
Subject: Plot bug or another "strange feature" ?

Posted by [Nicolas Decoster](#) on Tue, 16 May 2000 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi.

Does anybody notice and, eventually, know how to handle this "strange feature" of the plot procedure ?

```
IDL> s = indgen(1000) + 10000000000
```

```
IDL> plot, s, /ynozero
```

The plot is not a clean line joining down-left and up-right corners, but an ugly stairway...

Perhaps I missed the holy keyword...

Nicolas.

--

Téléphone : 00 (33) 5 62 88 11 16

Fax : 00 (33) 5 62 88 11 12

Nicolas.Decoster@Noveltis.fr

Noveltis

Parc Technologique du Canal

2, avenue de l'Europe

31520 Ramonville Saint Agne - France

Subject: Re: Plot bug or another "strange feature" ?

Posted by [Paul van Delst](#) on Tue, 16 May 2000 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Nicolas Decoster wrote:

>

> Paul van Delst wrote:

>>

>> Nicolas Decoster wrote:

>>>

>>> Hi.

>>>

>>> Does anybody notice and, eventually, know how to handle this "strange

>>> feature" of the plot procedure ?

>>>

>>> IDL> s = indgen(1000) + 10000000000

>>> IDL> plot, s, /ynozero

>>>

```

>>> The plot is not a clean line joining down-left and up-right corners, but
>>> an ugly stairway...
>>
>> I thought that sort of thing came about because:
>>
>> a) PLOT converts all its arguments to single precision floats. Do a
>> PRINT, FLOAT(s[0:100]),FORMAT='(e20.13)' to see what happens to the
>> float'd integers,
>>
>> b) people think they can represent numbers at the extremes of machine
>> precision exactly. Your "s" array, when converted to single precision
>> float, changes in the 8th or 9th d.p. While I think that IDL should
>> allow users to set the PLOT conversion to double precision if they want
>> (e.g. with a DOUBLE keyword or something), in general you can't expect
>> these sorts of numbers to be represented well - in IDL or any language.
>> That's not how floating point arithmetic works. All floating point
>> numbers are approximations to their actual value.
>
> I mainly agree with you. But "s" array is not a double array, it is a
> long array:

```

It doesn't matter what "s" is originally. When you PLOT it, at some point the contents are "converted" to *single precision* floating point numbers.

```

> I agree with you that plot converts everything to floating point number.
> But I think that it is important to see exactly what are the values of
> my integers (in fact my long integers) in a graph. And it is the same
> with double precision floating point numbers: I am not talking about the
> representation of real world numbers using floating point numbers.
> Example:
>
> IDL> b = [1000000000.00000001d, 1000000000.00000003d]
> IDL> print, b, format = '(e22.16)'
> 1.000000000000000001e+08
> 1.000000000000000003e+08
> IDL> plot, b, /ynozero
> % PLOT: Data range for axis has zero length.
> % Execution halted at: $MAIN$

```

Well now I'm confused because your example above deals *explicitly* with the representation (or the problems therein) of real world numbers using floating point arithmetic. By using PLOT, "b" effectively becomes an array of the same two numbers. Personally, I would like IDL's PLOT to "convert" everything to double precision but maybe that raises resource issues when you want to plot x/y arrays of, say, 2^{19} point - which I do all the time. What IDL does satisfies all but a miniscule percentage of scenarios. To me that reflects good design philosophy (you know,

diminishing returns and all that).

Taking your example above, I would prefer to plot the data (and similar) like so:

```
IDL> b = [100000000.00000001d, 100000000.00000003d]
IDL> print, b, format = '(e22.16)'
1.0000000000000001e+08
1.0000000000000003e+08
IDL> plot, b-(total(b)/double(n_elements(b))),
/ynozero,ytitle='y-ave(y)'
```

or even plot the percentage change or something. Anything than the actual magnitudes.

I think I understand what you mean and what you want - my point is (and I'm sure you already know this) that when you want to plot/print/whatever real numbers that reside near the boundaries of the ability of various architectures to represent them, you're going to run into problems as we have seen above. It's not a particularly IDL thing, you just have to be careful.

paulv

--

Paul van Delst Ph: (301) 763-8000 x7274
CIMSS @ NOAA/NCEP Fax: (301) 763-8545
Rm.202, 5200 Auth Rd. Email: pvandelst@ncep.noaa.gov
Camp Springs MD 20746

Subject: Re: Plot bug or another "strange feature" ?
Posted by [Paul van Delst](#) on Tue, 16 May 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

"J.D. Smith" wrote:

>
> Such a plot routine could easily be written by a motivated user. Consider that
> in order to generate drawing commands for typical display and hardcopy devices,
> IDL must convert your data to floating point values at some point in the
> process. You may disagree with where and how they perform this conversion, and
> therefore can write a general purpose plot wrapper which auto-detects the type
> of data, scales or offsets it into a range well matched to the internal plot's
> floating precision, and then undoes the scaling in the labelling of the tick
> values.

I didn't know we were talking about plotting using object graphics....
:o)

> It seems as if your argument is equivalent to complaining that your image value
> which has data ranging from 1.e-12 to 2.e-12 doesn't display correctly with a
> straight "tv" command. RSI *could* have developed some overly-generalized
> notions of when and how to scale our data, which would work sometimes and
> sometimes not, and when it didn't, it would take heavy magic and carnal
> knowledge to get what you want. Thankfully they didn't.

That's a good illustration of the point. In this context the errors generated by the examples given in this thread are tru blu features!

paulv

--

Paul van Delst Ph: (301) 763-8000 x7274
CIMSS @ NOAA/NCEP Fax: (301) 763-8545
Rm.202, 5200 Auth Rd. Email: pvandelst@ncep.noaa.gov
Camp Springs MD 20746

Subject: Re: Plot bug or another "strange feature" ?
Posted by [davidf](#) on Tue, 16 May 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

J.D. Smith (jdsmith@astro.cornell.edu) writes:

> RSI *could* have developed some overly-generalized
> notions of when and how to scale our data, which would work sometimes and
> sometimes not, and when it didn't, it would take heavy magic and carnal
> knowledge to get what you want. Thankfully they didn't.

Oh, really, JD! For a discussion group focused on spiritual matters today, I hardly think a post like this is appropriate. "Carnal knowledge", indeed! :-(

Cheers,

David

--

David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: <http://www.dfanning.com/>
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: Plot bug or another "strange feature" ?
Posted by [John-David T. Smith](#) on Tue, 16 May 2000 07:00:00 GMT

Nicolas Decoster wrote:

>

> Paul van Delst wrote:

>>

>> Nicolas Decoster wrote:

>>>

>>> Hi.

>>>

>>> Does anybody notice and, eventually, know how to handle this "strange
>>> feature" of the plot procedure ?

>>>

>>> IDL> s = indgen(1000) + 1000000000

>>> IDL> plot, s, /ynozero

>>>

>>> The plot is not a clean line joining down-left and up-right corners, but
>>> an ugly stairway...

>>

>> I thought that sort of thing came about because:

>>

>> a) PLOT converts all its arguments to single precision floats. Do a
>> PRINT, FLOAT(s[0:100]),FORMAT='(e20.13)' to see what happens to the
>> float'd integers,

>>

>> b) people think they can represent numbers at the extremes of machine
>> precision exactly. Your "s" array, when converted to single precision
>> float, changes in the 8th or 9th d.p. While I think that IDL should
>> allow users to set the PLOT conversion to double precision if they want
>> (e.g. with a DOUBLE keyword or something), in general you can't expect
>> these sorts of numbers to be represented well - in IDL or any language.
>> That's not how floating point arithmetic works. All floating point
>> numbers are approximations to their actual value.

>

> I mainly agree with you. But "s" array is not a double array, it is a
> long array:

>

> IDL> help, s

> S LONG = Array[1000]

>

> I agree with you that plot converts everything to floating point number.

> But I think that it is important to see exactly what are the values of

> my integers (in fact my long integers) in a graph. And it is the same

> with double precision floating point numbers: I am not talking about the

> representation of real world numbers using floating point numbers.

> Example:

>

> IDL> b = [1000000000.00000001d, 1000000000.00000003d]

> IDL> print, b, format = '(e22.16)'

```
> 1.00000000000000001e+08
> 1.00000000000000003e+08
> IDL> plot, b, /ynozero
> % PLOT: Data range for axis has zero length.
> % Execution halted at: $MAIN$
>
> Yes, perhaps a keyword would be fine, here. Or better, a plot procedure
> that computes the data range according to the precision of the data.
```

Such a plot routine could easily be written by a motivated user. Consider that in order to generate drawing commands for typical display and hardcopy devices, IDL must convert your data to floating point values at some point in the process. You may disagree with where and how they perform this conversion, and therefore can write a general purpose plot wrapper which auto-detects the type of data, scales or offsets it into a range well matched to the internal plot's floating precision, and then undoes the scaling in the labelling of the tick values. This linear scaling, which could be saved, e.g., in a system variable, could then be applied to all subsequent suitably overridden plotting routines.

It seems as if your argument is equivalent to complaining that your image value which has data ranging from 1.e-12 to 2.e-12 doesn't display correctly with a straight "tv" command. RSI *could* have developed some overly-generalized notions of when and how to scale our data, which would work sometimes and sometimes not, and when it didn't, it would take heavy magic and carnal knowledge to get what you want. Thankfully they didn't.

JD

--

J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|

Subject: Re: Plot bug or another "strange feature" ?
Posted by [Nicolas Decoster](#) on Tue, 16 May 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Paul van Delst wrote:

```
>
> Nicolas Decoster wrote:
>>
>> Hi.
>>
>> Does anybody notice and, eventually, know how to handle this "strange
>> feature" of the plot procedure ?
>>
```

```
>> IDL> s = indgen(1000) + 1000000000
>> IDL> plot, s, /ynozero
>>
>> The plot is not a clean line joining down-left and up-right corners, but
>> an ugly stairway...
>
> I thought that sort of thing came about because:
>
> a) PLOT converts all its arguments to single precision floats. Do a
> PRINT, FLOAT(s[0:100]),FORMAT='(e20.13)' to see what happens to the
> float'd integers,
>
> b) people think they can represent numbers at the extremes of machine
> precision exactly. Your "s" array, when converted to single precision
> float, changes in the 8th or 9th d.p. While I think that IDL should
> allow users to set the PLOT conversion to double precision if they want
> (e.g. with a DOUBLE keyword or something), in general you can't expect
> these sorts of numbers to be represented well - in IDL or any language.
> That's not how floating point arithmetic works. All floating point
> numbers are approximations to their actual value.
```

I mainly agree with you. But "s" array is not a double array, it is a long array:

```
IDL> help, s
S          LONG      = Array[1000]
```

I agree with you that plot converts everything to floating point number. But I think that it is important to see exactly what are the values of my integers (in fact my long integers) in a graph. And it is the same with double precision floating point numbers: I am not talking about the representation of real world numbers using floating point numbers. Example:

```
IDL> b = [1000000000.00000001d, 1000000000.00000003d]
IDL> print, b, format = '(e22.16)'
1.000000000000000001e+08
1.000000000000000003e+08
IDL> plot, b, /ynozero
% PLOT: Data range for axis has zero length.
% Execution halted at: $MAIN$
```

Yes, perhaps a keyword would be fine, here. Or better, a plot procedure that computes the data range according to the precision of the data.

Later,

Nicolas.

--

Tél. : 00 (33) 5 62 88 11 16

Fax : 00 (33) 5 62 88 11 12

Nicolas.Decoster@Noveltis.fr

Noveltis

Parc Technologique du Canal

2, avenue de l'Europe

31520 Ramonville Saint Agne - France

Subject: Re: Plot bug or another "strange feature" ?

Posted by [davidf](#) on Tue, 16 May 2000 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Nicolas Decoster (Nicolas.Decoster@Noveltis.fr) writes:

- > It sounds like a good strategy to me. I'll try it. But before I shave my
- > head and wear the peach clothes, I have a last question:
- >
- > How can I have "no expectations" from a _commercial_ product ?

No one said a spiritual practice is easy. Only that it is rewarding. :-)

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting

Phone: 970-221-0438 E-Mail: davidf@dfanning.com

Coyote's Guide to IDL Programming: <http://www.dfanning.com/>

Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: Plot bug or another "strange feature" ?

Posted by [Nicolas Decoster](#) on Tue, 16 May 2000 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

David Fanning wrote:

- >
- >
- > The thing about IDL is that just when you've lowered
- > your expectations enough, IDL turns around and does

- > something fantastic with no effort at all. This makes
- > your expectations rise again, etc. I try to practice
- > the Buddhist attitude of "no expectations". When I
- > manage to do it correctly I find myself in a state
- > of constant surprise, no matter what happens. :-)

It sounds like a good strategy to me. I'll try it. But before I shave my head and wear the peach clothes, I have a last question:

How can I have "no expectations" from a _commercial_ product ?

...

Cheers. :-)

Nicolas.

--

Tél. : 00 (33) 5 62 88 11 16
Fax : 00 (33) 5 62 88 11 12
Nicolas.Decoster@Noveltis.fr

Noveltis
Parc Technologique du Canal
2, avenue de l'Europe
31520 Ramonville Saint Agne - France

Subject: Re: Plot bug or another "strange feature" ?
Posted by [davidf](#) on Tue, 16 May 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

- > The thing about IDL is that just when you've lowered
- > your expectations enough, IDL turns around and does
- > something fantastic with no effort at all. This makes
- > your expectations rise again, etc. I try to practice
- > the Buddhist attitude of "no expectations". When I
- > manage to do it correctly I find myself in a state
- > of constant surprise, no matter what happens. :-)

P.S. Let's just say I'm a whole lot more successful with my practice when it comes to IDL than I am when it comes to, say, my tennis game. :-)

--

David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com

Subject: Re: Plot bug or another "strange feature" ?
Posted by [Paul van Delst](#) on Tue, 16 May 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Nicolas Decoster wrote:

>

> David Fanning wrote:

>>

>> Nicolas Decoster (Nicolas.Decoster@Noveltis.fr) writes:

>>

>>> Does anybody notice and, eventually, know how to handle this "strange
>>> feature" of the plot procedure ?

>>>

>>> IDL> s = indgen(1000) + 1000000000

>>> IDL> plot, s, /ynozero

>>>

>>> The plot is not a clean line joining down-left and up-right corners, but
>>> an ugly stairway...

>>>

>>> Perhaps I missed the holy keyword...

>>

>> No, I think you are just applying analog thinking to
>> a digital solution. :-)

>>

>> Cheers,

>>

>> David

>>

>> P.S. Perhaps a scaling factor to "spread" these
>> values out some. :-)

>>

>

> A solution to get the "beautifull" line is just:

>

> IDL> plot, s-min(s), /ynozero

>

> But doing so the y axis labels are wrong.

No they're not. I tried your example and all is as expected. A straight line whose y-values go from 0-999 (y axis labels got from 0-1000 as one would hope).

> And I thought IDL was clever enough to handle this properly.

> In fact this is my main error: I'm excepting too much from IDL. :-)

Sounds more like you don't expect enough. Still, given the number of your posts over the last couple of weeks berating IDLs "feature", I salute your stick-to-it-ness. :o)

paulv

--

Paul van Delst Ph: (301) 763-8000 x7274
CIMSS @ NOAA/NCEP Fax: (301) 763-8545
Rm.202, 5200 Auth Rd. Email: pvandelst@ncep.noaa.gov
Camp Springs MD 20746

Subject: Re: Plot bug or another "strange feature" ?
Posted by [Paul van Delst](#) on Tue, 16 May 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Nicolas Decoster wrote:

>
> Hi.
>
> Does anybody notice and, eventually, know how to handle this "strange
> feature" of the plot procedure ?
>
> IDL> s = indgen(1000) + 1000000000
> IDL> plot, s, /ynozero
>
> The plot is not a clean line joining down-left and up-right corners, but
> an ugly stairway...

I thought that sort of thing came about because:

a) PLOT converts all its arguments to single precision floats. Do a
PRINT, FLOAT(s[0:100]),FORMAT='(e20.13)' to see what happens to the
float'd integers,

b) people think they can represent numbers at the extremes of machine
precision exactly. Your "s" array, when converted to single precision
float, changes in the 8th or 9th d.p. While I think that IDL should
allow users to set the PLOT conversion to double precision if they want
(e.g. with a DOUBLE keyword or something), in general you can't expect
these sorts of numbers to be represented well - in IDL or any language.
That's not how floating point arithmetic works. All floating point
numbers are approximations to their actual value.

paulv

--

Paul van Delst Ph: (301) 763-8000 x7274

CIMSS @ NOAA/NCEP Fax: (301) 763-8545
Rm.202, 5200 Auth Rd. Email: pvandelst@ncep.noaa.gov
Camp Springs MD 20746
