Subject: using the WHERE function on a portion of an array
Posted by becky_s on Tue, 04 Mar 2008 19:23:56 GMT
View Forum Message <> Reply to Message

Dear all,

Please lend me your great expertise to help me solve this problem I have with the WHERE function.

I have a 3d array of heights, A, and another 3d array of observations at those heights, B. I have a third 3d array, C. I would like C[0,*,*] to contain values of B only if the corresponding value of A is between 0 and 1; C[1,*,*] would have values of B only if 1<=A<2, etc.

I thought this could be done via a WHERE function call, such as:
indices = WHERE(A[0,*,*] ge 4 AND A[0,*,*] lt 5, count)
if count gt 0 then C[4,indices] = B[0,indices]

but this does not work. Printing A[0,indices], I can see that these values are not b/w 4 and 5.

On the other hand, if I set each level I am looking at to its own 2d array, i.e.,
leva = A[0,*,*]
levb = B[0,*,*]
levc = C[4,*,*]
use these values in the same code written above, and add the statement at the end that C[4,*,*] = levc, then it works just fine. However, A and B are actually very large, so this isn't an option.

I'm guessing I do not understand some key part of the WHERE function. Would someone please shine some light on this for me? Thanks in advance.
Becky

Subject: Re: using the WHERE function on a portion of an array
Posted by Jean H. on Tue, 04 Mar 2008 23:20:45 GMT
View Forum Message <> Reply to Message

becky_s wrote:
> On Mar 4, 2:15 pm, Jean H <jghas...@DELTHIS.ucalgary.ANDTHIS.ca>
> wrote:
>
>> Now you are try to apply your 2D array in a 3D one, which can not work
>> properly.
>> To access your 3D array, you must either have a 3D index, or have a 1D

>> index.
>>
>> So in your case, you want to write in C, on the 5th plane:
>> indices1D_C = indices + (n_elements(C[0,*,*]) * 4
>> And you want to read B on the 1st plane:
>> indices1D_B = indices
>>
>> and then C[indices1D_C] = B[indices1D_B]
>>
>
> Jean,
> Well, that is pretty slick! I knew there had to be some problem with
> all my 2d to 3d dimension switching I was doing.
>
> I did have to modify your solution somewhat, though. I ended up with
> (I also generalized my previous code somewhat):
>
> indices = WHERE(A[i,*,*] ge j AND A[i,*,*] lt (j+1), count)
> if count gt 0 then begin
> indices1D_C = indices*n_elements(C[*,0,0]) + j
> indices1D_B = indices*n_elements(A[*,0,0]) + i
> C[indices1D_C] = B[indices1D_B]
> endif
>
> Thanks again.

Are you sure you are getting the correct indexes like that??

if you multiply the index by the number of elements, the result will be
wrong and can be out of bound!
ex: n_elements(A[*,0,0]) = 100 (10*10 array)
index = 99 (last element of the 2D array)
99 * 100 = 9 900 ==> you are in the 99th plane, even if you haven't
specified the plane!!!

on the other hand, if you know you are, let say, on the 3rd plane:
99 + (100 * (3-1)) = 299 ==> index of the last cell of the 3rd plane!

so it should really be indices1D_C = indices + n_elements(C[*,0,0]) * i

also, if you are computing this many time, you can same
n_elements(C[*,0,0]) in a variable and use it each time

Jean

Subject: Re: using the WHERE function on a portion of an array
Posted by becky_s on Fri, 07 Mar 2008 17:05:17 GMT
View Forum Message <> Reply to Message

Jean,

Thanks for the tip about n_elements, I will do that! As for the other stuff, I think we have the same idea but are talking at cross-purposes. In your above example,
> ex: n_elements(A[*,0,0]) = 100 (10*10 array)
> index = 99 (last element of the 2D array)
> 99 * 100 = 9 900 ==> you are in the 99th plane, even if you haven't
> specified the plane!!!

My "indices" variable is actually from the 2D array A[0,*,*], but my computation of the 1D index uses n_elements(A[*,0,0]). So, for example, let A be a 4x2x2 array:
2830 0 0 0
2830 0 0 0

0 0 0 0
2830 0 0 0

and B another 4x2x2 array:
20 0 0 0
60 0 0 0

0 0 0 0
9 0 0 0

indices = WHERE(A[0,*,*] ge 2830 AND A[0,*,*] lt 2831, count)
;indices evaluates to 0, 1, 3

;Now, convert to 1D:
indices1D_B = indices * n_elements(A[*,0,0]) + 0
;yields 0, 4, 12

When I plug 0, 4, and 12 into B, I get the correct values I was looking for,
B[0] = 20; B[4] = 60; B[12] = 9.

Hope that helps.
Becky