Subject: IDL 8.2.2 released
Posted by Mark Piper on Tue, 05 Feb 2013 17:39:54 GMT
View Forum Message <> Reply to Message

I'm happy to announce that IDL 8.2.2 is available for download from our website, www.exelisvis.com. I have a short write-up of what's new on the IDL blog:

http://idldatapoint.com/2013/02/05/idl-8-2-2-released/

including a link to download a set of example programs.

mp

Subject: Re: IDL 8.2.2 released
Posted by wlandsman on Tue, 05 Feb 2013 22:28:24 GMT
View Forum Message <> Reply to Message

I happy to see a new timestamp() function in V8.2.2, but those of using David Fanning's timestamp() function (http://www.idlcoyote.com/programs/timestamp.pro) need to be careful to avoid a conflict.
--Wayne

On Tuesday, February 5, 2013 12:39:54 PM UTC-5, Mark Piper wrote:
> I'm happy to announce that IDL 8.2.2 is available for download from our website, www.exelisvis.com. I have a short write-up of what's new on the IDL blog:
>
>
>
> http://idldatapoint.com/2013/02/05/idl-8-2-2-released/
>
>
>
> including a link to download a set of example programs.
>
>
>
> mp

Subject: Re: IDL 8.2.2 released
Posted by David Fanning on Tue, 05 Feb 2013 23:06:19 GMT
View Forum Message <> Reply to Message

Wayne Landsman writes:

> I happy to see a new timestamp() function in V8.2.2, but those of using David Fanning's timestamp() function (
> http://www.idlcoyote.com/programs/timestamp.pro) need to be careful to avoid a conflict.

To be honest, I see a LOT of things in this new version of IDL that have been in the Coyote Library for a long time. Always exciting to me to be recognized (if still usually unacknowledged). ;-)

I don't know if you have noticed, but when you get to be a certain age, you become invisible to young women (at Starbuck's, for example). This is the opposite of that. Someone is finally paying attention to what has been happening for the past couple of years.

I'm reminded of one of my favorite lines in Barry Lopez's story, Drought, from his book, River Notes:

> As deer and coyote sipped from the same tiny pool they abrogated their agreement, and the deer contemplated the loss of the coyote as he would the loss of a friend; for the enemy, like the friend, made you strong.

Anyway, I think borrowing good ideas from the Coyote Library augurs well for the continued success of IDL. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: IDL 8.2.2 released
Posted by wlandsman on Wed, 06 Feb 2013 07:29:36 GMT
View Forum Message <> Reply to Message

On Tuesday, February 5, 2013 6:06:19 PM UTC-5, David Fanning wrote:

> Anyway, I think borrowing good ideas from the Coyote Library augurs well
>
> for the continued success of IDL. :-)
>

But in the spirit of "No good deed goes unpunished", the first program I tried in V8.2.2 crashed because it was expecting the calling sequence of the Coyote TimeStamp() function, rather than

the different calling sequence of the ITTVIS TimeStamp() function, now ahead in the !PATH.

--Wayne

Subject: Re: IDL 8.2.2 released
Posted by Fabzi on Wed, 06 Feb 2013 09:44:47 GMT
View Forum Message <> Reply to Message

A bit frustrating though that the SVN bug has not been fixed yet, even
if exelis writes about it on its website:

http://tinyurl.com/brj77z4

On the bright side, the ROI bug I mentioned a few months ago has been
addressed:

67992 IDLanROIGroup caused IDL to shut down. Updated to handle greater
than 1024 IDLanROIs

https://groups.google.com/forum/?fromgroups=#!topic/comp.lan g.idl-pvwave/QoaALkXGMPQ

Fab

On 02/05/2013 06:39 PM, Mark Piper wrote:
> I'm happy to announce that IDL 8.2.2 is available for download from our website,
www.exelisvis.com. I have a short write-up of what's new on the IDL blog:
>
> http://idldatapoint.com/2013/02/05/idl-8-2-2-released/
>
> including a link to download a set of example programs.
>
> mp
>

Subject: Re: IDL 8.2.2 released
Posted by lecacheux.alain on Wed, 06 Feb 2013 12:29:54 GMT
View Forum Message <> Reply to Message

Le mardi 5 février 2013 18:39:54 UTC+1, Mark Piper a écrit :
> I'm happy to announce that IDL 8.2.2 is available for download from our website,
www.exelisvis.com. I have a short write-up of what's new on the IDL blog:
>
>

>
>   http://idldatapoint.com/2013/02/05/idl-8-2-2-released/
>
>
>
> including a link to download a set of example programs.
>
>
>
> mp

By running with 8.2.2 a program recently developed with 8.2.1, I can see a spectacular improvement in speed when using NG graphics.
Here is an excerpt involving a scatter plot of about 1500000 data points.
Now, NG appears to be even faster than DG !

```
IDL> tic & pl = plot(st[3,*], /XLOG, sqrt(total(st[0:2,*]^2, 1))/st[3,*], $
IDL>   /YLOG, SYMBOL='dot', YRANGE=[1e-3,1e3], LINESTYLE=6) & toc
% Time elapsed: 5.3750000 seconds.

IDL> tic & plot, st[3,*], /XLOG, sqrt(total(st[0:2,*]^2, 1))/st[3,*], $
IDL>   /YLOG, PSYM=3, YRANGE=[1e-3,1e3] & toc
% Time elapsed: 6.3440001 seconds.
```

alain.

Subject: Re: IDL 8.2.2 released
Posted by Fabzi on Wed, 06 Feb 2013 14:37:26 GMT
View Forum Message <> Reply to Message

On 02/06/2013 08:29 AM, wlandsman wrote:
> But in the spirit of "No good deed goes unpunished", the first program
>I tried in V8.2.2 crashed because it was expecting the calling
sequence of
>the Coyote TimeStamp() function, rather than the different calling
sequence
> of the ITTVIS TimeStamp() function, now ahead in the !PATH.
>

Yes... I've been renaming Coyote's TIMESTAMP into cgTIMESTAMP to be able to open my netcdf files, but only the Big Boss (i.e David) can decide what the best name for it now. Is David's TIMESTAMP obsolete?

Subject: Re: IDL 8.2.2 released
Posted by David Fanning on Wed, 06 Feb 2013 14:51:30 GMT

Fab writes:

> Yes... I've been renaming Coyote's TIMESTAMP into cgTIMESTAMP to be able
> to open my netcdf files, but only the Big Boss (i.e David) can decide
> what the best name for it now.

Yes, this is mere moments from being re-released as cgTimeStamp. I'm
just searching for all the places it is used in Coyote Library code
and on my web page now.

> Is David's TIMESTAMP obsolete?

Yes, for the millions of people who own IDL 8.2.2. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Wed, 06 Feb 2013 17:18:06 GMT

On Tuesday, February 5, 2013 4:06:19 PM UTC-7, David Fanning wrote:
>
> To be honest, I see a LOT of things in this new version of IDL that have
> been in the Coyote Library for a long time. Always exciting to me to be
> recognized (if still usually unacknowledged). ;-)
>

Hi David,

This concerns me, considering we take attribution seriously. In fact, for the new routines we implement, we're adding a References section at the end of the Help page. Check IMAGE_THRESHOLD, for example.

A timestamp routine, however, is about as generic a routine as is possible. So no, we did not

borrow this idea from you; in fact, the ENVI team developed this routine for their needs and gave it to us for inclusion in IDL.

Perhaps you could provide a list of other "things in this new version of IDL that have been in the Coyote Library for a long time"? I'll do my best to address any concerns.

mp

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Wed, 06 Feb 2013 17:20:31 GMT
View Forum Message <> Reply to Message

On Wednesday, February 6, 2013 2:44:47 AM UTC-7, Fab wrote:
> A bit frustrating though that the SVN bug has not been fixed yet, even
> if exelis writes about it on its website:
>
> http://tinyurl.com/brj77z4
>

Fab,

We're in the process of upgrading the version of Eclipse we use for the DE. In doing so, we're also updating our SVN plugin, which should take care of this bug. This work is scoped for 8.3, but it may happen earlier.

mp

Subject: Re: IDL 8.2.2 released
Posted by David Fanning on Wed, 06 Feb 2013 18:24:44 GMT
View Forum Message <> Reply to Message

wlandsman writes:

> But in the spirit of "No good deed goes unpunished", the first program I tried in V8.2.2 crashed because it was expecting the calling sequence of the Coyote TimeStamp() function, rather than the different calling sequence of the ITTVIS TimeStamp() function, now ahead in the !PATH.

I've updated the TimeStamp program in the Coyote Library to use the name cgTimeStamp. While I was doing it, I used the opportunity to retire FSC_Normalize and reincarnate this program as cgNormalize. Updated versions of the Coyote Library and other libraries (Catalyst, CoyotePlus, and Coyote_Retired) can be found in the usual places.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: IDL 8.2.2 released
Posted by Fabzi on Wed, 06 Feb 2013 19:18:21 GMT
View Forum Message <> Reply to Message

On 02/06/2013 06:18 PM, Mark Piper wrote:
> Perhaps you could provide a list of other "things in this new version
> of IDL that have been in the Coyote Library for a long time"?
> I'll do my best to address any concerns.

I'll partly answer for David because I noticed myself some similarities:

BOXPLOT -> cgBoxplot
SCATTERPLOT -> cgScatterPlot
IMAGE_THRESHOLD -> cgOTSU_THRESHOLD

These are surely not "inventions" from David, but things people have
been using intensively in the last years or so, since they were not
available per default in IDL...

Subject: Re: IDL 8.2.2 released
Posted by Louis Giglio on Wed, 06 Feb 2013 21:01:34 GMT
View Forum Message <> Reply to Message

Fab wrote:

> On 02/06/2013 06:18 PM, Mark Piper wrote:
>> Perhaps you could provide a list of other "things in this new version
>> of IDL that have been in the Coyote Library for a long time"?
>> I'll do my best to address any concerns.
>
> I'll partly answer for David because I noticed myself some similarities:
>
> BOXPLOT -> cgBoxplot
> SCATTERPLOT -> cgScatterPlot

> IMAGE_THRESHOLD -> cgOTSU_THRESHOLD
>
> These are surely not "inventions" from David, but things people have
> been using intensively in the last years or so, since they were not
> available per default in IDL...

I'm going to de-lurk after many years to comment on this. I'm not sure how many routines are David's, but what I have noticed is an increasing tendency for IDL releases to co-opt the functionality of a third-party routine that was in use long before the feature was formally added to IDL. Examples include LEGEND and COLORBAR. LEGEND is a particulary good example because F. K. Knight's version has been available for ~18 years before an official (and incompatible) version was added to IDL.

Co-opting the names of existing routines of course creates headaches because of the resulting naming conflicts.

While I suppose it's good that 8.2.2 includes a new BOXPLOT routine, I note that it conflicts with Martin Schultz's version which I've used for 13 years. It would be nice if each new release of IDL didn't clobber another existing routine in the process.

Subject: Re: IDL 8.2.2 released
Posted by Bob[4] on Wed, 06 Feb 2013 21:23:04 GMT
View Forum Message <> Reply to Message

I wonder if the very long existing bug where INTERPOLATE internally truncates all results to single precision even when double precision input is used is still present (note that it still returns the results in a double precision variable but half the precision is gone). It is in 8.1 so I have little hope it is gone in 8.2.2.

Am I the only one that has been burned by this?

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Wed, 06 Feb 2013 23:23:34 GMT
View Forum Message <> Reply to Message

On Wednesday, February 6, 2013 12:18:21 PM UTC-7, Fab wrote:
> On 02/06/2013 06:18 PM, Mark Piper wrote:
>
> I'll partly answer for David because I noticed myself some similarities:
>
> BOXPLOT -> cgBoxplot
> SCATTERPLOT -> cgScatterPlot
> IMAGE_THRESHOLD -> cgOTSU_THRESHOLD
>
> These are surely not "inventions" from David, but things people have

> been using intensively in the last years or so, since they were not
> available per default in IDL...

This is exactly my point, Fab: we've had requests for these particular routines for awhile, and we're now getting them into IDL. You may recall that I was excited to mention IMAGE_THRESHOLD when the topic of Otsu's method came up here a few months ago:

https://groups.google.com/d/msg/comp.lang.idl-pvwave/Vg2CTTj ZXnU/ifZRiymK92IJ

David does a good job of quickly turning around user requests such as these; we take a little longer.

mp

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Thu, 07 Feb 2013 00:00:32 GMT
View Forum Message <> Reply to Message

On Wednesday, February 6, 2013 2:01:34 PM UTC-7, Louis Giglio wrote:
>
> I'm going to de-lurk after many years to comment on this. I'm not sure how many routines are David's, but what I have noticed is an increasing tendency for IDL releases to co-opt the functionality of a third-party routine that was in use long before the feature was formally added to IDL. Examples include LEGEND and COLORBAR. LEGEND is a particulary good example because F. K. Knight's version has been available for ~18 years before an official (and incompatible) version was added to IDL.
>
> Co-opting the names of existing routines of course creates headaches because of the resulting naming conflicts.
>
> While I suppose it's good that 8.2.2 includes a new BOXPLOT routine, I note that it conflicts with Martin Schultz's version which I've used for 13 years. It would be nice if each new release of IDL didn't clobber another existing routine in the process.

Hi Louis,

This is a tough one, and IDL's blessing/curse of a single namespace is the problem. Because of this, when I started using IDL (and later when I taught IDL), it was drilled into me that if you release a routine into the wild, you should use a namespace identifier to separate it from another user's routine which may have the same name.

I empathize with the problems that we've created in recent releases by tromping on established routines like LEGEND, COLORBAR and now BOXPLOT and TIMESTAMP, and others. However, I assert that since we (VIS) make IDL, we own the primary
namespace. I think this is necessary to continue to move the language forward.

A possible workaround to allow you to call Martin Schultz's version of BOXPLOT would be to

compile it in a startup file:

https://groups.google.com/d/msg/comp.lang.idl-pvwave/6ibcyYf UXzA/WJW6eYZLP5YJ

mp

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Thu, 07 Feb 2013 00:09:57 GMT
View Forum Message <> Reply to Message

I love the fact that people care enough about IDL (like I do) to post comments on this thread! Please keep them coming. I'll try to respond to each comment in a critical and responsible manner.

Thanks,
mp
IDL Product Manager
mark.piper@exelisvis.com

Subject: Re: IDL 8.2.2 released
Posted by David Fanning on Thu, 07 Feb 2013 00:24:17 GMT
View Forum Message <> Reply to Message

Mark Piper writes:

> I empathize with the problems that we've created in recent releases by tromping on established routines like LEGEND, COLORBAR and now BOXPLOT and TIMESTAMP, and others. However, I assert that since we (VIS) make IDL, we own the primary
> namespace. I think this is necessary to continue to move the language forward.

I don't argue with this. I just think it would be nice if it were done
in a more neighborly way, with some advance notice so library providers
don't have to drop everything they are doing to make their customers
whole on a moment's notice.

One summer I was looking out the bank of window's on the back side of my
house at the tall trees and Long's Peak in the distance, when I noticed
all the big trees in front of my view being felled. They were replaced
with a tall, white concrete block wall. I still don't speak to that
neighbor, and no one in the neighborhood invites them to the block
parties we have on the Fourth of July. They, too, owned the biggest,
most expensive house on the block. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: IDL 8.2.2 released
Posted by Andy Sayer on Thu, 07 Feb 2013 14:22:20 GMT
View Forum Message <> Reply to Message

Also de-lurking to agree with Louis' comment: one of the reasons my group haven't updated from IDL 7 to IDL 8 is because the update breaks compatability with pre-existing code which we have been using for years, because of routines like this.

I know that's partially a code management issue which is our fault (trained as scientists first and programmers second, so some things such as namespace identifiers did not occur to us or others until we had been using IDL for some years), but as they say hindsight is 20:20.

On Wednesday, February 6, 2013 4:01:34 PM UTC-5, Louis Giglio wrote:
> Fab wrote:
>
>
>
>> On 02/06/2013 06:18 PM, Mark Piper wrote:
>
>>> Perhaps you could provide a list of other "things in this new version
>
>>> of IDL that have been in the Coyote Library for a long time"?
>
>>> I'll do my best to address any concerns.
>
>>
>
>> I'll partly answer for David because I noticed myself some similarities:
>
>>
>
>> BOXPLOT -> cgBoxplot
>
>> SCATTERPLOT -> cgScatterPlot
>
>> IMAGE_THRESHOLD -> cgOTSU_THRESHOLD
>

>>
>
>> These are surely not "inventions" from David, but things people have
>
>> been using intensively in the last years or so, since they were not
>
>> available per default in IDL...
>
>
>
> I'm going to de-lurk after many years to comment on this. I'm not sure how many routines are David's, but what I have noticed is an increasing tendency for IDL releases to co-opt the functionality of a third-party routine that was in use long before the feature was formally added to IDL. Examples include LEGEND and COLORBAR. LEGEND is a particulary good example because F. K. Knight's version has been available for ~18 years before an official (and incompatible) version was added to IDL.
>
>
>
> Co-opting the names of existing routines of course creates headaches because of the resulting naming conflicts.
>
>
>
> While I suppose it's good that 8.2.2 includes a new BOXPLOT routine, I note that it conflicts with Martin Schultz's version which I've used for 13 years. It would be nice if each new release of IDL didn't clobber another existing routine in the process.

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Thu, 07 Feb 2013 16:16:21 GMT
View Forum Message <> Reply to Message

On Wednesday, February 6, 2013 5:24:17 PM UTC-7, David Fanning wrote:
>
> I don't argue with this. I just think it would be nice if it were done
> in a more neighborly way, with some advance notice so library providers
> don't have to drop everything they are doing to make their customers
> whole on a moment's notice.
>

Hi David,

I like this idea. From now on, three weeks before a release (when we're certain what's going into it), I'll publish a list of new routines. This should give some advance time for library maintainers to check their routines. Would this work?

mp

Subject: Re: IDL 8.2.2 released
Posted by Matt[2] on Thu, 07 Feb 2013 16:20:55 GMT
View Forum Message <> Reply to Message

Mark Piper <mpiper@ittvis.com> writes:

> This is a tough one, and IDL's blessing/curse of a single namespace is the
> problem.

I understand the curse of a single namespace, what are the blessings? This is actually a serious question. I can see no advantages to a user that we are stuck with a single namespace. Is there any remote hope of this in the future?

Cheers,
Matt

--
Matthew Savoie - Senior Software Developer
National Snow and Ice Data Center
(303) 735-0785 http://nsidc.org

Subject: Re: IDL 8.2.2 released
Posted by David Fanning on Thu, 07 Feb 2013 16:21:08 GMT
View Forum Message <> Reply to Message

Mark Piper writes:

> I like this idea. From now on, three weeks before a release (when we're certain what's going into it), I'll publish a list of new routines. This should give some advance time for library maintainers to check their routines. Would this work?

Absolutely! Thank you. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Thu, 07 Feb 2013 16:55:39 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 9:20:55 AM UTC-7, Matt wrote:
>
> I understand the curse of a single namespace, what are the blessings? This
> is actually a serious question. I can see no advantages to a user that we
> are stuck with a single namespace. Is there any remote hope of this in the
> future?
>

Hi Matt,

I'll quote Wayne Landsman (from a thread I linked to above):

"IDL is often used for quick and dirty analyses, and for these cases a function or variable that is easy to type has its virtues."

though this may be less of an issue with command completion in the Workbench and IDLWAVE.

One of our developers has adapted the concept of a package for IDL, and it's backward compatible. It's still in the skunkworks, but I want to let people know that we're thinking about this.

mp

Subject: Re: IDL 8.2.2 released
Posted by chris_torrence@NOSPAM on Thu, 07 Feb 2013 17:11:10 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 9:20:55 AM UTC-7, Matt wrote:
> Mark Piper writes:
>
>
>
>> This is a tough one, and IDL's blessing/curse of a single namespace is the
>
>> problem.
>
>
>
> I understand the curse of a single namespace, what are the blessings? This
>
> is actually a serious question. I can see no advantages to a user that we
>
> are stuck with a single namespace. Is there any remote hope of this in the
>

> future?
>
>
>
>
>
> Cheers,
>
> Matt
>
>
>
> --
>
> Matthew Savoie - Senior Software Developer
>
> National Snow and Ice Data Center
>
> (303) 735-0785 http://nsidc.org

Hi Matt,

That is a great question. IDL's path management was based off of a Unix/Vax operating environment. If an executable is on your path, then your call will succeed. If it's not on your path, then your call will fail. If there are two copies on the path, then the first one wins.

This is simple, easy to understand and explain, and works well, up to a point!

IDL was designed and marketed as a "complete" solution, where you get everything in the box. There have been very few "add-on" packages where you might need to worry about namespaces and libraries. As Mark pointed out, we encouraged outside developers to use a prefix when creating their libraries, like David has done with his "cg" routines. That is an excellent approach.

Perhaps IDL has reached the threshold where we need to start worrying about namespaces, libraries, and packages. But if we do add support for that, we need to do it in a way that is backwards compatible, and is actually useful for a majority of our customers (both current and future users).

Hope this helps.
Cheers,
Chris
IDL Project Lead
ExelisVIS

Subject: Re: IDL 8.2.2 released
Posted by Bob[4] on Thu, 07 Feb 2013 18:53:16 GMT

On Wednesday, February 6, 2013 2:23:04 PM UTC-7, bobnn...@gmail.com wrote:
> I wonder if the very long existing bug where INTERPOLATE internally truncates all results to single precision even when double precision input is used is still present (note that it still returns the results in a double precision variable but half the precision is gone). It is in 8.1 so I have little hope it is gone in 8.2.2.
>
>
>
> Am I the only one that has been burned by this?

Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!

```
;; File interpolate_test.pro

x = dindgen(6)
y = dindgen(6)
x_i = dindgen(5) + 0.1d

print, 'INTERPOLATE: incorrect answer, accurate only to 8 digits'
print,interpolate(y,x_i),FORMAT='(G20.16)'

print, 'INTERPOL: correct answer, accurate to 16 digits'
print,interpol(y,x,x_i),FORMAT='(G20.16)'

end

IDL> .run interolate_test.pro
% Compiled module: $MAIN$.
INTERPOLATE: incorrect answer, accurate only to 8 digits
  0.1000000014901161
   1.100000023841858
   2.099999904632568
   3.099999904632568
   4.099999904632568
INTERPOL: correct answer, accurate to 16 digits
  0.1000000000000000
   1.100000000000000
   2.100000000000000
   3.100000000000000
   4.100000000000000
```

Subject: Re: IDL 8.2.2 released

Posted by David Fanning on Thu, 07 Feb 2013 19:15:48 GMT
View Forum Message <> Reply to Message

bobnnamtrop@gmail.com writes:

> Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!

This fix seems pretty straightforward. Are you *sure* you sent it to
Exelis? Did you send this kind of example, so they knew what you were
talking about? I think even I could probably wander through the C code
and fix something like this. ;-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: IDL 8.2.2 released
Posted by Bob[4] on Thu, 07 Feb 2013 20:14:29 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 12:15:48 PM UTC-7, David Fanning wrote:
> bobnn...@gmail.com writes:
>
>
>
>> Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!
>
>
>
> This fix seems pretty straightforward. Are you *sure* you sent it to
>
> Exelis? Did you send this kind of example, so they knew what you were
>

> talking about? I think even I could probably wander through the C code
>
> and fix something like this. ;-)
>
>
>
> Cheers,
>
>
>
> David

Yes, I agree, it should be very easy to fix. I sent in a bug report (with the example) on this twice, once in 2004 and then in 2011. It was acknowledged as a problem but never fixed.

Bob

Subject: Re: IDL 8.2.2 released
Posted by dain.cilke on Thu, 07 Feb 2013 22:07:45 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 11:53:16 AM UTC-7, bobnn...@gmail.com wrote:
> On Wednesday, February 6, 2013 2:23:04 PM UTC-7, bobnn...@gmail.com wrote:
>
>> I wonder if the very long existing bug where INTERPOLATE internally truncates all results to single precision even when double precision input is used is still present (note that it still returns the results in a double precision variable but half the precision is gone). It is in 8.1 so I have little hope it is gone in 8.2.2.
>
>>
>
>>
>
>>
>
>> Am I the only one that has been burned by this?
>
>
>
> Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!
>
>
>
> ;; File interpolate_test.pro

```
>
>
>
>  x = dindgen(6)
>
>  y = dindgen(6)
>
>  x_i = dindgen(5) + 0.1d
>
>
>
>  print, 'INTERPOLATE: incorrect answer, accurate only to 8 digits'
>
>  print,interpolate(y,x_i),FORMAT='(G20.16)'
>
>
>
>  print, 'INTERPOL: correct answer, accurate to 16 digits'
>
>  print,interpol(y,x,x_i),FORMAT='(G20.16)'
>
>
>
>  end
>
>
>
>  IDL> .run interolate_test.pro
>
>  % Compiled module: $MAIN$.
>
>  INTERPOLATE: incorrect answer, accurate only to 8 digits
>
>    0.10000000014901161
>
>     1.100000023841858
>
>     2.099999904632568
>
>     3.099999904632568
>
>     4.099999904632568
>
>  INTERPOL: correct answer, accurate to 16 digits
>
>    0.10000000000000000
>
>     1.100000000000000
```

>
>   2.100000000000000
>
>   3.100000000000000
>
>   4.100000000000000

Thanks for bring this to our attention! This bug has been fixed. Now interpolate will perform all calculations in double precision.

Subject: Re: IDL 8.2.2 released
Posted by dain.cilke on Thu, 07 Feb 2013 22:10:32 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 11:53:16 AM UTC-7, bobnn...@gmail.com wrote:
> On Wednesday, February 6, 2013 2:23:04 PM UTC-7, bobnn...@gmail.com wrote:
>
>> I wonder if the very long existing bug where INTERPOLATE internally truncates all results to single precision even when double precision input is used is still present (note that it still returns the results in a double precision variable but half the precision is gone). It is in 8.1 so I have little hope it is gone in 8.2.2.
>
>>
>
>>
>
>>
>
>> Am I the only one that has been burned by this?
>
>
>
> Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!
>
>
>
> ;; File interpolate_test.pro
>
>
>
> x = dindgen(6)
>
> y = dindgen(6)
>

```
> x_i = dindgen(5) + 0.1d
>
>
>
> print, 'INTERPOLATE: incorrect answer, accurate only to 8 digits'
>
> print,interpolate(y,x_i),FORMAT='(G20.16)'
>
>
>
> print, 'INTERPOL: correct answer, accurate to 16 digits'
>
> print,interpol(y,x,x_i),FORMAT='(G20.16)'
>
>
>
> end
>
>
>
> IDL> .run interolate_test.pro
>
> % Compiled module: $MAIN$.
>
> INTERPOLATE: incorrect answer, accurate only to 8 digits
>
>   0.1000000014901161
>
>    1.100000023841858
>
>    2.099999904632568
>
>    3.099999904632568
>
>    4.099999904632568
>
> INTERPOL: correct answer, accurate to 16 digits
>
>   0.1000000000000000
>
>    1.100000000000000
>
>    2.100000000000000
>
>    3.100000000000000
>
>    4.100000000000000
```

Hi Bob,
Thanks for bring this to our attention! This bug has been fixed. Now interpolate will perform all calculations in double precision.

Cheers,
Dain
IDL Code Monkey
ExelisVIS

Subject: Re: IDL 8.2.2 released
Posted by Bob[4] on Thu, 07 Feb 2013 22:55:14 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 3:10:32 PM UTC-7, dain....@gmail.com wrote:
> On Thursday, February 7, 2013 11:53:16 AM UTC-7, bobnn...@gmail.com wrote:
>
>> On Wednesday, February 6, 2013 2:23:04 PM UTC-7, bobnn...@gmail.com wrote:
>
>>
>
>>> I wonder if the very long existing bug where INTERPOLATE internally truncates all results to single precision even when double precision input is used is still present (note that it still returns the results in a double precision variable but half the precision is gone). It is in 8.1 so I have little hope it is gone in 8.2.2.
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> Am I the only one that has been burned by this?
>
>>
>
>>
>
>>
>
>>
>

>> Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!
>
>>
>
>>
>
>>
>
>> ;; File interpolate_test.pro
>
>>
>
>>
>
>>
>
>> x = dindgen(6)
>
>>
>
>> y = dindgen(6)
>
>>
>
>> x_i = dindgen(5) + 0.1d
>
>>
>
>>
>
>>
>
>> print, 'INTERPOLATE: incorrect answer, accurate only to 8 digits'
>
>>
>
>> print,interpolate(y,x_i),FORMAT='(G20.16)'
>
>>
>
>>
>
>>
>
>> print, 'INTERPOL: correct answer, accurate to 16 digits'

>
>>
>
>> print,interpol(y,x,x_i),FORMAT='(G20.16)'
>
>>
>
>>
>
>>
>
>> end
>
>>
>
>>
>
>>
>
>> IDL> .run interolate_test.pro
>
>>
>
>> % Compiled module: $MAIN$.
>
>>
>
>> INTERPOLATE: incorrect answer, accurate only to 8 digits
>
>>
>
>> 0.1000000014901161
>
>>
>
>> 1.100000023841858
>
>>
>
>> 2.099999904632568
>
>>
>
>> 3.099999904632568
>
>>
>
>> 4.099999904632568

>
>>
>
>> INTERPOL: correct answer, accurate to 16 digits
>
>>
>
>>   0.1000000000000000
>
>>
>
>>   1.100000000000000
>
>>
>
>>   2.100000000000000
>
>>
>
>>   3.100000000000000
>
>>
>
>>   4.100000000000000
>
>
>
> Hi Bob,
>
> Thanks for bring this to our attention! This bug has been fixed. Now interpolate will perform all calculations in double precision.
>
>
>
> Cheers,
>
> Dain
>
> IDL Code Monkey
>
> ExelisVIS

OK, that is great.

However, I assume that single precision inputs will still be done in single precision (or at least returned in single precision). I dislike routines that arbitrarily change the precision.

Bob

Subject: Re: IDL 8.2.2 released
Posted by chris_torrence@NOSPAM on Fri, 08 Feb 2013 16:14:53 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 9:47:13 PM UTC-7, timoth...@gmail.com wrote:
>> Also, if you have the CR number for a particular bug, I'd be happy to check on its status for you.
>
>>
>
>>
>
>>
>
>> mp
>
>
>
> Hi Mark,
>
>
>
> I would like to take you up on your offer. CRS 67311
>
> Its about the very slow restore times of large arrays from .sav files. We are using IDL in an emergency warning system where every second counts.
>
>
>
> Thanks,
>
> Tim

Hi Tim,

This bug is on the plate for IDL 8.3. I just did some quick tests, and it looks like some of the speed difference is because the IDL save files are saved in "big endian" format. When you read this on a Windows, Linux, or Mac machine, it needs to convert all of the arrays to little endian. Something we are doing in the C code must be slower than the Python big->little endian conversion.

In the meantime, if you are desperate for a speedup, you could use just an unstructured readu/writeu. It's not as flexible as the save file, but it will be as fast (if not faster) than the Python. On my Mac, using your original reproduce code, I could read that 400 Mb array in 6 seconds using a save file, and 0.6 seconds using a straight readu.

Just out of curiosity, are you using scipy's "idlsave" package to read IDL save files into Python?

-Chris
ExelisVIS

Subject: Re: IDL 8.2.2 released
Posted by dain.cilke on Fri, 08 Feb 2013 16:24:14 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 3:55:14 PM UTC-7, bobnn...@gmail.com wrote:
> On Thursday, February 7, 2013 3:10:32 PM UTC-7, dain....@gmail.com wrote:
>
>> On Thursday, February 7, 2013 11:53:16 AM UTC-7, bobnn...@gmail.com wrote:
>
>>
>
>>> On Wednesday, February 6, 2013 2:23:04 PM UTC-7, bobnn...@gmail.com wrote:
>
>>
>
>>>
>
>>>> I wonder if the very long existing bug where INTERPOLATE internally truncates all results to single precision even when double precision input is used is still present (note that it still returns the results in a double precision variable but half the precision is gone). It is in 8.1 so I have little hope it is gone in 8.2.2.
>
>>
>
>>>
>
>>>>
>
>>
>
>>>>
>
>>>
>
>>
>
>>>>
>

>
>>>
>
>>
>
>>>> Am I the only one that has been burned by this?
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> Well, I guess I am the only one who cares. But in case people just didn't know what I meant here is an example. Note that this is a toy example. The point is that if you have a function which you expect to work in double precision and you call interpolate then your results are truncated to single with no indication. I know of no other IDL built in function that does this!
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> ;; File interpolate_test.pro
>
>>
>
>>>
>
>>
>

```
>>>
>
>>
>
>>>
>
>>
>
>>>  x = dindgen(6)
>
>>
>
>>>
>
>>
>
>>>  y = dindgen(6)
>
>>
>
>>>
>
>>
>
>>>  x_i = dindgen(5) + 0.1d
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>  print, 'INTERPOLATE: incorrect answer, accurate only to 8 digits'
>
>>
>
>>>
>
>>
>
```

```
>>> print,interpolate(y,x_i),FORMAT='(G20.16)'
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> print, 'INTERPOL: correct answer, accurate to 16 digits'
>
>>
>
>>>
>
>>
>
>>> print,interpol(y,x,x_i),FORMAT='(G20.16)'
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> end
>
>>
>
>>>
>
>>
>
```

>>>
>
>>
>
>>>
>
>>
>
>>> IDL> .run interolate_test.pro
>
>>
>
>>>
>
>>
>
>>> % Compiled module: $MAIN$.
>
>>
>
>>>
>
>>
>
>>> INTERPOLATE: incorrect answer, accurate only to 8 digits
>
>>
>
>>>
>
>>
>
>>> 0.10000000014901161
>
>>
>
>>>
>
>>
>
>>> 1.100000023841858
>
>>
>
>>>
>
>>
>

>>>   2.099999904632568
>
>>
>
>>>
>
>>
>
>>>   3.099999904632568
>
>>
>
>>>
>
>>
>
>>>   4.099999904632568
>
>>
>
>>>
>
>>
>
>>> INTERPOL: correct answer, accurate to 16 digits
>
>>
>
>>>
>
>>
>
>>>   0.10000000000000000
>
>>
>
>>>
>
>>
>
>>>   1.100000000000000
>
>>
>
>>>
>
>>
>

>>>   2.100000000000000
>
>>
>
>>>
>
>>
>
>>>   3.100000000000000
>
>>
>
>>>
>
>>
>
>>>   4.100000000000000
>
>>
>
>>
>
>>
>
>> Hi Bob,
>
>>
>
>> Thanks for bring this to our attention! This bug has been fixed. Now interpolate will perform all calculations in double precision.
>
>>
>
>>
>
>>
>
>> Cheers,
>
>>
>
>> Dain
>
>>
>
>> IDL Code Monkey
>
>>

>
>> ExelisVIS
>
>
>
> OK, that is great.
>
>
>
> However, I assume that single precision inputs will still be done in single precision (or at least returned in single precision). I dislike routines that arbitrarily change the precision.
>
>
>
> Bob

Hi Bob,

Single precision inputs will result in single precision outputs. However, the internals of the function will be in double precision.

Hope this helps,
Dain

Subject: Re: IDL 8.2.2 released
Posted by Fabzi on Fri, 08 Feb 2013 17:04:03 GMT
View Forum Message <> Reply to Message

On 02/08/2013 05:14 PM, Chris Torrence wrote:
> In the meantime, if you are desperate for a speedup,
> you could use just an unstructured
> readu/writeu. It's not as flexible as the save file,
> but it will be as fast (if not faster)
> than the Python.

I could also recommend using NetCDF 4 files, which are quite flexible
and really fast to write/read. I made a small workbench test to write
and read a [500,500,500] array. Funny enough, both file have same size
on disk (477.8 Mb) but the IN/OUT differences are striking:

% Time elapsed IDL save: 8.8444729 seconds.
% Time elapsed NCDF save: 0.57288098 seconds.
% Time elapsed IDL restore: 8.8600481 seconds.
% Time elapsed NCDF restore: 0.63916397 seconds.

Fab

Subject: Re: IDL 8.2.2 released
Posted by Fabzi on Fri, 08 Feb 2013 17:15:49 GMT
View Forum Message <> Reply to Message

On 02/08/2013 06:04 PM, Fab wrote:
> I could also recommend using NetCDF 4 files, which are quite flexible

I forgot to mention that NetCDF files become really flexible when using
david fannings NCDF_FILE object. I could easily imagine a simple wrapper
for save that could go like this:
nsave, data1, data2, ..., datai, FILENAME='sav.nc'

as long as the arguments are arrays and not structures or so. Restore
would of course a bit different, but also cool.

Subject: Re: IDL 8.2.2 released
Posted by Matt Haffner on Fri, 08 Feb 2013 21:48:37 GMT
View Forum Message <> Reply to Message

It's trivial to get around, but the Mac installer package (as of a few minutes ago download) triggers the Gatekeeper "you shall not install!" dialog. Might be confusing/scary for novice users...

Also, re:packages or namespaces... that would be a great addition. +1 from me! TCL might be a good model for a language that also needed to support lots of legacy single-namespace code while still implementing something useful for growing, future development.

- Matt

On Tuesday, February 5, 2013 11:39:54 AM UTC-6, Mark Piper wrote:
> I'm happy to announce that IDL 8.2.2 is available for download from our website, www.exelisvis.com. I have a short write-up of what's new on the IDL blog:
>
>
>
> http://idldatapoint.com/2013/02/05/idl-8-2-2-released/
>
>
>
> including a link to download a set of example programs.
>
>
>
> mp

Subject: Re: IDL 8.2.2 released

Posted by timothyja123 on Sat, 09 Feb 2013 10:44:01 GMT
View Forum Message <> Reply to Message

> This bug is on the plate for IDL 8.3.

This is great to hear.

> I just did some quick tests, and it looks like some of the speed difference is because the IDL save files are saved in "big endian" format. When you read this on a Windows, Linux, or Mac machine, it needs to convert all of the arrays to little endian. Something we are doing in the C code must be slower than the Python big->little endian conversion.

The thing I noticed was that each array was restored one at a time, as I'm running a 6 core Xeon processor I guess I was hoping that this would become multithreaed so that multiple arrays would be restored in parallel although I guess I'm making assumptions about the inner-workings of IDL.

> Just out of curiosity, are you using scipy's "idlsave" package to read IDL save files into Python?

Yeah I used the original version of the idlsave package (which scipy added to their library)

Thanks to everyone for the suggestions on using a different format to speed things up. I'm not sure if this is an option right now as the data is coming from a different part of the orginisation and its up to them to change the format but is definitely something to consider.

Tim

Subject: Re: IDL 8.2.2 released
Posted by timothyja123 on Wed, 13 Feb 2013 00:43:39 GMT
View Forum Message <> Reply to Message

On Saturday, February 9, 2013 4:04:03 AM UTC+11, Fab wrote:
> On 02/08/2013 05:14 PM, Chris Torrence wrote:
>
>> In the meantime, if you are desperate for a speedup,
>
>> you could use just an unstructured
>
>> readu/writeu. It's not as flexible as the save file,
>
>> but it will be as fast (if not faster)
>
>> than the Python.
>
>
>
> I could also recommend using NetCDF 4 files, which are quite flexible
>

> and really fast to write/read. I made a small workbench test to write
>
> and read a [500,500,500] array. Funny enough, both file have same size
>
> on disk (477.8 Mb) but the IN/OUT differences are striking:
>
>
>
> % Time elapsed IDL save: 8.8444729 seconds.
>
> % Time elapsed NCDF save: 0.57288098 seconds.
>
> % Time elapsed IDL restore: 8.8600481 seconds.
>
> % Time elapsed NCDF restore: 0.63916397 seconds.
>
>
>
> Fab

I have been looking into your NetCDF suggestion. I thought I would start by converting one sav file on giving it a test. I have hit a problem however in that the IDL library does not seem to have an option to compress the NetCDF files. To give you an idea of what I'm working with my compressed IDL files are already 15GB so I need to be able to compress them. I'm I missing something? Is there a way to compress that I'm not seeing?

The NetCDF website says "NetCDF-4 only allows users to create data with the zlib library (due to licensing restrictions on the szlib library)" so I assume it should be supported.
Also the IDL documentation says "In NetCDF 4 files, data is created and accessed with the HDF5 library." The h5d_create function has a gzip param but the NCDF_CREATE does not.

Subject: Re: IDL 8.2.2 released
Posted by Fabzi on Wed, 13 Feb 2013 10:04:35 GMT
View Forum Message <> Reply to Message

Hi Timothy

On 02/13/2013 01:43 AM, timothyja123@gmail.com wrote:
> I have been looking into your NetCDF suggestion.
>
> I have hit a problem however in that the IDL library
> does not seem to have an option to compress the NetCDF files.

Which version of IDL are you using? With 7.1.1 (patched for netCDF) and higher this should be no problem. But you have to specify that you want to use netCDF 4 when creating the file, with the /NETCDF4_FORMAT keyword: http://www.exelisvis.com/docs/NCDF_CREATE.html

NetCDF4 allows you to create files larger than 2gb (upper limit for netCDF3 files) and also to use the three compression keywords when defining a variable (GZIP, CHUNCK_DIMENSIONS, SHUFFLE): http://www.exelisvis.com/docs/NCDF_VARDEF.html

NetCDF is very fast, but the more you compress, the slowier it will be though. I found almost no win in file size when going from GZIP=5 to GZIP=9 so I suggest to stick by 5. I also noticed that creating compressed files takes longer but that reading them is almost as fast as uncompressed ones, but this should be tested in a proper benchmark. The compression will be very efficient if your data is very recurrent (many zeros for example).

And again, I highly recommend David's NCDF_FILE object, which makes it much more fun to work with ncdf files and has no limitations (all keywords are accessible):
http://www.idlcoyote.com/programs/ncdf_file__define.pro
(the only drawback of David's tool is that it is "all objects" and therefore parsing a lot of files having a lot of variables/attributes is a bit slow)

Cheers,

Fab

Subject: Re: IDL 8.2.2 released
Posted by timothyja123 on Wed, 13 Feb 2013 22:52:46 GMT
View Forum Message <> Reply to Message

On Wednesday, February 13, 2013 9:04:35 PM UTC+11, Fab wrote:
> Hi Timothy
>
>
>
>
>> I have been looking into your NetCDF suggestion.
>
>>
>
>> I have hit a problem however in that the IDL library
>
>> does not seem to have an option to compress the NetCDF files.
>
>
>
> Which version of IDL are you using? With 7.1.1 (patched for netCDF) and

> higher this should be no problem. But you have to specify that you want
> to use netCDF 4 when creating the file, with the /NETCDF4_FORMAT
> keyword: http://www.exelisvis.com/docs/NCDF_CREATE.html
>
>
> NetCDF4 allows you to create files larger than 2gb (upper limit for
> netCDF3 files) and also to use the three compression keywords when
> defining a variable (GZIP, CHUNCK_DIMENSIONS, SHUFFLE):
> http://www.exelisvis.com/docs/NCDF_VARDEF.html
>
>
> NetCDF is very fast, but the more you compress, the slowier it will be
> though. I found almost no win in file size when going from GZIP=5 to
> GZIP=9 so I suggest to stick by 5. I also noticed that creating
> compressed files takes longer but that reading them is almost as fast as
> uncompressed ones, but this should be tested in a proper benchmark. The
> compression will be very efficient if your data is very recurrent (many
> zeros for example).
>
>
> And again, I highly recommend David's NCDF_FILE object, which makes it
> much more fun to work with ncdf files and has no limitations (all
> keywords are accessible):
> http://www.idlcoyote.com/programs/ncdf_file__define.pro
> (the only drawback of David's tool is that it is "all objects" and
> therefore parsing a lot of files having a lot of variables/attributes is
> a bit slow)

>
>
>
> Cheers,
>
>
>
> Fab

Hi Fab,

Yes this works I didn't realise at first that compressing was done on a per variable bases. Thanks you for pointing that out. The new files definitely load much faster and also seem to compress slightly more which is another plus.

Subject: Re: IDL 8.2.2 released
Posted by Michael Galloy on Thu, 14 Feb 2013 00:16:09 GMT
View Forum Message <> Reply to Message

On 2/5/13 10:39 AM, Mark Piper wrote:
> I'm happy to announce that IDL 8.2.2 is available for download from
> our website, www.exelisvis.com. I have a short write-up of what's new
> on the IDL blog:
>
> http://idldatapoint.com/2013/02/05/idl-8-2-2-released/
>
> including a link to download a set of example programs.

By the way, I have updated Modern IDL for IDL 8.2.2. If you have bought a PDF of Modern IDL in the past, you should have received an email with a link to download the update. Please let me know if you haven't received your update.

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)
Research Mathematician
Tech-X Corporation

Subject: Re: IDL 8.2.2 released
Posted by Michael Galloy on Thu, 14 Feb 2013 00:19:40 GMT
View Forum Message <> Reply to Message

On 2/8/13 2:48 PM, Matt Haffner wrote:
> It's trivial to get around, but the Mac installer package (as of a
> few minutes ago download) triggers the Gatekeeper "you shall not
> install!" dialog. Might be confusing/scary for novice users...

Just to be explicit, right click on the installer, select "Open", and
then OK the warning that appears.

I can't remember the timing for IDL 8.2.1, but I think it came out right
after Mountain Lion so I was willing to cut them some slack. But a
signed installer would be nice for future releases...

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)
Research Mathematician
Tech-X Corporation

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Thu, 14 Feb 2013 16:23:50 GMT
View Forum Message <> Reply to Message

On Wednesday, February 13, 2013 5:19:40 PM UTC-7, Mike Galloy wrote:
> On 2/8/13 2:48 PM, Matt Haffner wrote:
>
>> It's trivial to get around, but the Mac installer package (as of a
>> few minutes ago download) triggers the Gatekeeper "you shall not
>> install!" dialog. Might be confusing/scary for novice users...
>
> Just to be explicit, right click on the installer, select "Open", and
> then OK the warning that appears.
>

Thank you, Matt & Mike, for pointing this out; we're looking into it.

mp

Subject: Re: IDL 8.2.2 released
Posted by tom.grydeland on Wed, 20 Mar 2013 09:58:17 GMT
View Forum Message <> Reply to Message

On Thursday, February 7, 2013 12:09:57 AM UTC, Mark Piper wrote:
> I love the fact that people care enough about IDL (like I do) to post comments on this thread!

Please keep them coming. I'll try to respond to each comment in a critical and responsible manner.
>
>

Hello,

I heard that plotting speed had improved in 8.2.2 and decided to give it a go.

Building up plots incrementally is still painfully slow. The routine below creates an illustration of a phased array antenna where all elements are aligned radially or tangentially. On a reasonable computer, building the plot takes an unreasonable 44 seconds to complete. (linux x86_64 m64)

I am profiling the call, and getting some very interesting statistics:

Hit count	Time self(s)	Time+sub(s)	Sys	Routine
292577	1.0121676922	1.0121676922	1	IDLGRPLOT::GETPROPERTY
1311191	2.1339576244	2.1339576244	1	IDLITCOMPONENT::GETPROPERTY
116646	1.2261884212	10.1981632710	0	IDLITVISPLOT::GETXYZRANGE
560278	2.1338310242	2.1338310242	1	IDL_CONTAINER::GET
619133	0.7483308315	0.7483308315	1	MIN
297796	1.9745259285	3.3823873997	0	_IDLITCONTAINER::GET
259440	5.5053386688	11.2065567970	0	_IDLITCONTAINER::_GETIDENTIFIERS
256828	1.5982770920	1.9980626106	0	_IDLITVISUALIZATION::GETPROPERTY
118096	3.9485795498	14.2965276241	0	_IDLITVISUALIZATION::GETXYZRANGE
122931	1.0249559879	1.3674829006	0	_IDLITVISUALIZATION::SEEKPIXELATEDVISUALIZATION
233776	4.5677740574	5.8650910854	0	_IDLITVISUALIZATION::_ACCUMULATEXYZRANGE

So the time spent per routine call isn't bad, but the hit counts ...

No less than 26 routines are being called more than 90 thousand times each.

12 routines take up more than a second of accumulated time each, totalling over 28 seconds between them.

We're talking about 121 points and 240 lines here. Not exactly staggering.

> Thanks,
> mp
> IDL Product Manager
> mark.piper@exelisvis.com

Cheers,

--T

```
function radial_antenna, n, length=length

  if n_elements(length) eq 0 then length = 0.9

  ii = indgen(n)-(n-1.)/2
  xx = ii[*,ii]
  yy = transpose(xx)
  cg = plot(xx[*], yy[*], 'D', aspect_ratio=1)
  ;; cg.refresh, /disable

  nx = n_elements(xx)
  if nx mod 2 then begin
    ;; if there is an element in the centre, eliminate it
    tmp = [indgen(nx/2), nx/2+1+indgen(nx/2)]
    xx = xx[tmp]
    yy = yy[tmp]
  endif
  th = atan(yy, xx)

  for ii=0L, n_elements(xx)-1 do begin
    cx = length/2*[-1, 1]*cos(th[ii])
    cy = length/2*[-1, 1]*sin(th[ii])
    !null = plot(xx[ii]+cx, yy[ii]+cy, 'k', /current, /overplot)
    !null = plot(xx[ii]-cy, yy[ii]+cx, 'r', /current, /overplot)
  endfor

  cg.refresh
  return, cg
end


;; main routine
profiler, /reset
profiler, /system
profiler

cg = radial_antenna(11)

profiler, /report, filename='report.txt'

end
```

Subject: Re: IDL 8.2.2 released

Posted by lecacheux.alain on Wed, 20 Mar 2013 16:07:23 GMT
View Forum Message <> Reply to Message

Le mercredi 20 mars 2013 10:58:17 UTC+1, Tom Grydeland a écrit :
> On Thursday, February 7, 2013 12:09:57 AM UTC, Mark Piper wrote:
>
>> I love the fact that people care enough about IDL (like I do) to post comments on this thread! Please keep them coming. I'll try to respond to each comment in a critical and responsible manner.
>
>>
>
>>
>
>
>
> Hello,
>
>
>
> I heard that plotting speed had improved in 8.2.2 and decided to give it a go.
>
>
>
> Building up plots incrementally is still painfully slow. The routine below creates an illustration of a phased array antenna where all elements are aligned radially or tangentially. On a reasonable computer, building the plot takes an unreasonable 44 seconds to complete. (linux x86_64 m64)
>
>
>
> I am profiling the call, and getting some very interesting statistics:
>
>
>
> Hit count Time self(s) Time+sub(s) Sys Routine
>
> 292577 1.0121676922 1.0121676922 1 IDLGRPLOT::GETPROPERTY
>
> 1311191 2.1339576244 2.1339576244 1 IDLITCOMPONENT::GETPROPERTY
>
> 116646 1.2261884212 10.1981632710 0 IDLITVISPLOT::GETXYZRANGE
>
> 560278 2.1338310242 2.1338310242 1 IDL_CONTAINER::GET
>
> 619133 0.7483308315 0.7483308315 1 MIN
>
> 297796 1.9745259285 3.3823873997 0 _IDLITCONTAINER::GET
>
> 259440 5.5053386688 11.2065567970 0 _IDLITCONTAINER::_GETIDENTIFIERS

>
>   256828   1.5982770920   1.9980626106   0 _IDLITVISUALIZATION::GETPROPERTY
>
>   118096   3.9485795498   14.2965276241   0 _IDLITVISUALIZATION::GETXYZRANGE
>
>   122931   1.0249559879   1.3674829006   0
_IDLITVISUALIZATION::SEEKPIXELATEDVISUALIZATION
>
>   233776   4.5677740574   5.8650910854   0
_IDLITVISUALIZATION::_ACCUMULATEXYZRANGE
>
>
>
> So the time spent per routine call isn't bad, but the hit counts ...
>
>
>
> No less than 26 routines are being called more than 90 thousand times each.
>
>
>
> 12 routines take up more than a second of accumulated time each, totalling over 28 seconds
between them.
>
>
>
> We're talking about 121 points and 240 lines here. Not exactly staggering.
>
>
>
>> Thanks,
>
>> mp
>
>> IDL Product Manager
>
>> mark.piper@exelisvis.com
>
>
>
> Cheers,
>
>
>
> --T
>
>
>

```
>
>
>
>
>
>
> function radial_antenna, n, length=length
>
>
>
>   if n_elements(length) eq 0 then length = 0.9
>
>
>
>   ii = indgen(n)-(n-1.)/2
>
>   xx = ii[*,ii]
>
>   yy = transpose(xx)
>
>   cg = plot(xx[*], yy[*], 'D', aspect_ratio=1)
>
>   ;; cg.refresh, /disable
>
>
>
>   nx = n_elements(xx)
>
>   if nx mod 2 then begin
>
>     ;; if there is an element in the centre, eliminate it
>
>     tmp = [indgen(nx/2), nx/2+1+indgen(nx/2)]
>
>     xx = xx[tmp]
>
>     yy = yy[tmp]
>
>   endif
>
>   th = atan(yy, xx)
>
>
>
>   for ii=0L, n_elements(xx)-1 do begin
>
>     cx = length/2*[-1, 1]*cos(th[ii])
>
```

```
>      cy = length/2*[-1, 1]*sin(th[ii])
>
>      !null = plot(xx[ii]+cx, yy[ii]+cy, 'k', /current, /overplot)
>
>      !null = plot(xx[ii]-cy, yy[ii]+cx, 'r', /current, /overplot)
>
>    endfor
>
>
>
>    cg.refresh
>
>    return, cg
>
>  end
>
>
>
>
>
>  ;; main routine
>
>  profiler, /reset
>
>  profiler, /system
>
>  profiler
>
>
>
>  cg = radial_antenna(11)
>
>
>
>  profiler, /report, filename='report.txt'
>
>
>
>  end
```

You should not call the NG plot function in a loop. The POLYLINE function with using CONNECTIVITY keyword is much better in your case.
For instance, replacing your:

```
  for ii=0L, n_elements(xx)-1 do begin
    cx = length/2*[-1, 1]*cos(th[ii])
```

```
    cy = length/2*[-1, 1]*sin(th[ii])
    !null = plot(xx[ii]+cx, yy[ii]+cy, 'k', /current, /overplot)
    !null = plot(xx[ii]-cy, yy[ii]+cx, 'r', /current, /overplot)
  endfor
```

by :

```
  cx = fltarr(2,n_elements(xx)) & cy = cx
  conn = lonarr(3,n_elements(xx))
  for ii=0L, n_elements(xx)-1 do begin
    cx[*,ii] = length/2*[-1, 1]*cos(th[ii])
    cy[*,ii] = length/2*[-1, 1]*sin(th[ii])
    conn[*,ii] = [2,2*ii,2*ii+1]
  endfor
  nx2 = 2*N_elements(xx)
  tic
  !null = polyline(reform([1,1]#xx + cx, nx2), reform([1,1]#yy + cy, nx2), 'k', CONNECTIVITY=conn,
TARGET=cg, /DATA)
  !null = polyline(reform([1,1]#xx - cy, nx2), reform([1,1]#yy + cx, nx2), 'r', CONNECTIVITY=conn,
TARGET=cg, /DATA)
  toc
```

the execution time goes from 36.1 sec. on my machine down to 0.031 sec.
A factor 1000 which has to be explained by Exelis...

alain.

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Wed, 20 Mar 2013 21:10:08 GMT
View Forum Message <> Reply to Message

On Wednesday, February 13, 2013 5:19:40 PM UTC-7, Mike Galloy wrote:
>
> I can't remember the timing for IDL 8.2.1, but I think it came out right
> after Mountain Lion so I was willing to cut them some slack. But a
> signed installer would be nice for future releases...
>

We now have a Developer ID certificate from Apple, so IDL 8.2.3 will be a signed application and you shouldn't run into Gatekeeper issues on install.

mp

Subject: Re: IDL 8.2.2 released

Posted by tom.grydeland on Thu, 21 Mar 2013 09:38:08 GMT
View Forum Message <> Reply to Message

> Le mercredi 20 mars 2013 10:58:17 UTC+1, Tom Grydeland a écrit :
>> Building up plots incrementally is still painfully slow.

On Wednesday, March 20, 2013 4:07:23 PM UTC, alx wrote:
> You should not call the NG plot function in a loop. The POLYLINE function with using CONNECTIVITY keyword is much better in your case.

Sure. Myself, I used PLOT with NaNs as every third value. That also works.

> the execution time goes from 36.1 sec. on my machine down to 0.031 sec.
> A factor 1000 which has to be explained by Exelis...

This is what I'm getting at. Your workaround, and mine as well, forces all the lines drawn in one command to have the same properties, and makes it impossible to hold on to separate handles for them, should that be of interest. Say you wanted the ability to draw lines in individual colors, or to selectively highlight one of the lines at a later point.

Saying "don't call NG routines in a loop" is useful practical advice, but unsatisfactory. Varying N in my example demonstrates quadratic increase in time with N, so it appears that all existing graphic elements are queried (e.g. for XYZ boundaries) whenever a new element is added. Surely there is an object in the graphics bestiary which could be responsible for remembering and updating the X/Y/Z extrema instead of having to recompute them on every operation? For extra points, identify all other instances where every element in a graphic is being queried.

> alain.

--T

Subject: Re: IDL 8.2.2 released
Posted by Mark Piper on Thu, 21 Mar 2013 16:41:34 GMT
View Forum Message <> Reply to Message

On Thursday, March 21, 2013 3:38:08 AM UTC-6, Tom Grydeland wrote:
>
> Saying "don't call NG routines in a loop" is useful practical advice, but unsatisfactory. Varying N in my example demonstrates quadratic increase in time with N, so it appears that all existing graphic elements are queried (e.g. for XYZ boundaries) whenever a new element is added. Surely there is an object in the graphics bestiary which could be responsible for remembering and updating the X/Y/Z extrema instead of having to recompute them on every operation? For extra points, identify all other instances where every element in a graphic is being queried.
>

This is a problem that needs to be solved. I'll discuss it with Chris.

mp

Subject: Re: IDL 8.2.2 released
Posted by chris_torrence@NOSPAM on Tue, 16 Apr 2013 04:08:24 GMT
View Forum Message <> Reply to Message

On Thursday, March 21, 2013 10:41:34 AM UTC-6, Mark Piper wrote:
> On Thursday, March 21, 2013 3:38:08 AM UTC-6, Tom Grydeland wrote:
>
>>
>
>> Saying "don't call NG routines in a loop" is useful practical advice, but unsatisfactory. Varying N in my example demonstrates quadratic increase in time with N, so it appears that all existing graphic elements are queried (e.g. for XYZ boundaries) whenever a new element is added. Surely there is an object in the graphics bestiary which could be responsible for remembering and updating the X/Y/Z extrema instead of having to recompute them on every operation? For extra points, identify all other instances where every element in a graphic is being queried.
>
>>
>
>
>
> This is a problem that needs to be solved. I'll discuss it with Chris.
>
>
>
> mp

Hi Tom et al.,

This has been fixed for IDL 8.2.3. As long as you disable refresh (the commented out line of code in your example) then the time to add new plots is the same regardless of the # of plots. In your particular example, once I added back in the Refresh,/disable line, the time for radial_antenna(12) went from 40 seconds down to around 4 seconds.

Note that if you don't disable the refresh, then each time a new plot is added, IDL will recompute the plot range to make sure the axes cover all of the plots.

Cheers,

Chris
ExelisVIS

Subject: Re: IDL 8.2.2 released
Posted by tushar on Fri, 26 Apr 2013 06:31:08 GMT

View Forum Message <> Reply to Message

hey is their any link to download this setup of idl 8.2.2 I am not getting it how to download the setup and for that do i have to make an account in the site ??

Subject: Re: IDL 8.2.2 released
Posted by tom.grydeland on Fri, 26 Apr 2013 10:06:50 GMT
View Forum Message <> Reply to Message

On Tuesday, April 16, 2013 4:08:24 AM UTC, Chris Torrence wrote:

> This has been fixed for IDL 8.2.3.
> [...]

> Note that if you don't disable the refresh, then each time a new plot is added, IDL will recompute the plot range to make sure the axes cover all of the plots.

Obviously. What I tried to say was that the quadratic behaviour suggests to me that this recomputation queries _all_ existing objects in the plot, something which is quite unnecessary. The plot range necessary for all _previously_ added objects does not change as a result of adding another one.

This is quite orthogonal to whether refresh is disabled or not.

((In pseudocode of an AXIS object of some description, assuming BOX will compute the union of two bounding-box arguments, and REDUCE works as one would expect, this can be expressed as

```
self.bbox = BOX(self.bbox, newOjb.bbox)   ; bbox kept in property of AXIS
```

instead of

```
bbox = REDUCE('BOX', self.objects)        ; bbox recomputed and not kept
```

))

> the time for radial_antenna(12) went from 40 seconds down to around 4 seconds.

Obviously a great improvement, but I cannot shake a feeling that there must be at least another factor of 10 within reach.

For comparison, on our old compute-server (Dual-core AMD Opteron, 2.6 GHz, 5200 bogomips) and with X over the network, an old version of That Other Vectorized Language (r2008a) does radial_antenna with time per antenna of 0.75 ms consistently for n from 10 to 30 (100 to 900 antennas in the pattern).

On my desktop (i5, 3.2 GHz, 6400 bogomips), the time per antenna is 70 ms for n=6, 130 ms for

n=10, 234 ms for n=14, and that is as far as my patience extends for now.

If you run the instrumented code below, the final plot should show a roughly constant value, not one that increases quadratically. Does it in 8.2.3?

```
function radial_antenna, n, length=length

  if n_elements(length) eq 0 then length = 0.9

  ii = indgen(n)-(n-1.)/2
  xx = ii[*,ii]
  yy = transpose(xx)
  cg = plot(xx[*], yy[*], 'D', aspect_ratio=1)
  cg.refresh, /disable

  nx = n_elements(xx)
  if nx mod 2 then begin
    ;; if there is an element in the centre, eliminate it
    tmp = [indgen(nx/2), nx/2+1+indgen(nx/2)]
    xx = xx[tmp]
    yy = yy[tmp]
  endif
  th = atan(yy, xx)

  tic
  for ii=0L, n_elements(xx)-1 do begin
    cx = length/2*[-1, 1]*cos(th[ii])
    cy = length/2*[-1, 1]*sin(th[ii])
    !null = plot(xx[ii]+cx, yy[ii]+cy, 'k', /current, /overplot)
    !null = plot(xx[ii]-cy, yy[ii]+cx, 'r', /current, /overplot)
  endfor

  cg.refresh
  return, toc() / n^2
end


;; main routine

nn = 6 + 4*indgen(3)
tt = 0. * nn

foreach ii, nn, idx do tt[idx] = radial_antenna(ii)

!null = plot(nn, tt)


end
```

> Chris

--T

Subject: Re: IDL 8.2.2 released
Posted by chris_torrence@NOSPAM on Fri, 26 Apr 2013 21:33:07 GMT
View Forum Message <> Reply to Message

Hi Tom,

I slightly tweaked your code by adding a "cg.Close" right after the Refresh, so that it closed the window before doing the next antenna. I computed all of the antenna times between 6 and 14, and got the following results:

n	time/n^2
6	0.0458333
7	0.0473469
8	0.0465000
9	0.0475802
10	0.0498900
11	0.0508099
12	0.0507153
13	0.0508225
14	0.0529898

So, with IDL 8.2.3, the times are no longer quadratic, which is good, but they are still showing a slight linear trend.

I'll keep looking at the issue, and let you know what I find. Thanks for all of the code. That really helps!

-Chris
ExelisVIS
