Subject: Re: Turning off math error checking for a code block
Posted by John-David T. Smith on Thu, 17 Jan 2002 20:30:09 GMT
View Forum Message <> Reply to Message

Paul van Delst wrote:
>
> Kenneth Bowman wrote:
>>
>> In article <3C47094C.1F1879D2@ssec.wisc.edu>, "Liam E. Gumley"
<Liam.Gumley@ssec.wisc.edu> wrote:
>>
>>> The FINITE function returns 1 where the argument is finite, and 0 where
>>> the argument is infinite *or* NaN (see p. 134 of my book). Try the
>>> following:
>>>
>>> x_min = 2.0
>>> index = where(finite(x) eq 1, count)
>>> if (count gt 0) then print, where(x[index] lt x_min)
>>
>> I am aware of that. These are relatively large vectors (10^5 to 10^6 elements),
>> however, and this operation is repeated many times, so I am trying to avoid
>> extracting the finite values (or creating an array index to them). This is my
>> "innermost loop", and efficiency is important. I know there are NaN's. I prefer
>> to simply turn off the error messages.
>
> Hmm. This is straying way off topic...and don't take it the wrong way or anything, but how come
> you don't prefer to simply prevent the NaNs from occurring in the first place?
>
> (To the NG) Does IDL stop processing compound logical tests before they're completed? What
> about:
>
> i = WHERE((FINITE(x) EQ 1) AND (x LT x_min), ni)
>
> Will the second test for any particular index value still get performed if the first one fails?
> I should look this up in me IDL book, I know.....

No, IDL booleans are not short-circuiting, which is a real pain for
cases like:

if n_elements(a) ne 0 AND a eq 4 then ....

which requires more deeply nested if's to pull off. That said, they
are overloaded to perform array based boolean computations, so it's
not clear if a short-circuiting version would even have been possible.
The only solution might have been to keep the array and scalar boolean
operators separate.

JD