Subject: Re: Finding Common Elements in Two Arrays
Posted by dutch on Thu, 02 Jun 1994 12:04:16 GMT
View Forum Message <> Reply to Message

In article <1JUN199416454238@stars.gsfc.nasa.gov>, kucera@stars.gsfc.nasa.gov (Terry Kucera) writes:
>> I'm looking for a quick way to compare two arrays in IDL, A and B,
>> and determine which elements of B are also in A,
>> so if:
>> A=[2,1,3,5,3,8,2,5]
>> B=[3,4,2,8,7,8]
>> I would get [0,2,3,5], because 3, 2, and 8 are in A as well as B.
>>
>> I can do this with loops, but that takes too long for big arrays. Does anyone
>> have a way to do this using array functions or perhaps an external routine?
>> Terry Kucera
>> kucera@stars.gsfc.nasa.gov
>>

Perhaps the following function is what you are looking for. As it is currently written, the function returns the VALUES of repeated elements, rather than the INDICES into the first vector (as in your example), but this is a trivial modification, if necessary!.
The only hiccup, is what to return if there are no repeated elements, in my case I have chosen to return the value -999. This function is a rewrite of a similar function in MATLAB. (in MATLAB, the function can return the empty set as a null vector (with zero elements). In IDL, the function could be modified to return a second argument, which indicates the number of elements in the intersection.

Rough benchmarks on our Vax 4090 are as follows:
2 random vectors each of length 10,000 elements - about 1 second
2 random vectors each of length 100,000 elements - about 12 seconds

I expect this is considerably better than alternative solutions using one or more FOR loops!.

---------- 8< start: intersection.pro >8 ----------

```
FUNCTION intersection,a,b
;+
; general purpose routine to return the intersection
; (i.e. common elements) of two vectors. Uses USERLIB
; routine UNIQ to obtain unique elements of a,b
; Written by M.J.Dutch June-1994.
; Centre de Recherches en Physique des Plasmas, EPFL, Switzerland
;-
```

```
ab=[a(uniq(a,sort(a))),b(uniq(b,sort(b)))] ; combine unique elements of a,b
ab=ab(sort(ab))                            ; sort the combined vector
nab=n_elements(ab)
diff=ab(1:nab-1)-ab(0:nab-2)
ind=where(diff eq 0)                       ; find repeated elements
if (ind(0) ne -1) then return,ab(ind) else return,-999

end

---------- 8<    end: intersection.pro    >8 ----------



Enjoy!

-- Michael
#####################################################################
# Dr. Michael Dutch          email: dutch@eltcv1.epfl.ch #
# Centre de Recherches en Physique des Plasmas           #
# Ecole Polytechnique Federale de Lausanne               #
# 21 Ave des Bains                        Aussie.Abroad #
# CH-1007 Lausanne, SWITZERLAND                ,--_|\      #
#--------------------------------------    /      \     #
# I'd rather have a full bottle in front     \_.-*._/     #
# of me than a full frontal lobotomy.              v      #
#####################################################################
```

Subject: Re: Finding Common Elements in Two Arrays
Posted by zawodny on Thu, 02 Jun 1994 14:17:08 GMT

View Forum Message <> Reply to Message

In article <1JUN199416454238@stars.gsfc.nasa.gov>, kucera@stars.gsfc.nasa.gov (Terry Kucera) writes:
> I'm looking for a quick way to compare two arrays in IDL, A and B,
> and determine which elements of B are also in A,
> so if:
> A=[2,1,3,5,3,8,2,5]
> B=[3,4,2,8,7,8]
> I would get [0,2,3,5], because 3, 2, and 8 are in A as well as B.
>
> I can do this with loops, but that takes too long for big arrays. Does anyone
> have a way to do this using array functions or perhaps an external routine?
>   Terry Kucera
>   kucera@stars.gsfc.nasa.gov

If you are dealing strictly with integers, I'd use histogram on both

vectors, multiply them together and then use where to find non-zero values.

```
Hist_a = histogram(a,max=maximum_expected_value)
Hist_b = histogram(b,max=maximum_expected_value)
Inter  = where(Hist_a*Hist_b)
```

and Inter should then contain the array you seek.

--

Joseph M. Zawodny (KO4LW) NASA Langley Research Center
Internet: zawodny@arbd0.larc.nasa.gov MS-475, Hampton VA, 23681-0001
Packet: ko4lw@n4hog.va.usa

Subject: Re: Finding Common Elements in Two Arrays
Posted by zawodny on Thu, 02 Jun 1994 14:23:59 GMT
View Forum Message <> Reply to Message

In article <2skpl4$qr5@reznor.larc.nasa.gov> zawodny@arbd0.larc.nasa.gov (Joseph M Zawodny) writes:
> In article <1JUN199416454238@stars.gsfc.nasa.gov>, kucera@stars.gsfc.nasa.gov (Terry Kucera) writes:
>> I'm looking for a quick way to compare two arrays in IDL, A and B,
>> and determine which elements of B are also in A,
>> so if:
>> A=[2,1,3,5,3,8,2,5]
>> B=[3,4,2,8,7,8]
>> I would get [0,2,3,5], because 3, 2, and 8 are in A as well as B.
>>
>> I can do this with loops, but that takes too long for big arrays. Does anyone
>> have a way to do this using array functions or perhaps an external routine?
>> Terry Kucera
>> kucera@stars.gsfc.nasa.gov
>
> If you are dealing strictly with integers, I'd use histogram on both
> vectors, multiply them together and then use where to find non-zero values.
>
> Hist_a = histogram(a,max=maximum_expected_value)
> Hist_b = histogram(b,max=maximum_expected_value)
> Inter = where(Hist_a*Hist_b)
>
> and Inter should then contain the array you seek.

No, Sorry, I mis read your post. Inter tells you the value of the elemnts that are both arrays, not which elements are the same. Hist_b does, however, contain this information. You will also have to use the REVERSE_INDICES keyword in the second histogram call. Basically you'd have to use Inter and Hist_b jointly to construct an array of subscripts.

--

Joseph M. Zawodny (KO4LW) NASA Langley Research Center
Internet: zawodny@arbd0.larc.nasa.gov MS-475, Hampton VA, 23681-0001
Packet: ko4lw@n4hog.va.usa