Subject: Re: Comparing 2 arrays
Posted by Paul Van Delst[1] on Mon, 27 Aug 2007 17:52:04 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Conor writes:
>
>> Hmm... I think Jean might be on to something. After all, the error in
>> question hear is the rounding error of the computer, and that rounding
>> error is always an error on the last 'bit' of a floating point
>> number. So for instance if you had two floating point numbers:
>>
>> 1.1123453e15
>> and
>> 1.1123454e15
>>
>> These might be the same number (to within the rounding error) but the
>> difference between them is about 6.7e07. That's assuming of course
>> that I'm properly understanding floating point representation (I'm an
>> astronomer, not a computer engineer).
>
> I guess I'm thinking more about how a number got INTO
> the array in the first place. If it got there as a result
> of some kind of calculation, accumulative rounding errors
> could be a great deal larger than machine precision. And
> yet, you might still want these numbers to be "equal" for
> analysis purposes.

Jean H. is absolutely correct. The magnitude of the numbers in question must enter into the computation of the value that corresponds to "close enough to zero".

E.g. If elements of the array that you are comparing are 1.23456789e+64 and 1.23456783e+64, then comparing to an absolute value that assumes the values are near 1.0 (such as 1.0e-06) will alway fail even if they are considered equal for the application in question.

In Fortran95-speak, the equivalent is:

```
  ABS(x - y) < (ulp * SPACING(MAX(ABS(x),ABS(y))))
```

where if the result is .TRUE., the numbers are considered equal.

The intrinsic function SPACING(x) returns the absolute spacing of numbers near the value of x,

```
              {    EXPONENT(x)-DIGITS(x)
              {  2.0                    for x /= 0
   SPACING(x) = {
```

{
{ TINY(x)   for x == 0

The "ulp" factor scales the comparison.

So, Jean H's "epsilon * data1.A" is a way of computing the absolute spacing between two numbers in IDL-space.

cheers,

paulv
