Subject: MIN(), MAX() with NaN values
Posted by wlandsman on Sat, 19 Jul 2014 14:24:14 GMT
View Forum Message <> Reply to Message

The documentation for MIN() says that

"Note: If the MIN function is run on an array containing NaN values and the NAN keyword is not set, an invalid result will occur."

However, most of the time MIN() seems to work fine with NaN values.

```
IDL> print,!version
{ x86_64 darwin unix Mac OS X 8.3 Nov 15 2013      64      64}
IDL> a = randomn(seed,500,500)
IDL> a[1,1] = replicate(!values.f_nan,4)
IDL> print,min(a)
    -4.76949
```

The invalid results seem to occur when the NaN values are at the edge of the image (or in the first or last value of a vector)

```
IDL> a = findgen(10)
IDL> a[0] = !values.f_nan
IDL> print,min(a)
         NaN
```

I can't fault the design or documentation of MIN() -- it says to use /NAN when NaN values are present. But we had difficulty debugging our code since our images have a few randomly located NaN values, and so using MIN() without /NaN was working 99% of the time. We have now added /NaN to the code -- and taken the factor of ~3 speed penalty -- so it works 100% of the time.

Subject: Re: MIN(), MAX() with NaN values
Posted by Matthew Argall on Sun, 20 Jul 2014 03:00:09 GMT
View Forum Message <> Reply to Message

I noticed the same thing earlier today!

```
IDL> print, min([1.0, !values.f_nan])
     1.00000
IDL> print, max([1.0, !values.f_nan])
     1.00000
IDL> print, min([1.0, !values.f_nan], NAN=0)
     1.00000
IDL> print, max([1.0, !values.f_nan], NAN=0)
     1.00000
IDL> help, !version
```

```
** Structure !VERSION, 8 tags, length=104, data length=100:
   ARCH            STRING    'x86_64'
   OS              STRING    'darwin'
   OS_FAMILY       STRING    'unix'
   OS_NAME         STRING    'Mac OS X'
   RELEASE         STRING    '8.2'
   BUILD_DATE      STRING    'Apr 10 2012'
   MEMORY_BITS     INT             64
   FILE_OFFSET_BITS
                   INT             64
```