Subject: Summary: comparing arrays
Posted by rfinch on Thu, 07 Mar 1991 16:02:23 GMT
View Forum Message <> Reply to Message

In article <265@locke.water.ca.gov>, rfinch@caldwr.water.ca.gov (Ralph Finch) writes:
>
> PV-Wave 3.02.
>
> I need to find all elements in 1 array that are in another. I'd like
> to do this:
>
> a=[1,2,3,4,5]
> b=[2,4]
> index=where(a eq b)
>
> However, it doesn't work. Any ideas?

I got 3 replies from people who had written routines to solve this problem. They all kindly agreed to my posting them here...they should, however, be considered the property of the authors. Please contact them for more info. The authors' names and addresses are inserted somewhere in the header lines in each routine.

There are 4 routines; function intersect calls the function runlength.

```
#!/bin/sh
# shar: Shell Archiver  (v1.22)
#
# Run the following text with /bin/sh to create:
#   match.pro
#   intersection.pro
#   intersect.pro
#   runlength.pro
#
sed 's/^X//' << 'SHAR_EOF' > match.pro &&
Xpro match,a,b,suba,subb
X;+
X; NAME:
X;   MATCH
X; PURPOSE:
X;   Routine to match values in two vectors.
X; CALLING SEQUENCE:
X; match,a,b,suba,subb
X; INPUTS:
X; a,b - two vectors to match elements
X; OUTPUTS:
X; suba - subscripts of elements in vector a with a match
X;  in vector b
```

```
X; subb - subscripts of the positions of the elements in
X;  vector b with matchs in vector a.
X;
X; suba and subb are ordered such that a(suba) equals b(subb)
X; SIDE EFFECTS:
X; !ERR is set to the number of matches.
X; RESTRICTIONS:
X; a and b should not have duplicate values within them.
X; You can use rem_dup function to remove duplicate values
X; in a vector
X; HISTORY:
X; D. Lindler  Mar. 1986.
X;-
X;;Wayne Landsman
X;;landsman@stars.gsfc.nasa.gov
 X;------------------------------------------------------ ---------------
Xna=n_elements(a)  ;number of elements in a
Xnb=n_elements(b)  ;number of elements in b
Xinda=lindgen(na)  ;indices in a
Xindb=lindgen(nb)  ;indices in b
Xc=[a,b]    ;combined list of a and b
Xind=[inda,indb]   ;combined list of indices
Xvec=[lonarr(na),replicate(1,nb)];flag of which vector in combined list
X    ; 0 - a   1 - b
X;
X; sort combined list
X;
Xsub=sort(c)
Xc=c(sub)
Xind=ind(sub)
Xvec=vec(sub)
X;
X; find duplicates in sorted combined list
X;
Xn=na+nb    ;total elements in c
Xfirstdup=where( (c eq shift(c,-1)) and (vec ne shift(vec,-1)))
Xif !err lt 1 then begin  ;any found?
X suba=lonarr(1)-1
X subb=lonarr(1)-1
X return
Xend
Xdup=lonarr(!err*2)  ;both duplicate values
Xeven=indgen(n_elements(firstdup))*2
Xdup(even)=firstdup
Xdup(even+1)=firstdup+1
Xind=ind(dup)   ;indices of duplicates
Xvec=vec(dup)   ;vector id of duplicates
Xsuba=ind(where(vec eq 0)) ;a subscripts
```

```
Xsubb=ind(where(vec eq 1)) ;b subscripts
Xreturn
Xend
SHAR_EOF
chmod 0664 match.pro || echo "restore of match.pro fails"
sed 's/^X//' << 'SHAR_EOF' > intersection.pro &&
X; intersection.pro
X;
X; hsc  2/13/91
X;
X; This function, given two vectors returns the intersection
X; of those two vectors in another vector.
X;
X;          ------------INPUT-------------
X;
X; a - The first vector.
X;   type: vector, any type
X;
X; b - The second vector.
X;   type: vector, any type
X;
X;          ------------OUTPUT------------
X;
X; i - The union of a and b.
X;   type: vector, any type
X
X;;Howard Cohl
X;;Research Asst.
X;;National Solar Observatory
X;;Sunspot, NM
X;;88349
X
X;;hcohl@sunspot.noao.edu
X
Xfunction intersection,a,b
X
X  as=a(sort(a))
X  bs=b(sort(b))
X
X  loca=where(as ne shift(as,1))
X  locb=where(bs ne shift(bs,1))
X
X  as2=as(loca)
X  bs2=bs(locb)
X
X  c=[as2,bs2]
X
X  cs=c(sort(c))
```

```
X
X  locc=where(cs eq shift(cs,1))
X
X  i=cs(locc)
X
Xreturn,i
X
Xend
X
SHAR_EOF
chmod 0664 intersection.pro || echo "restore of intersection.pro fails"
sed 's/^X//' << 'SHAR_EOF' > intersect.pro &&
 X;------------------------------------------------------------ ---
X;+
X; NAME:
X;       INTERSECT
X; PURPOSE:
X;       Return the elements common to two given arrays.
X; CATEGORY:
X; CALLING SEQUENCE:
X;       z = intersect(x,y)
X; INPUTS:
X;       x, y = arrays (not necessarily same size).  in
X; KEYWORD PARAMETERS:
X; OUTPUTS:
X;       z = array of elements in common.            out
X; COMMON BLOCKS:
X; NOTES:
X;       Note: if z is a scalar 0 then no elements were
X;         in common.
X; MODIFICATION HISTORY:
X;       R. Sterner  19 Mar, 1986.
X; R. Sterner, 4 Mar, 1991 --- converted to IDL v2.
X;-
X;;  Ray Sterner                    sterner%str.decnet@warper.jhuapl.edu
X;;  Johns Hopkins University        North latitude 39.16 degrees.
X;;  Applied Physics Laboratory      West longitude 76.90 degrees.
X;;  Laurel, MD 20723-6099
 X;------------------------------------------------------------ ---
X
X function intersect,x,y, help=hlp
X
X if (n_params(0) lt 2) or keyword_set(hlp) then begin
X   print,' Return the elements common to two given arrays.'
X   print,' z = intersect(x,y)'
X   print,'   x, y = arrays (not necessarily same size).  in'
X   print,'   z = array of elements in common.            out'
X   print,' Note: if z is a scalar 0 then no elements were'
```

```
X   print,'   in common.'
X   return, -1
X endif
X
X xs = runlength(x(sort(x))) ; Keep only unique elements.
X ys = runlength(y(sort(y)))
X
X zs = [xs,ys]   ; Merge the 2 arrays.
X zs = zs(sort(zs))  ; Sort.
X
X d = zs - shift(zs,1)  ; Find differences between elements.
X
X w = where(d eq 0, count) ; Elements common to both arrays
X     ; occur twice, giving 0 diffs.
X
X if count eq 0 then return, 0 ; Scalar 0 means no common elements.
X return, zs(w)   ; Vector of common elements.
X
X end
X
SHAR_EOF
chmod 0664 intersect.pro || echo "restore of intersect.pro fails"
sed 's/^X//' << 'SHAR_EOF' > runlength.pro &&
 X;------------------------------------------------------------ ---
X;+
X; NAME:
X;       RUNLENGTH
X; PURPOSE:
X;       Give run lengths for array values.
X; CATEGORY:
X; CALLING SEQUENCE:
X;       y = runlength(x,[r])
X; INPUTS:
X;       x = 1-d array of values.                in
X; KEYWORD PARAMETERS:
X; OUTPUTS:
X;       y = X with multiple values squeezed out.  out
X;       r = run length of each element in Y.      out
X; COMMON BLOCKS:
X; NOTES:
X; MODIFICATION HISTORY:
X;       RES  30 Jan, 1986.
X; R. Sterner, 25 Sep, 1990 --- converted to IDL V2.
X;       Johns Hopkins University Applied Physics Laboratory.
X;-
X;;  Ray Sterner                    sterner%str.decnet@warper.jhuapl.edu
X;;  Johns Hopkins University        North latitude 39.16 degrees.
X;;  Applied Physics Laboratory      West longitude 76.90 degrees.
```

```
X;;  Laurel, MD 20723-6099
 X;------------------------------------------------------- ---
X
X FUNCTION RUNLENGTH,X,R, help=hlp
X
X if (n_params(0) lt 1) or keyword_set(hlp) then begin
X   print,' Give run lengths for array values.'
X   print,' y = runlength(x,[r])'
X   print,'   x = 1-d array of values.                in'
X   print,'   y = X with multiple values squeezed out.  out'
X   print,'   r = run length of each element in Y.      out'
X   return, -1
X endif
X
X ;---  The easiest way to understand how this works is to try
X ;---  these statements interactively.
X A = X - SHIFT(X,1) ; Distance to next value.
X A(0) = 1  ; Always want first value.
X W = WHERE(A NE 0) ; Look for value changes.
X Y = X(W)  ; Pick out unique values.
X IF N_PARAMS(0) LT 2 THEN RETURN, Y
X R = ([W,N_ELEMENTS(X)])(1:(N_ELEMENTS(Y))) - W  ; run lengths.
X RETURN, Y
X END
SHAR_EOF
chmod 0664 runlength.pro || echo "restore of runlength.pro fails"
exit 0
--
Ralph Finch   916-445-0088
rfinch@water.ca.gov  ...ucbvax!ucdavis!caldwr!rfinch
Any opinions expressed are my own; they do not represent the DWR
```