idl-fortran.

T04. Why does XPALETTE edit my color table incorrectly?

Here is the answer from RSI support:

{begin quote}
The color applications such as xpalette and xloadct use a common block called "colors" to keep track of the color vectors. When you call tvlct, your vectors are loaded into the colormap, but they are not put into the colors common block.

When xpalette starts, it checks to see if the colors have been defined in the common block and uses them if they have. Otherwise it sets them to the standard black and white colormap, in which r,g,b are each linear ramps. (The colors are as you expect because there is only one colormap.)

Admittedly, this is not the most desirable situation. You would like your colors which you loaded with TVLCT to be recognized by xpalette. We modified xpalette (and xloadct) to use the current colormap when they start up by getting the current vectors with TVLCT.

Another alternative would be to use the following lines to define the common block prior to calling your current version of xpalette.

[assume here that you have a routine called "restore" which reads colors from a file somewhere and creates vectors r, g, and b. -pat]

```
IDL> restore, file='ryan.sav', r, g, b   ;get the vectors from somewhere
IDL> tvlct, r, g, b
IDL> common colors, r_orig, g_orig, b_orig, r_curr, g_curr, b_curr
IDL> r_orig = r & r_curr = r
IDL> g_orig = g & g_curr = g
IDL> b_orig = b & b_curr = b
IDL> xpalette
```

{end quote}

T05. Is there on-line help for IDL?

Try ?.

T06. I run IDL 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?

[Note: This is only a problem under OpenWindows2. -pat]

Add the following to your .Xdefaults:

```
Idl*colors: -5
```

which reserves some colors for the colormap so that IDL does not exhaust all the available colors. (For a nice summary of Sun IDL interactions with OpenWindows, see $IDL_DIR/notes/openwin3.doc)

Another solution is to 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 IDL User's Guide, page 10-8:

{begin quote}
IDL 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, IDL'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 IDL is in and which program unit called it).
{end quote}

T08. Is there a major mode for editing IDL code in Emacs?

No and yes.
VNI has developed a PV~WAVE major mode, which they have made available under the FSF copyleft. Given the (remaining) strong similarity between PV~WAVE and IDL, it has many of the features that one would like in such a mode.
It is available by anonymous ftp from ftp.pvi.com (128.138.213.22) in the ./PVI/emacs directory

[wave-mode.el seems to work pretty well. Be warned, however, that it is a memory hog and it will do a lot of garbage-collection. -pat]

T09. How do I get 3-D widgets under OpenLook 3.0?

This subject is discussed on page 53 of the OpenWindows Version 3 User's Guide Release Manual.

OpenWindows 3.0 added 3-D appearance for widgets. In order for the 3-D look to work, it must be enabled and the background color

must be a medium tone color such as "grey" or "wheat".
Add the following resources to your ~/.Xdefaults file:

For plain IDL:

```
Idl*threeD:     TRUE
Idl*background: PeachPuff2
```

For IMSL/IDL:

```
Imslidl*threeD:     TRUE
Imslidl*background: PeachPuff2
```

T10. Why does one of the widgets appear red under OpenWindows?

This subject is discussed on page 53 of the OpenWindows Version 3 User's Guide Release Manual.

The OLIT widget toolkit, which is used by IDL, added the concept of "mouseless focus" under version 3.0. The red widget indicates where the current mouseless focus is. Use the arrow keys to traverse the widgets, and the spacebar to make a selection.

T11. Where are all the IDL routines and userlib procedures?

The basic routines are not accessible, for obvious reasons. The userlib, statlib and widget procedures are in $IDL_DIR/lib/. The procedure XDL also displays the full pathname. The system variable !path also contains the directory names for all accessible IDL procedures.

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. I don't know about PV-Wave, but the following works in IDL:

```
; 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 4 images, using i as position index
   tv, data(*,*,i), i, xsize=ximsize, ysize=yimsize
endfor
```

T13. Does case matter in IDL?

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 IDL 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.)

IDL 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.

```
   ----------
   |       |  ORIGINAL PORTRAIT PAGE  (Printer only prints on
 Y |       |        OUTPUT WINDOW                 this area.)
   |       |
   |  X    |
   o-----------
   |   y   + |  OUTPUT WINDOW ROTATED ABOUT LOWER LEFT CORNER
 x |       + |
   |       + |
   ------------
   ++++++++++    OUTPUT WINDOW RESIZED FOR LANDSCAPE ORIENTATION
       ^
       |   DIRECTION OF NEEDED OFFSET
```

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?

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

```
IDL> help
% 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 IDL User's Manual, page 2-11:

{begin quote}
These sizes represent a compromise between an unlimited program space and conservation of memory. User procedures and functions are compiled in this large program area. After successful compilation, a new memory are of the required size is allocated to contain the newly compiled program unit.

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 30000 and 5000 bytes respectively:

```
            .SIZE 30000 5000
```

{end quote}

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 support replies:

{begin quote}
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.
{end quote}

==

Disclaimer:

I do not work for RSI and I am in no way answerable to them. Questions and

This file can always be found at jaameri.gsfc.nasa.gov [128.183.88.75] in pub/idl/idl-faq. A URL entry for this file would look like:

file://jaameri.gsfc.nasa.gov/pub/idl/idl-faq

Additions and corrections should be sent to:

idl-faq@jaameri.gsfc.nasa.gov

Many thanks to the following for their contributions.

black@breeze.rsre.mod.uk (John Black)
claflin@claes.space.lockheed.com (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)

--

patrick m. ryan
nasa / goddard space flight center / oceans and ice branch / hughes stx
pat@jaameri.gsfc.nasa.gov / patrick.m.ryan@x500.gsfc.nasa.gov