Subject: Array sorting by row
Posted by humphreymurray on Thu, 17 Aug 2006 08:47:31 GMT
View Forum Message <> Reply to Message

Hi,

In IDL, is there a way to independly sort the columns of a 2d matrix
without looping through and sorting each row individually?

Currenly I'm using:

```
for x = long(0), x_size - 1 do begin
   sorted_indexs[0,x] = sort(matrix[*,x])
endfor
```

For example, if the matrix contained the following values:

4 2 0 5
9 0 1 5
0 4 2 1
1 2 3 4

I want the result matrix to contain index's like:

2 1 0 3
1 2 3 0
0 3 2 1
0 1 2 3

Here, each column is sorted as if it's an independant vector.

Cheers.

Subject: Re: Array sorting by row
Posted by JD Smith on Tue, 22 Aug 2006 23:35:05 GMT
View Forum Message <> Reply to Message

Find below the routine SORT_ND, which efficiently sorts arrays of any
size along any arbitrary dimension, using the concepts discussed in
this thread. Finally HIST_ND has a friend...

JD

```
;+
; NAME:
;       SORT_ND
;
```

```
; PURPOSE:
;
;       Efficiently perform an N-dimensional sort along any dimension
;       of an array.
;
; CALLING SEQUENCE:
;
;       inds=sort_nd(array,dimension)
;
; INPUTS:
;
;       array: An array of at least 2 dimensions to sort.
;
;       dimension: The dimension along which to sort, starting at 1
;         (1:rows, 2:columns, ...).
;
; OUTPUTS:
;
;       inds: An index array with the same dimensions as the input
;         array, containing the (1D) sorted indices.  Can be used
;         directly to index the arary (ala SORT).
;
; EXAMPLE:
;
;       a=randomu(sd,5,4,3,2)
;       sorted=a[sort_nd(a,2)]
;
; SEE ALSO:
;
;       HISTOGRAM
;
; MODIFICATION HISTORY:
;
;       Tue Aug 22 15:51:12 2006, J.D. Smith <jdsmith@as.arizona.edu>
;
;  Written, based on discussion on c.l.i-p, 08/2006.
;
;-
 ;#####################################################################
####################
;
; LICENSE
;
;  Copyright (C) 2006 J.D. Smith
;
;  This file is free software; you can redistribute it and/or modify
;  it under the terms of the GNU General Public License as published
;  by the Free Software Foundation; either version 2, or (at your
```

```
;  option) any later version.
;
;  This file is distributed in the hope that it will be useful, but
;  WITHOUT ANY WARRANTY; without even the implied warranty of
;  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
;  General Public License for more details.
;
;  You should have received a copy of the GNU General Public License
;  along with this file; see the file COPYING.  If not, write to the
;  Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor,
;  Boston, MA 02110-1301, USA.
;
 ;#################################################################
####################

function sort_nd, array, dimension
  sz=size(array,/DIMENSIONS)
  ndim=n_elements(sz)
  s=sort(array)

  if dimension eq 1 then begin  ; mark along dimension with index
     inds=s/sz[0]
  endif else begin
     p=product(sz,/CUMULATIVE,/PRESERVE_TYPE)
     inds=s mod p[dimension-2]
     if dimension lt ndim then inds+=s/p[dimension-1]*p[dimension-2]
  endelse

  h=histogram(inds,REVERSE_INDICES=ri)
  ri=s[ri[n_elements(temporary(h))+1:*]]
  if dimension eq 1 then return,reform(ri,sz,/OVERWRITE) $
  else begin            ; target dimension is collected to front, rearrange it
     t=[dimension-1,where(lindgen(ndim) ne dimension-1)]
     ri=reform(ri,sz[t],/OVERWRITE)
     return,transpose(ri,sort(t))
  endelse
end
```