Subject: Re: Locate an underflow
Posted by Paul van Delst on Wed, 23 May 2001 16:59:29 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
>
> m.hadfield@niwa.cri.nz ("Mark Hadfield") writes:
>
>> From: "Francesco" <francesco.spada@jrc.it>
>>> I've got this error
>>>
>>> % Program caused arithmetic error: Floating underflow
>>>
>>> Is it possible to know where it appens without putting a million of
>>> stop or breakpoint in my program?
>
>> Set the !EXCEPT system variable to 2.
>
>> From IDL 5.4 documentation:
>
>>   !EXCEPT
>
>>   An integer variable that controls when IDL checks for invalid
>>   mathematical computations (exceptions), such as division by zero.
>>   The three allowed values are:
>
>>   Value Description
>
>>   0 Never report exceptions.
>
>>   1 Report exceptions when the interpreter is returning to an
>>   interactive prompt (the default).
>
>>   2 Report exceptions at the end of each IDL statement. Note that
>>   this slows IDL by roughly 5% compared to setting !EXCEPT=1.
>
>>   For more information on invalid mathematical computations and
>>   error reporting, see Math Errors. The value of !EXCEPT is used by
>>   the CHECK_MATH function to determine when to return errors. See
>>   CHECK_MATH for details.
>
>> ---
>> Mark Hadfield
>> m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
>> National Institute for Water and Atmospheric Research
>
> What I would really like is something that would differentiate between
> underflow errors and all other exceptions. I think that most users don't know

> what an underflow "error" is, and their initial reaction is that something bad
> is going on.

Hmm. I do see your point, but if I grab someone else's code (not just IDL code BTW) the first thing I do is run their supplied test case (I hope there is one) with all warning flags on (for IDL, !EXCEPT = 2; for Fortran or similar, set the platform specific compiler switch to trap under/overflows, divide by zero, etc.).

If, on running said code, I get a crapload of underflow errors, it's an indication that that either a) the code hasn't been tested very well or b) the programmer didn't really think about the problem enough (and I'm guilty of both of these.... most of the time actually). If there are (usually harmelss) underflow errors, how do I know that there won't be other more serious errors at some point for different input?

Writing code for operational use (i.e. has to work all the time and when it doesn't has to handle the error) has made me think a lot more carefully about what numbers are getting crunched and/or squashed in my code.

> Even when you do know that underflow "errors" are (almost always) harmless,
> continuously getting these messages is highly annoying. I'd use !EXCEPT=0 to
> suppress underflow messages, except that it would also suppress other
> potentially more serious error messages as well.

You state that there are cases where underflows are not harmless (via the "almost always" qualifier) but then go on to say you would like to suppress their reporting? Why not fix the error so you don't get underflows for that rare case where they CAN cause a problem?

paulv

--
Paul van Delst   A little learning is a dangerous thing;
CIMSS @ NOAA/NCEP   Drink deep, or taste not the Pierian spring;
Ph: (301)763-8000 x7274 There shallow draughts intoxicate the brain,
Fax:(301)763-8545   And drinking largely sobers us again.
                           Alexander Pope.