Subject: Re: Compare two variables
Posted by Joost Aan de Brugh on Fri, 11 Jul 2008 15:41:14 GMT
View Forum Message <> Reply to Message

On Jul 11, 2:05 pm, d.po...@gmail.com wrote:
> Hi everyone
> I have got a problem and I don't know how to solve it:
> I have two variables like this:
> A, which col*row=2*4
> B, which col*row=4*100.
> All A's couples (x,y)'s are somewhere in the 2th and 3th column of
> Result. I want to take this rows (I want to extract that rows in
> result which 2th and 3th columns are settled in the B's).for
> example:
> A=[[1,2], [3,4] ,[5,6], [7,8]]
>
> B=[….[11,22,1,2,],….. [44,55,3,4]….. ,[22,99,5,6], ….[77,66,7,8]…]
>
> …..=means there are some other data
> I just want to extract these rows and put them in a new variable
> Any help?
> Cheers

There are two problems. One is that you have two variables you have to compare and one is that you have a whole vector of values they can be.

Probably you are familiar with the Where-function

idx = Where(array <condition considering array as a scalar>)

idx will be a vector of indices where the array has values that meet the condition. If there are none, idx will be -1.

The problem that you have to check both the 2nd and the 3rd column will force constructions like

idx1 = Where(vector1 eq value1)
idx2 = Where(vector2 eq value2)
if idx1[0] eq -1 or idx2[0] eq -1 then screw it
...

It is better to create another vector. As you work with integers, you compress two integers into one.
totalB = B[2,*] + B[3,*]
DB = (total*(total+1))/2 + B[2,*] ; The value in DB[j] determines exactly what B[2,j] and B[3,j] are
totalA = A[0,*] + A[1,*]
DA = (total*(total+1))/2 + A[0,*] ; Same joke, so if DA[i] eq DB[j]

then A[0,i] eq B[2,j] and A[1,i] eq B[3,j]

And now we are left with two vectors. As long as DA only contains four values (like your example), you can use a forloop

```
for j=0,4-1 do begin
  idx = Where(DB eq DA[j])
  if idx[0] eq -1 then continue ; Subscripting [0] is required if idx
can have more than one value.
  if N_Elements(idxs) eq 0 then idxs = idx else idxx = [idxx,idx] ;
Add the index to the result vector.
end

result = B[*,idxs]
```

There must be a more efficient way to do the last part, but Where(DB eq DA) does not work.

You can also do it at the array way, create one array like. Watch out where to use Transposes (Check it out first)

```
MA =

DA[0]  DA[1]  DA[2]  DA[3]
DA[0]  DA[1]  DA[2]  DA[3]
DA[0]  DA[1]  DA[2]  DA[3]
DA[0]  DA[1]  DA[2]  DA[3]
DA[0]  DA[1]  DA[2]  DA[3]
...
```

and one like

```
MB =

DB[0]  DB[0]  DB[0]  DB[0]
DB[1]  DB[1]  DB[1]  DB[1]
DB[2]  DB[2]  DB[2]  DB[2]
DB[3]  DB[3]  DB[3]  DB[3]
DB[4]  DB[4]  DB[4]  DB[4]
...

MC = LonArr(4,100) ; Comparison
idx = Where(MB eq MA)
MC[idx] = 1

IDL> print,MC

0  0  0  0 ; Drop this B-index
```

0 1 0 0 ; There is an A-index that fits
0 0 0 0
0 0 1` 0
1 0 0 0
...

idxs = total(MC,1)

IDL> print,idxs

0,1,0,1,1,...

result = B[*,idxs]

; This is more work, but probably faster than a for-loop if we have
more than 4 rows in A.

I hope this helps you. Probably there are better ways, but this should
work.
