Subject: Re: Floating Underflow/Overflow
Posted by Craig Markwardt on Mon, 15 Oct 2001 14:54:30 GMT
View Forum Message <> Reply to Message

bente@uni-wuppertal.de (Kay) writes:

> Hi,
>
> i get Floating Overflow/Underflow error messages during my
> calculations, but the result seems to be correct, can these warnings
> be ignored then?

> edges and this seemed to be the easiest way) I think, that the results
> get to low for larger radiuses so IDL makes this error message. Is it
> possible to tell IDL to round to zero then or what do i have to do?

Hi Kay--

In all likelihood you can ignore the over- and underflows. You can use !EXCEPT = 2 to find out where in your program the exceptions are being created. You can also turn off exceptions using !EXCEPT if that is crucial.

While I have argued in the past that most people don't need underflow errors, and that they should be silent, other folks on the newsgroup have argued that we should strive to avoid them, so such errors should be printed. Leaving the question of right vs. wrong on error messages aside for the moment, I indeed think it is important to avoid over and underflows.

My guess is that you are getting both when you use EXP() in the Fermi distribution. To avoid this you can use some simple techniques. One idea is to use thresholding to keep all values in-bounds, like this, EXP((X) > (-1e-38) < 1e38). That is not really satisfactory though because sometimes you *want* the effect of an underflow. That is perhaps best solved using the WHERE() command to locate extremal values and treat them specially.

Good luck,
Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Floating Underflow/Overflow
Posted by Paul van Delst on Mon, 15 Oct 2001 15:55:01 GMT
View Forum Message <> Reply to Message

Kay wrote:
>
> Hi,
>
> i get Floating Overflow/Underflow error messages during my
> calculations, but the result seems to be correct, can these warnings
> be ignored then?
> Iï¿½m calculating a Fermi Distribution (I want a sphere with smooth
> edges and this seemed to be the easiest way) I think, that the results
> get to low for larger radiuses so IDL makes this error message. Is it
> possible to tell IDL to round to zero then or what do i have to do?

First test your code after setting !EXCEPT = 2. this will tell you on what lines of code you are getting the over/underflow. Then you can alter your algorithm to avoid them depending on your needs.

E.g. if the numerical precision is a good enough tolerance level, you can do stuff like

```
  ; set some tolerance level, with double keyword just in case.
  tolerance = ( machar(double=double) ).eps

  if ( x LT tolerance ) then $
    result = KEYWORD_SET( double ) ? 0.0d0 : 0.0 $
  else $
    result = y / x
```

I think a lot of posters will probably say this is overkill, and maybe it is for what you want to do. My recent experience has shown that, in general, one can ignore the underflow errors if:
1) you are sure your code will always be executed in the same regime, and
2) you don't care if you code crashes and burns, or produces a crappy number every now and again.

My e.g.:

the global forecast model here used to run up to about the, oh, around 2hPa pressure level. At these high altitudes the amount of water vapour is negligible. But it's not negligible compared to numerical precision. I prototyped, in IDL, the forward, then tangent-linear (TL), then adjoint (AD) model of the radiative transfer code used to simulate the satellite measurements in the forecast model - which uses up the FOURTH power of the integrated water vapour amount. To test the adjoint model, you expect agreement between the TL and AD modeuls to numerical precision (in double precision). I was getting huge errors at the top of the atmosphere for some satellite channels in the AD model. The reason was that my test code went up to 0.005hPa where the integrated water vapour amounts are next to nothing and (next to nothing)^4 in a denominator was catastrophic. I put in a test (similar to the above code) and the errors went away, no more under/overflow errors. The forecast model doesn't go up as high as I tested the

radiative transfer code....yet. But in the future it may (it recently got bumped up to the 0.1hPa level)

So, like I said before, it depends on what you want to do. The operational forecast model always has to run. It can't crash with an error or produce a crappy number unless it's flagged as crappy.

Just my opinion of course, but I treat all fp errors as serious.

paulv

--
Paul van Delst Religious and cultural
CIMSS @ NOAA/NCEP purity is a fundamentalist
Ph: (301)763-8000 x7274 fantasy
Fax:(301)763-8545 V.S.Naipaul

Subject: Re: Floating Underflow/Overflow
Posted by bente on Tue, 16 Oct 2001 09:24:18 GMT
View Forum Message <> Reply to Message

Thanks alot!
Kay

Subject: Re: Floating Underflow/Overflow
Posted by George N. White III on Tue, 16 Oct 2001 15:19:37 GMT
View Forum Message <> Reply to Message

On Mon, 15 Oct 2001, Paul van Delst wrote:

> Kay wrote:
>> i get Floating Overflow/Underflow error messages during my
>> calculations, but the result seems to be correct, can these warnings
>> be ignored then?
>> [...]

"Seems to be correct" only offers some hope that when you understand the reason for underflow you will realize that it doesn't affect your results. Even then, you may want to alter the program to avoid underflow:

1. underflow is often an expensive way to set a variable to zero -- when the source of underflow is understood it may become obvious that a significant chunk of calculation isn't needed and shouldn't be used (e.g., by adding a range test to omit the section where underlow occurs when the

inputs are "out of bounds").

2. modern hardware with branch prediction and combined f.p. ops
is generally tuned for peak performance in typical cases, and can
fall down very badly when handling exceptions. Even if your current
program performs adequately, once you have analyzed the underflow
you may want to document it in case you encounter performance problems
in the future.

> First test your code after setting !EXCEPT = 2. this will tell you on
> what lines of code you are getting the over/underflow. Then you can
> alter your algorithm to avoid them depending on your needs.
> [...]

> So, like I said before, it depends on what you want to do. The
> operational forecast model always has to run. It can't crash with an
> error or produce a crappy number unless it's flagged as crappy.
>
> Just my opinion of course, but I treat all fp errors as serious.

I suspect that operational forecast models have to run in a set
amount of time too.

--
George N. White III <gnw3@acm.org> Bedford Institute of Oceanography

---

Subject: Re: Floating Underflow/Overflow
Posted by Craig Markwardt on Tue, 16 Oct 2001 19:18:21 GMT
View Forum Message <> Reply to Message

"George N. White III" <WhiteG@dfo-mpo.gc.ca> writes:

> On Mon, 15 Oct 2001, Paul van Delst wrote:
>
>> Kay wrote:
>>> i get Floating Overflow/Underflow error messages during my
>>> calculations, but the result seems to be correct, can these warnings
>>> be ignored then?
>>> [...]
>
> "Seems to be correct" only offers some hope that when you understand
> the reason for underflow you will realize that it doesn't affect
> your results. Even then, you may want to alter the program to
> avoid underflow:
>
> 1. underflow is often an expensive way to set a variable to zero -- when
> the source of underflow is understood it may become obvious that a

> significant chunk of calculation isn't needed and shouldn't be used (e.g.,
> by adding a range test to omit the section where underlow occurs when the
> inputs are "out of bounds").
>
> 2. modern hardware with branch prediction and combined f.p. ops
> is generally tuned for peak performance in typical cases, and can
> fall down very badly when handling exceptions. Even if your current
> program performs adequately, once you have analyzed the underflow
> you may want to document it in case you encounter performance problems
> in the future.

On the other hand, the performance of IDL falls down rather badly when dealing with conditional tests on large arrays, especially when FOR loops cannot be avoided. Even using WHERE() usually makes a pretty large performance hit.

One little trick for avoiding underflows in exponentials might go like this. If you wish to compute Y = EXP(-X), for large X, then you can usually use a "mask" variable to avoid the underflow.

```
MASK = (X LT UPPER_LIMIT)
Y = MASK * EXP(-X*MASK)
```

Note that this doesn't catch overflows which will be more disastrous.

Craig

--

-- --------------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
-- --------------

Subject: Re: Floating Underflow/Overflow
Posted by bente on Thu, 18 Oct 2001 08:05:39 GMT
View Forum Message <> Reply to Message

Hi, again.

Wow didnï¿½t expected to get so much response ;-)

> On the other hand, the performance of IDL falls down rather badly when
> dealing with conditional tests on large arrays, especially when FOR
> loops cannot be avoided. Even using WHERE() usually makes a pretty
> large performance hit.

The peformance is the large problem I have, my PC isnï¿½t so fast (350MHz with only 128MB Ram. And i have to work through a 256x256x128 floating Point array with 3 FOR-Loops (i need the complete Indizes to get the Radius from a specific point to the current Voxel (donï¿½t no some faster way to get this)

Itï¿½s not so that this lasts hours then, but i gets annoying if you want to change a value a bit and then wait several minutes for the result

> One little trick for avoiding underflows in exponentials might go like
> this. If you wish to compute Y = EXP(-X), for large X, then you can
> usually use a "mask" variable to avoid the underflow.
>
> MASK = (X LT UPPER_LIMIT)
> Y = MASK * EXP(-X*MASK)

Iï¿½ll try this out, thanks for the hint.
