Subject: Re: Turning off math error checking for a code block
Posted by Liam E. Gumley on Thu, 17 Jan 2002 17:26:36 GMT
View Forum Message <> Reply to Message

Kenneth Bowman wrote:

```
>
> I have an array x that is likely to have missing values in it, indicated by NaN's.
> I would like to search the array for values less than x_min.  Because of the NaN's,
> WHERE generates a floating point error, e.g.,
>
> IDL> print, x
>      0.00000        NaN      2.00000      3.00000
> IDL> print, where(x lt 2.0)
>           0
> % Program caused arithmetic error: Floating illegal operand
>
> As best I understand the interaction between !EXCEPT and CHECK_MATH,
> in order to suppress this error message, while still checking errors elsewhere
> in the code, I must do the following:
>
> error       = CHECK_MATH(/PRINT)          ;If any errors have occurred, print
> save_except = !EXCEPT                     ;Save current exception flag
> !EXCEPT     = 0                           ;Set exception flag to 0
> i           = WHERE(x LT x_min, ni)       ;Find all x < x_min
> error       = CHECK_MATH()                ;Clear accumulated error status
> !EXCEPT     = save_except                 ;Restore exception flag
>
> Am I making this harder than it needs to be?
```

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)
```

Cheers,
Liam.
Practical IDL Programming
http://www.gumley.com/

Subject: Re: Turning off math error checking for a code block
Posted by Paul van Delst on Thu, 17 Jan 2002 17:30:57 GMT
View Forum Message <> Reply to Message

Kenneth Bowman wrote:

>
> I have an array x that is likely to have missing values in it, indicated by NaN's. I would like to search the array for values less than x_min. Because of the NaN's, WHERE generates a floating point error, e.g.,
>
```
>  IDL> print, x
>       0.00000          NaN      2.00000      3.00000
>  IDL> print, where(x lt 2.0)
>            0
>  % Program caused arithmetic error: Floating illegal operand
```

Flag the NaN's and Inf's with the result of the FINITE statement. Then do a WHERE on the result.

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: Turning off math error checking for a code block
Posted by Vapuser on Thu, 17 Jan 2002 19:30:17 GMT

View Forum Message <> Reply to Message

k-bowman@null.com (Kenneth Bowman) writes:

> I have an array x that is likely to have missing values in it, indicated by NaN's. I would like to search the array for values less than x_min. Because of the NaN's, WHERE generates a floating point error, e.g.,
>
```
>  IDL> print, x
>       0.00000          NaN      2.00000      3.00000
>  IDL> print, where(x lt 2.0)
>            0
>  % Program caused arithmetic error: Floating illegal operand
>
```

Hmmmm..... I don't get this result.

```
IDL Version 5.3 (IRIX mipseb). (c) 1999, Research Systems, Inc.
Installation number: 12619.
Licensed for use by: Jet Propulsion Lab
```

```
IDL> y=[0,!values.f_nan,2,0.]
IDL> print,where( y LT 2,nx),nx
           0           3
           2
% Program caused arithmetic error: Floating illegal operand
IDL> !except=0
IDL> print,where( y LT 2,nx),nx
           0           3
           2
IDL> exit
```

The help says that for !except=1 (the default) it only reports the exception upon arriving at back at an interactive prompt. On my SGI, it still does the `where' and returns the correct answer, it just complains.

```
>
> As best I understand the interaction between !EXCEPT and CHECK_MATH,
> in order to suppress this error message, while still checking errors
> elsewhere in the code, I must do the following:
>
> error       = CHECK_MATH(/PRINT)          ;If any errors have occurred, print
> save_except = !EXCEPT                     ;Save current exception flag
> !EXCEPT     = 0                           ;Set exception flag to 0
> i           = WHERE(x LT x_min, ni)       ;Find all x < x_min
> error       = CHECK_MATH()                ;Clear accumulated error status
> !EXCEPT     = save_except                 ;Restore exception flag
>
> Am I making this harder than it needs to be?
>
```

You could use FINITE. But why bother? Your method saves you one iteration over the array.

whd
--
William Daffer: 818-354-0161: William.Daffer@jpl.nasa.gov

Subject: Re: Turning off math error checking for a code block
Posted by k-bowman on Thu, 17 Jan 2002 19:48:31 GMT
View Forum Message <> Reply to Message

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.

Perhaps a /NAN keyword on the WHERE function would be useful? (as in, for example, TOTAL)

Ken

P.S. For symmetry, wouldn't it be nice to have an INFINITE function?

Subject: Re: Turning off math error checking for a code block
Posted by Paul van Delst on Thu, 17 Jan 2002 20:23:54 GMT

View Forum Message <> Reply to Message

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......

paulv

>
> P.S. For symmetry, wouldn't it be nice to have an INFINITE function?

And don't forget the complementary NOT_A_NAN() function. :o)

--
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: 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

Subject: Re: Turning off math error checking for a code block
Posted by Paul van Delst on Thu, 17 Jan 2002 20:54:31 GMT
View Forum Message <> Reply to Message

Vapuser wrote:
>
> k-bowman@null.com (Kenneth Bowman) writes:
>
>> I have an array x that is likely to have missing values in it, indicated by NaN's. I would like to search the array for values less than x_min. Because of the NaN's, WHERE generates a floating point error, e.g.,
```
>>
>> IDL> print, x
>>       0.00000        NaN      2.00000      3.00000
>> IDL> print, where(x lt 2.0)
>>            0
>> % Program caused arithmetic error: Floating illegal operand
>>
>
```
>   Hmmmm..... I don't get this result.

Sure you do - you just have a 0 instead of a 3 as the 4th element of y:

```
> IDL> y=[0,!values.f_nan,2,0.]
```

The where result of the original is one index - the 0th one:

```
IDL> x=[0,!values.f_nan,2,3.]
IDL> print,where( x LT 2,nx),nx
           0
           1
% Program caused arithmetic error: Floating illegal operand
```

paulv

p.s. After trying the simple example above, it sure is annoying to get that error message. :o\

--
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: Turning off math error checking for a code block
Posted by Pavel A. Romashkin on Thu, 17 Jan 2002 21:36:08 GMT
View Forum Message <> Reply to Message

You've got a lot of input. Now you know how to avoid the situation
causing the message. In case you just want to not see the *message* and
couldn't care less about floating errors *in this case*, how about trying

!Except = 0

Pavel

Kenneth Bowman wrote:
>
> As best I understand the interaction between !EXCEPT and CHECK_MATH, in order to suppress this error message, while still checking errors elsewhere in the code, I must do the following:
>

Subject: Re: Turning off math error checking for a code block
Posted by Vapuser on Thu, 17 Jan 2002 21:55:35 GMT
View Forum Message <> Reply to Message

Paul van Delst <paul.vandelst@noaa.gov> writes:

> Vapuser wrote:
>>
>> k-bowman@null.com (Kenneth Bowman) writes:

```
>>>
>>>  I have an array x that is likely to have missing values in it,
>>>  indicated by NaN's.  I would like to search the array for values
>>>  less than x_min.  Because of the NaN's, WHERE generates a
>>>  floating point error, e.g.,
>>>
>>>  IDL> print, x
>>>       0.00000         NaN      2.00000      3.00000
>>>  IDL> print, where(x lt 2.0)
>>>            0
>>>  % Program caused arithmetic error: Floating illegal operand
>>>
>>
>>    Hmmmm..... I don't get this result.
>
>  Sure you do - you just have a 0 instead of a 3 as the 4th element of y:
>
```

Oooopppsss. My error! I guess I just can't read today. For some
reason I thought that Ken's x[3] == 0 too!

mea culpa.

```
>>  IDL> y=[0,!values.f_nan,2,0.]
>
>  The where result of the original is one index - the 0th one:
>
```

Yes.

```
>  IDL> x=[0,!values.f_nan,2,3.]
>  IDL> print,where( x LT 2,nx),nx
>            0
>            1
>  % Program caused arithmetic error: Floating illegal operand
>
>  paulv
>
>  p.s. After trying the simple example above, it sure is annoying to
>  get that error message. :o\
>
```

True, but setting !except=0 makes that go away.

whd
--

William Daffer: 818-354-0161: William.Daffer@jpl.nasa.gov

Subject: Re: Turning off math error checking for a code block
Posted by k-bowman on Thu, 17 Jan 2002 22:50:16 GMT
View Forum Message <> Reply to Message

In article <888zawn5bq.fsf@catspaw.jpl.nasa.gov>, Vapuser <vapuser@catspaw.jpl.nasa.gov> wrote:

> The help says that for !except=1 (the default) it only reports the
> exception upon arriving at back at an interactive prompt. On my SGI, it
> still does the `where' and returns the correct answer, it just
> complains.

The problem is that changing !EXCEPT does not reset the accumulated error status, so the errors that occur with !EXCEPT = 0 are still present. Hence the need to call CHECK_MATH (twice).

I want to be able to run my whole code with !EXCEPT = 2 (or 1) but not generate errors for the blocks that I *know* will have floating point exceptions in them.

I think my solution is probably the simplest way to do what I want to do.

Thanks everybody, Ken

Subject: Re: Turning off math error checking for a code block
Posted by k-bowman on Thu, 17 Jan 2002 22:50:54 GMT
View Forum Message <> Reply to Message

In article <3C4732DA.E40F6BA6@noaa.gov>, Paul van Delst <paul.vandelst@noaa.gov> wrote:

> Kenneth Bowman wrote:

> 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?

I am doing repeated interpolation on a limited domain [x_min, x_max]. Each time I need to check to see if any interpolation points have moved outside the domain bounds. If so, I set them to NaN.

There are other ways to solve this, but I prefer using NaNs as "missing data" flags. If I inadvertently use an NaN elsewhere in my code, I want to generate a floating point exception.

Regards, Ken

Subject: Re: Turning off math error checking for a code block
Posted by Craig Markwardt on Fri, 18 Jan 2002 00:06:22 GMT
View Forum Message <> Reply to Message

k-bowman@null.com (Kenneth Bowman) writes:

> 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.

I have found that an operation on an array which contains NANs is
slowed down considerably. I think it is because each operation causes
a floating point exception which is handled in the OS. I use WHERE
most of the time when this comes up. Occassionally I get "floating
exception" messages, but big whoop.

Craig

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

Subject: Re: Turning off math error checking for a code block
Posted by Paul van Delst on Fri, 18 Jan 2002 15:47:32 GMT
View Forum Message <> Reply to Message

Kenneth Bowman wrote:
>
> In article <888zawn5bq.fsf@catspaw.jpl.nasa.gov>, Vapuser <vapuser@catspaw.jpl.nasa.gov>
wrote:
>

>>   The help says that for !except=1 (the default) it only reports the
>>   exception upon arriving at back at an interactive prompt. On my SGI, it
>>   still does the `where' and returns the correct answer, it just
>>   complains.
>
> The problem is that changing !EXCEPT does not reset the accumulated error status, so the errors that occur with !EXCEPT = 0 are still present. Hence the need to call CHECK_MATH (twice).
>
> I want to be able to run my whole code with !EXCEPT = 2 (or 1) but not generate errors for the blocks that I *know* will have floating point exceptions in them.
>
> I think my solution is probably the simplest way to do what I want to do.

Given the (valid) reason the NaN's are in your array (flagging missing values) and that you want to _not_ accumulate certain well-identified errors, I think you're right.

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: Turning off math error checking for a code block
Posted by Martin Downing on Fri, 18 Jan 2002 17:22:37 GMT
View Forum Message <> Reply to Message

craig wrote:
> I have found that an operation on an array which contains NANs is
> slowed down considerably. I think it is because each operation causes
> a floating point exception which is handled in the OS. I use WHERE
> most of the time when this comes up. Occassionally I get "floating
> exception" messages, but big whoop.

To illustrate craigs point:

```
IDL> a = replicate(!values.f_nan,1024,1024)

IDL> b = replicate(2.0,1024,1024)

IDL> help, a,b

A FLOAT = Array[1024, 1024]
```

```
B FLOAT = Array[1024, 1024]

IDL> t = systime(1) & for i =0,9 do c = total(a * 2) & print, systime(1) - t

3.1250000

IDL> t = systime(1) & for i =0,9 do c = total(a * 2) & print, systime(1) - t

3.1240001

IDL> a = replicate(2.0,1024,1024)

IDL> help, a

A FLOAT = Array[1024, 1024]

IDL> t = systime(1) & for i =0,9 do c = total(a * 2) & print, systime(1) - t

0.71099997
```

Subject: Re: Turning off math error checking for a code block
Posted by Vapuser on Fri, 18 Jan 2002 17:45:48 GMT
View Forum Message <> Reply to Message

k-bowman@null.com (Kenneth Bowman) writes:

> In article <888zawn5bq.fsf@catspaw.jpl.nasa.gov>, Vapuser <vapuser@catspaw.jpl.nasa.gov> wrote:
>
>> The help says that for !except=1 (the default) it only reports the
>> exception upon arriving at back at an interactive prompt. On my SGI, it
>> still does the `where' and returns the correct answer, it just
>> complains.
>
> The problem is that changing !EXCEPT does not reset the accumulated
> error status, so the errors that occur with !EXCEPT = 0 are still
> present. Hence the need to call CHECK_MATH (twice).
>

Oh, absolutely. I was only addressing the issue of the annoying
error message itself. !except=0 doesn't turn off the exception, just
the message. It's somewhat misnamed, I would have called it
'math_error_verbosity' or some such.

> I want to be able to run my whole code with !EXCEPT = 2 (or 1) but
> not generate errors for the blocks that I *know* will have floating
> point exceptions in them.
>
> I think my solution is probably the simplest way to do what I want to do.
>

I agree completely.

> Thanks everybody, Ken

whd

--
William Daffer: 818-354-0161: William.Daffer@jpl.nasa.gov