Subject: compare 2-d array with vector
Posted by Danxia on Mon, 24 Sep 2012 07:32:17 GMT
View Forum Message <> Reply to Message

Hi all, I'm having problem when comparing a 2-d array with a given vector. For example, if giving a 2-d array
a=[[1,1,1],
[2,2,2],
[3,3,3],
[4,4,4]]
and a vector b=[1,2],
how can I get all the subscripts of array a that's equal to any element in vector b, which in this example is [0,1,2,3,4,5]. Please let me know if I failed to make this clear. Looking forward to any reply. Thank you very much!

Subject: Re: compare 2-d array with vector
Posted by Craig Markwardt on Tue, 18 Dec 2012 01:45:40 GMT
View Forum Message <> Reply to Message

On Monday, December 17, 2012 2:25:30 PM UTC-5, Chris Torrence wrote:
> On Monday, September 24, 2012 9:30:25 AM UTC-6, Mark Piper wrote:
>
>> On Monday, September 24, 2012 9:19:38 AM UTC-6, Craig Markwardt wrote:
>
>>
>
>>>
>
>>
>
>>> Yeah, I filed a support request on this several years ago, but no action so far.
>
>>
>
>>>
>
>>
>
>>
>
>>
>
>>
>
>> Heinz & Craig,
>
>>
>
>>
>

>
>>
>
>> This looks eminently reasonable. I'll look at getting it into the backlog today.
>
>>
>
>>
>
>>
>
>> mp
>
>
>
> This is now fixed and will be in IDL 8.8.2. You can now pass a 1-element array or a scalar into value_locate.

Like. Thanks.

Subject: Re: compare 2-d array with vector
Posted by Fabzi on Tue, 08 Jan 2013 12:53:12 GMT
View Forum Message <> Reply to Message

On 01/07/2013 11:30 PM, Jeremy Bailin wrote:
> What do you expect him to do? Our conclusion was that it's inherently
> undefined, so there's not much point in asking for consistency.
>
> -Jeremy.

You're right it's a detail but I would expect value_locate to return -1
and no warning message.

let's say I want to attribute rank -1 to missing data, 0 to data below
my first level bound and so on.

Solution 1 (throwing a warning):

```
data = FINDGEN(10) & data[1] = !VALUES.F_NAN
levels = [1L,3,6,9]
pnovalid = where(~ FINITE(data), n_novalid)
rank = VALUE_LOCATE(levels, data) + 1
if n_novalid ne 0 then rank[pnovalid] = -1
print, rank
```

Solution 2 (no warning):

```
data = FINDGEN(10) & data[1] = !VALUES.F_NAN
levels = [1L,3,6,9]
pvalid = where(FINITE(data), n_valid)
rank = LONARR(N_ELEMENTS(data)) - 1
rank[pvalid] = VALUE_LOCATE(levels, data[pvalid]) + 1
print, rank
```

I agree, they are quite similar but I would expect value_locate to be at least consistent and maybe be clearer in the documentation. Nothing critical, of course.
