Subject: Re: remove duplicates WITHOUT sorting
Posted by Vince Hradil on Mon, 12 Feb 2007 17:23:21 GMT
View Forum Message <> Reply to Message

On Feb 12, 11:04 am, "rpert...@gmail.com" <rpert...@gmail.com> wrote:
> Hello All,
> I have an array of 2xn which is a list of coordinates (x,y positions)
> and I need to remove any duplicates in this array. Since they are x,y
> coordinates I cannot really sort the array, hence cannot use the Uniq
> function. Is there another way of doing this?
> My array is similar to:
> [2 30
> 4 5
> 3 9
> 51 19
> 12 15
> 3 9
> 11 22
> 32 33
> 14 25]
>
> As you see index 4,5 is same as index 10,11. How do i rewrite this
> array without duplicates? Is there a general way of doing this, as
> some of my coordinates (to make matters even more complicated) are not
> exactly the same, but are close (3,9) and (3.5,9) - But I think I can
> figure this out with some if...then statements. Not sure how to remove
> them though. Any help is appreciated!
> Thank you,
> Radha

How about this:

```
function uniq_coords, coords
cc = complex( reform(coords[0,*]), reform(coords[1,*]) )
cc = cc[uniq(cc,sort(cc))]
cc = transpose( [ [float(cc)],[imaginary(cc)] ] )
return, cc
```

Of course this will make the return value floats, but you can fix that
up to handle other types if you want.
Also, I have no idea how well this will perform - time to execute,
robustness, etc - just seemed to work on your test array. "caveat
emptor"

Subject: Re: remove duplicates WITHOUT sorting
Posted by Craig Markwardt on Mon, 12 Feb 2007 17:23:23 GMT

"rpertaub@gmail.com" <rpertaub@gmail.com> writes:
> Hello All,
> I have an array of 2xn which is a list of coordinates (x,y positions)
> and I need to remove any duplicates in this array. Since they are x,y
> coordinates I cannot really sort the array, hence cannot use the Uniq
> function. Is there another way of doing this?
> My array is similar to:
> [2 30
> 4 5
> 3 9
> 51 19
> 12 15
> 3 9
> 11 22
> 32 33
> 14 25]
>
> As you see index 4,5 is same as index 10,11. How do i rewrite this
> array without duplicates? Is there a general way of doing this, as
> some of my coordinates (to make matters even more complicated) are not
> exactly the same, but are close (3,9) and (3.5,9) - But I think I can
> figure this out with some if...then statements. Not sure how to remove
> them though. Any help is appreciated!

If your array is small enough, I would make an array of distances
between all the points and focus on the points which are closer than
your desired tolerance. Something like,

```
x = reform(data(0,*))
y = reform(data(1,*))
;; Form into square arrays, replicate in x and y direction
xx = x # (y*0 + 1)
yy = (x*0 + 1) # y
;; Compute difference in X and Y coordinate
dxx = xx - transpose(xx)
dyy = yy - transpose(yy)
;; Compute distance
drr = sqrt(dxx*dxx + dyy*dyy)
```

The matrix is symmetric, so if you are really smart you can reduce
your work by 50%. This starts to get ugly for more than a few
thousand points. In which case you would indeed need to sort them
into (overlapping) bands or squares, and work on one chunk at a time.

Good luck!
Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@REMOVEcow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: remove duplicates WITHOUT sorting
Posted by David Fanning on Mon, 12 Feb 2007 17:38:31 GMT
View Forum Message <> Reply to Message

rpertaub@gmail.com writes:

> I have an array of 2xn which is a list of coordinates (x,y positions)
> and I need to remove any duplicates in this array. Since they are x,y
> coordinates I cannot really sort the array, hence cannot use the Uniq
> function. Is there another way of doing this?

I don't have time to follow up myself, but there is an
extremely nifty method for removing elements from an array
using Histogram in the Histogram Tutorial. I think it would
be simple to apply it to this problem.

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: remove duplicates WITHOUT sorting
Posted by Brian Larsen on Mon, 12 Feb 2007 18:38:38 GMT
View Forum Message <> Reply to Message

In the spirit of histogram has anyone written code to turn the
REVERSE_INDICES stuff (which I can never seem to really get my head
around) into something less confusing like a really helpful structure
with tags i_vec and o_vec?

Just asking before I reinvent the wheel next time I need to use the
REVERSE_INDICES.

Brian

---------- ---------
Brian A. Larsen
Dept. of Physics
Space Science and Engineering Lab (SSEL)
Montana State University - Bozeman
Bozeman, MT 59717

Subject: Re: remove duplicates WITHOUT sorting
Posted by btt on Mon, 12 Feb 2007 19:23:51 GMT
View Forum Message <> Reply to Message

Brian Larsen wrote:
> In the spirit of histogram has anyone written code to turn the
> REVERSE_INDICES stuff (which I can never seem to really get my head
> around) into something less confusing like a really helpful structure
> with tags i_vec and o_vec?
>
> Just asking before I reinvent the wheel next time I need to use the
> REVERSE_INDICES.
>

Hi,

Geez, I know just what you mean. I put the whole mess into one function
and let it do the thinking. You might try the following which is just a
cooked down version of what is shown in the online help...

```
***BEGIN
;+
; NAME:
;   hist_index
;
; PURPOSE:
;   Use this routine to extract the ith bin elements in
;   REVERSE_INDICES as returned by the HISTOGRAM function.
;
; CALLING SEQUENCE:
;   result = hist_index(reverse_indices, i, [chunksize = value],[count =
variable])
;
; RETURNED VALUE: (lifted from IDL online manual)
;   Returns subscripts of the original array elements
;   falling in the ith bin.
;
;  If none are in that bin then -1 is returned.  Use COUNT to check.
;
; ARGUMENTS:
```

```
;  REVERSE_INDICES  See HISTOGRAM function description.
;       You may supply this as a vector or a pointer to a vector.
;   I set this equal to the ith bin from which to return the indices.
This maybe
;       a vector of bins (contiguous or otherwise).
;
; KEYWORDS:
;   CHUNKSIZE  Set equal to the number of elements used in each
;       chunking operation (default = 10000)  This is useful
;       for very large arrays.  Memory is allocated in chunks.
;       This maybe temporarily increased to more than chuncksize if
required.
;   COUNT  Set equal to a named variable to return the
;       number of elements in the returned value.
;       Set to 0 if returned value is -1.
;
; EXAMPLE:
;   get all of the pixels in the 10th bin
;IDL> img=read_png( filepath('mineral.png', SUBDIR=['examples','data']))
;IDL>  h = histogram(img, reverse_indices = r)
;IDL> index = hist_index(r,10, count= cnt)
;IDL> help, cnt, index
;CNT             LONG      =           25
;INDEX           LONG      = Array[25]
;
; COMMENT:
;  Take care to consider the number of bins available in histogram.
;  For example, in the example above there are 255 bins (it's a byte image)
;   but it is possible to get a real (but meaningless) result.
;   In the example below, fictious bin 300 is requested.
;IDL> index = hist_index(r,300, count= cnt)
;IDL> help, cnt, index
;CNT             LONG      =         2800
;INDEX           LONG      = Array[2800]
;
; MODIFICATION HISTORY;
; 25 MAY 2004, BT written
; 22 OCT 2004 BT added multiple bin requests with chunking
;       added error handling
;-

FUNCTION hist_index, r, i, $
    CHUNKSIZE = chunksize, $
    COUNT = count

COMPILE_OPT IDL2
ON_ERROR, 2
```

```
    ni = n_elements(i)
    ctr = 0
    If n_elements(chunkSize) EQ 0 then CS = 10000 else CS = chunksize[0]
    currentSize = CS
    index = lonarr(CS)


    If SIZE(r,/TYPE) EQ 10 then Begin
       ; a pointer to R
    For j = 0, ni-1 Do begin
       ;data supplied as a pointer to vector
    if (*r)[i[j]] NE (*r)[i[j]+1] Then begin
       idx = (*r)[(*r)[i[j]]:(*r)[i[j]+1]-1]
       count = SIZE(idx,/N_ELEMENTS)
          ;increase the size of the indexkeeper if needed
       If (ctr + count) GT currentSize Then Begin
          index = [index, lonarr(CS > count)]
          currentSize += (CS > count)
       EndIf
       index[ctr] = idx
       ctr += count
    EndIf

    EndFor

    Endif Else Begin

    For j = 0, ni-1 Do begin
       ; data supplied as a vector
    if r[i[j]] NE r[i[j]+1] Then begin
       idx = r[r[i[j]]:r[i[j]+1]-1]
       count = SIZE(idx,/N_ELEMENTS)
          ;increase the size of the indexkeeper if needed
       If (ctr + count) GT currentSize Then Begin
          index = [index, lonarr(CS > count )]
          currentSize += (CS > count)
       EndIf
       index[ctr] = idx
       ctr += count
    EndIf

    EndFor

    EndElse

    Count = ctr
If Count GT 0 then $
```

```
    Return, index[0:ctr-1] Else $
    Return, -1
END
***END
```

Subject: Re: remove duplicates WITHOUT sorting
Posted by JD Smith on Mon, 12 Feb 2007 21:15:41 GMT
View Forum Message <> Reply to Message

On Mon, 12 Feb 2007 09:04:45 -0800, rpertaub@gmail.com wrote:

> Hello All,
> I have an array of 2xn which is a list of coordinates (x,y positions)
> and I need to remove any duplicates in this array. Since they are x,y
> coordinates I cannot really sort the array, hence cannot use the Uniq
> function. Is there another way of doing this?

It's easiest to recast as 1D. HIST_2D (or HIST_ND) does this for you, but it's easy to do yourself:

```
 index=x + (max(x)+1)*y
```

and then using UNIQ on this list of indices should give you the row positions of unique coordinates. HISTOGRAM can work as well (either with HIST_ND, or by first constructing this index vector above), and it will be faster, but, as usual, will consume lots of memory (and potentially be very slow) if your coordinates are sparsely sprinkled over a large range of values (your current example is somewhat sparse, but not horrible). UNIQ, with its SORT based implementation, doesn't suffer from that issue.

JD

Subject: Re: remove duplicates WITHOUT sorting
Posted by rpertaub@gmail.com on Tue, 13 Feb 2007 15:18:42 GMT
View Forum Message <> Reply to Message

On Feb 12, 4:15 pm, JD Smith <jdsm...@as.arizona.edu> wrote:
> On Mon, 12 Feb 2007 09:04:45 -0800, rpert...@gmail.com wrote:
>> Hello All,
>> I have an array of 2xn which is a list of coordinates (x,y positions)
>> and I need to remove any duplicates in this array. Since they are x,y
>> coordinates I cannot really sort the array, hence cannot use the Uniq
>> function. Is there another way of doing this?
>

> It's easiest to recast as 1D. HIST_2D (or HIST_ND) does this for you, but
> it's easy to do yourself:
>
> index=x + (max(x)+1)*y
>
> and then using UNIQ on this list of indices should give you the row
> positions of unique coordinates. HISTOGRAM can work as well (either
> with HIST_ND, or by first constructing this index vector above), and
> it will be faster, but, as usual, will consume lots of memory (and
> potentially be very slow) if your coordinates are sparsely sprinkled over
> a large range of values (your current example is somewhat sparse, but not
> horrible). UNIQ, with its SORT based implementation, doesn't suffer from
> that issue.
>
> JD

Thank You All for responding.
I basically did sort in the end, but sorted the x coords and
corresponding y coords stayed with
its pair. Then did a shift to eliminate recurrences:

```
SortIndex=Sort(AllCoords[0,*])
for j=0,1 Do AllCoords[j,*] = AllCoords[j,sortindex]
Print,"Sorted All Cords is",AllCoords
print,' '


B=AllCoords - Shift(AllCoords,2)
print,' '
C=AllCoords[*,where(B[0,*] ge 0.5 or B[1,*] ge 0.6)]
csize=size(C,/dimensions)
print,'size c' ,csize[0],csize[1]
```

Subject: Re: remove duplicates WITHOUT sorting
Posted by Jean H. on Tue, 13 Feb 2007 16:10:14 GMT
View Forum Message <> Reply to Message

> Thank You All for responding.
> I basically did sort in the end, but sorted the x coords and
> corresponding y coords stayed with
> its pair. Then did a shift to eliminate recurrences:
>
> SortIndex=Sort(AllCoords[0,*])
> for j=0,1 Do AllCoords[j,*] = AllCoords[j,sortindex]
> Print,"Sorted All Cords is",AllCoords

> print,' '

You can have an X coord beeing repeated, but with different Y coord...

Jean H.

>
> B=AllCoords - Shift(AllCoords,2)
> print,' '
> C=AllCoords[*,where(B[0,*] ge 0.5 or B[1,*] ge 0.6)]
> csize=size(C,/dimensions)
> print,'size c' ,csize[0],csize[1]
>

Subject: Re: remove duplicates WITHOUT sorting
Posted by Vince Hradil on Wed, 14 Feb 2007 15:32:45 GMT
View Forum Message <> Reply to Message

On Feb 13, 9:18 am, "rpert...@gmail.com" <rpert...@gmail.com> wrote:
> On Feb 12, 4:15 pm, JD Smith <jdsm...@as.arizona.edu> wrote:
>
>
>
>> On Mon, 12 Feb 2007 09:04:45 -0800, rpert...@gmail.com wrote:
>>> Hello All,
>>> I have an array of 2xn which is a list of coordinates (x,y positions)
>>> and I need to remove any duplicates in this array. Since they are x,y
>>> coordinates I cannot really sort the array, hence cannot use the Uniq
>>> function. Is there another way of doing this?
>
>> It's easiest to recast as 1D. HIST_2D (or HIST_ND) does this for you, but
>> it's easy to do yourself:
>
>> index=x + (max(x)+1)*y
>
>> and then using UNIQ on this list of indices should give you the row
>> positions of unique coordinates. HISTOGRAM can work as well (either
>> with HIST_ND, or by first constructing this index vector above), and
>> it will be faster, but, as usual, will consume lots of memory (and
>> potentially be very slow) if your coordinates are sparsely sprinkled over
>> a large range of values (your current example is somewhat sparse, but not
>> horrible). UNIQ, with its SORT based implementation, doesn't suffer from
>> that issue.
>
>> JD

>
> Thank You All for responding.
> I basically did sort in the end, but sorted the x coords and
> corresponding y coords stayed with
> its pair. Then did a shift to eliminate recurrences:
>
> SortIndex=Sort(AllCoords[0,*])
> for j=0,1 Do AllCoords[j,*] = AllCoords[j,sortindex]
> Print,"Sorted All Cords is",AllCoords
> print,' '
>
> B=AllCoords - Shift(AllCoords,2)
> print,' '
> C=AllCoords[*,where(B[0,*] ge 0.5 or B[1,*] ge 0.6)]
> csize=size(C,/dimensions)
> print,'size c' ,csize[0],csize[1]

So did you try the cast-to-complex, recast-to-float approach?
