
Subject: Turning off math error checking for a code block
Posted by [k-bowman](#) on Thu, 17 Jan 2002 16:26:07 GMT
[View Forum Message](#) <> [Reply to Message](#)

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?

Ken

Subject: Re: Turning off math error checking for a code block
Posted by [k-bowman](#) on Fri, 18 Jan 2002 22:47:42 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <Ln_18.43167\$WQ1.6917653@news6-win.server.ntlworld.com>, "Martin Downing" <martin.downing@ntlworld.com> wrote:

```
> 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.
```

Actually, I don't think it is faster. The program below creates an array with 10⁶ elements with about 10% of the values set to NaN. The goal is to set all the finite values of the array less than x_min to NaN.

I tried two methods

- 1) turning off error messages and doing a single WHERE, and
- 2) doing one WHERE to find the finite values and a second WHERE to find the values less than `x_min`.

The problem is that the two WHERE's require creating an intermediate array and repeated indirect indexing. It may be possible to do the second method more efficiently, but I don't see how.

The code below produces

```
IIDL> TEST_FINITE
Turn off error checking, t = 0.082054019
Do WHERE twice,          t = 1.7191260
```

so, turning off error messages seems to be much faster.

Also, I prefer that my codes produce floating point errors only when there really are floating point errors. I don't want to get in the habit of ignoring FP error messages.

Ken

```
PRO TEST_FINITE
```

```
n      = 1024L*1024L
nmiss  = 100L*1024L
x      = RANDOMN(seed, n)           ;Create 10^ random numbers
miss   = LONG(n*RANDOMU(seed, nmiss)) ;Generate 10^5 random indices
x[miss] = !VALUES.F_NAN             ;Set random indices to missing
x_min  = 0.0
```

```
t      = SYSTIME(/SECONDS)
error  = CHECK_MATH(/PRINT)
save_except = !EXCEPT
!EXCEPT = 0
i      = WHERE(x LT x_min, ni)
error  = CHECK_MATH()
!EXCEPT = save_except
PRINT, 'Turn off error checking, t = ', SYSTIME(/SECONDS) - t
```

```
t = SYSTIME(/SECONDS)
i = WHERE(FINITE(x), ni)
IF (ni GT 0) THEN BEGIN
  y = x[i]
  j = WHERE(y LT x_min, nj)
```

```

IF (nj GT 0) THEN BEGIN
    y[j] = !VALUES.F_NAN
    x[i] = y
ENDIF
ENDIF
PRINT, 'Do WHERE twice,      t = ', SYSTIME(/SECONDS) - t

END

```

Subject: Re: Turning off math error checking for a code block
 Posted by [bowman](#) on Fri, 18 Jan 2002 23:30:43 GMT
[View Forum Message](#) <> [Reply to Message](#)

Well, it makes more sense if both blocks of code do the same work. ;-)

Turning off error checking is still faster, though.

```

Turn off error checking, t =    0.45098495
Do WHERE twice,      t =    2.5393230

```

Ken

PRO TEST_FINITE

```

n      = 1024L*1024L
nmiss  = 100L*1024L
x      = RANDOMN(seed, n)           ;Create 10^ random numbers
miss   = LONG(n*RANDOMU(seed, nmiss)) ;Generate 10^5 random indices
x[miss] = !VALUES.F_NAN             ;Set random indices to missing
x_min  = 0.0

```

```

t      = SYSTIME(/SECONDS)
error  = CHECK_MATH(/PRINT)
save_except = !EXCEPT
!EXCEPT = 0
i      = WHERE(x LT x_min, ni)
IF (ni GT 0) THEN x[i] = !VALUES.F_NAN
error  = CHECK_MATH()
!EXCEPT = save_except
PRINT, 'Turn off error checking, t = ', SYSTIME(/SECONDS) - t

```

```

t = SYSTIME(/SECONDS)
i = WHERE(FINITE(x), ni)
IF (ni GT 0) THEN BEGIN

```

```
y = x[i]
j = WHERE(y LT x_min, nj)
IF (nj GT 0) THEN BEGIN
    y[j] = !VALUES.F_NAN
    x[i] = y
ENDIF
ENDIF
PRINT, 'Do WHERE twice,      t = ', SYSTIME(/SECONDS) - t

END
```
