Subject: accessing reverse_indices
Posted by Jaco van Gorkom on Wed, 29 May 2002 11:28:56 GMT
View Forum Message <> Reply to Message

Hi all,

yesterday I got myself soo frustrated with the HISTOGRAM
function and with its awefulsome keyword REVERSE_INDICES,
that I decided to make this world a slightly better place by
writing a utility function to access the reverse indices.
I don't know about you all, but I confuse myself every
time I try the A[R[R[i]:R[i+1]-1]] approach. I much prefer
to get the subscripts of the elements in the 3rd bin of
the histogram by
  h = histogram(a, reverse_ind=r)
  bin3 = histobin(r, 3)
The really good stuff starts once you need to access a
number of bins: just call histobin with an array of bin
numbers. No more writing of awkward ad-hoc loops for me!
As an example, to set all the elements in bins of less
than 10 elements to zero:
  bins = where(h lt 10, cnt)
  if cnt ne 0 then a[histobin(r, bins)] = 0

cheers,
  Jaco

P.S. The code also contains the answer to a challenge put by
     JD in August last year, on expanding paired index ranges.
     Six lines of code, no histogram.

```
;+
; NAME:
; HISTOBIN
;
; PURPOSE:
; This function returns the reverse indices of specified
; histogram bins from an array as supplied by the
; REVERSE_INDICES keyword of HISTOGRAM.
;
; CATEGORY:
; Utilities
;
; CALLING SEQUENCE:
; Result = HISTOBIN(Reverse_indices, Bins)
;
; INPUTS:
; Reverse_indices: An array in the form as provided by the
```

```
;  REVERSE_INDICES keyword to the HISTOGRAM function.
;
; Bins: A scalar or array specifying the number(s) of the
;  histogram bin(s) (starting at zero) for which the reverse
;  indices are desired. Take care not to specify bin numbers
;  higher than the number of bins in the histogram.
;
; KEYWORD PARAMETERS:
; COUNT: On output, contains the number of indices returned. If
;  all specified bins are empty, COUNT will be set to zero and
;  the function returns -1L as result.
;
; OUTPUTS:
; This function returns the elements of the Reverse_indices array
; that belong to the specified histogram bins. See the IDL Online
; Help about the HISTOGRAM function for a description of the
; format of the reverse indices array. The result of this function
; has the same data type as Reverse_indices, which should normally
; be either 32-bit or 64-bit integer.
;
;
; EXAMPLE:
; Create the histogram of array A:
;  H = HISTOGRAM(A, REVERSE_INDICES = R)
;  ; Set all elements of A that are in the 3rd bin of H to 0.
;  A[ HISTOBIN(R, 2) ] = 0
;
; To discard all elements of A which are in a bin by themselves:
;  B = WHERE(H GT 2, Count)
;  IF COUNT NE 0 THEN A = A[ HISTOBIN(R, B) ]
;
; MODIFICATION HISTORY:
;  Written by: Jaco van Gorkom, 29 May 2002
;   gorkom@rijnh.nl / j.c.van.gorkom@fz-juelich.de
;-

function HISTOBIN, ri, bins, count=count

 compile_opt idl2

 ; Handle the case of only one scalar bin in the obvious way:
 if size(bins, /n_dimensions) eq 0 then begin
  if ri[bins] ne ri[bins+1] then begin
   result = ri[ri[bins]:ri[bins+1]-1]
   count = n_elements(result)
   RETURN, result
  endif else begin
   count = 0L
```

```
   RETURN, -1L  ; no elements found!
  endelse
 endif

 ; Now the general case of any number of bins:
 ; get the start and end index for the subranges in
 ; ri corresponding to each bin
 startind = ri[bins]
 endind = ri[bins+1] - 1
 ; check for empty bins
 notEmpty = where(endind ge startind, count)
 if count lt n_elements(startind) then begin
  if count eq 0 then $
    RETURN, -1L  ; no elements found!
  startind = startind[notEmpty]
  endind = endind[notEmpty]
 endif

 riType = size(ri,/type)
  ; the data type to be used for indexing (LONG or LONG64)

 ; expand the subranges to a full index vector into ri.
 ; Loopless, of course ;-)
 length = temporary(endind) - startind + 1
 cumLength = [0, $
  fix(total(temporary(length), /cumulative, /double), type=riType)]
 totalLength = cumLength[n_elements(cumLength)-1]
 indices = make_array(totalLength, /index, type=riType)
 helpind = value_locate(cumLength, indices)
 indices = $
  temporary(indices) + startind[helpind] - cumLength[helpind]

 ; return the requested elements of ri
 count = n_elements(indices)
 RETURN, ri[indices]
end
```
