Subject: Re: Incrementing an Array
Posted by John-David T. Smith on Fri, 27 Apr 2001 21:31:33 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:
>
> s007amf@news.wright.edu (ALAN FRAZIER) writes:
>
>> Once again I am struggling with the finer points of IDL. What I am trying
>> to do seems simple, but is causing me some headaches. What I am trying to
>> do is this:
>>
>> matrix[x,y,z] = matrix[x,y,z] + 1
>>
>> Where matrix is a 3D array and x,y,z are 1D arrays. Matrix starts out
>> all 0's and I am trying to increment values at elements [x,y,z]. The
>> problem
>> is that some elements should be incremented multiple times. All the
>> proper elements are incremented once, but the second, third,
>> fourth....increments are not showing up. Any ideas? I know that I could
>> write this with loops, but that would be too slow.
>
> This is a common error using array indices. The problem is that each
> element evaluated in parallel, not in sequence.
>
> You really want to use HISTOGRAM for this one. JD Smith wrote an
> n-dimensional histogramming function which you can find here:

I wasn't going to say anything because the answer to this one is in the
manual entry for histogram! Take a look. By the way a few minor bugs
in hist_nd were squashed. Attached is the latest incarnation. There's
really no need to use it for this problem though... simply transform
your indices into 1-d before using the method at the end of the manual
entry.

```
ind=x+nx*(y+ny*z)
matrix=HISTOGRAM(ind, INPUT=matrix, MIN=0, MAX=N_ELEMENTS(matrix)-1)
```

JD

```
;+
; NAME:
;  HIST_ND
;
; PURPOSE:
;
;       Perform an N-dimensional histogram, also known as the joint
;       density function of N variables, ala HIST_2D.
```

```
;
; CALLING SEQUENCE:
; hist=HIST_ND(V,BINSIZE,MIN=,MAX=,NBINS=,REVERSE_INDICES=)
;
; INPUTS:
;
; V: A NxP array representing P data points in N dimensions.
;
; BINSIZE: The size of the bin to use. Either a P point vector
;       specifying a separate size for each dimension, or a scalar,
;       which will be used for all dimensions.  If BINSIZE is not
;       passed, NBINS must be.
;
; OPTIONAL INPUTS:
;
;       MIN: The minimum value for the histogram.  Either a P point
;       vector specifying a separate minimum for each dimension, or a
;       scalar, which will be used for all dimensions.  If omitted,
;       the natural minimum within the dataset will be used.
;
;       MAX: The maximum value for the histogram.  Either a P point
;       vector specifying a separate maximmum for each dimension, or a
;       scalar, which will be used for all dimensions. If omitted, the
;       natural maximum within the dataset will be used.
;
;       NBINS: Rather than specifying the binsize, you can pass NBINS,
;       the number of bins in each dimension, which can be a P point
;       vector, or a scalar.  If BINSIZE it also passed, NBINS will be
;       ignored, otherwise BINSIZE will then be calculated as
;       binsize=(max-min)/nbins.  Note that *unlike* RSI's version of
;       histogram as of IDL 5.4, this keyword actually works as
;       advertised, giving you NBINS bins over the range min to max.
;
; KEYWORD PARAMETERS:
;
; MIN,MAX,NBINS: See above
;
;       REVERSE_INDICES: Set to a named variable to receive the
;       reverse indices, for mapping which points occurred in a given
;       bin.
;
; OUTPUTS:
;
;       The N-Dimensional histogram, of size N1xN2xN3x...xND where the
;       Ni's are the number of bins implied by the data, and/or
;       optional inputs min, max and binsize.
;
; OPTIONAL OUTPUTS:
```

```
;
; The reverse indices
;
; EXAMPLE:
;
; v=randomu(sd,3,100)
; h=hist_nd(v,.25,MIN=0,MAX=1,REVERSE_INDICES=ri)
;
; SEE ALSO:
;
;  HISTOGRAM, HIST_2D
;
; MODIFICATION HISTORY:
;
;       Fri Apr 20 12:57:34 2001, JD Smith <jdsmith@astro.cornell.edu>
;
;               Slight update to NBINS logic.  More aggressive keyword
;               checking.
;
;       Wed Mar 28 19:41:10 2001, JD Smith <jdsmith@astro.cornell.edu>
;
;  Written, based on HIST_2D, and suggestions of CM.
;
;-

function hist_nd,V,bs,MIN=mn,MAX=mx,NBINS=nbins,REVERSE_INDICES=ri
  s=size(V,/DIMENSIONS)
  if n_elements(s) ne 2 then message,'Input must be N(dimensions) x P (points)'
  if s[0] gt 8 then message, 'Only up to 8 dimensions allowed'

  if n_elements(mx) eq 0 then begin
     mx=make_array(s[0],TYPE=size(V,/TYPE))
     need_mn=n_elements(mn) eq 0
     if need_mn then mn=mx
     for i=0,s[0]-1 do begin
        mx[i]=max(V[i,*],MIN=tmn)
        if need_mn then mn[i]=tmn
     endfor
  endif

  if s[0] gt 1 then begin
     if n_elements(mn)    eq 1 then mn=replicate(mn,s[0])
     if n_elements(mx)    eq 1 then mx=replicate(mx,s[0])
     if n_elements(bs)    eq 1 then bs=replicate(bs,s[0])
     if n_elements(nbins) eq 1 then nbins=replicate(nbins,s[0])
  endif

  if n_elements(bs) eq 0 then begin
```

```
    if n_elements(nbins) ne 0 then begin
      nbins=long(nbins)       ;No fractional bins, please
      bs=float(mx-mn)/nbins
    endif else message,'Must pass either binsize or NBINS'
  endif else nbins=long((mx-mn)/bs)

  total_bins=nbins[s[0]-1]      ;Accumulate the size of all bins
  h=(nbins[s[0]-1]-1)<long((V[s[0]-1,*]-mn[s[0]-1])/bs[s[0]-1]) >0L
  for i=s[0]-2,0,-1 do begin
    ;; The scaled indices, s[n]+a[n-1]*(s[n-1]+a[n-2]*(s[n-2]+...
    h=nbins[i]*h+((nbins[i]-1)<long((V[i,*]-mn[i])/bs[i])>0L)
    total_bins=total_bins*nbins[i]
  endfor

  ret=make_array(TYPE=3,DIMENSION=nbins,/NOZERO)
  if arg_present(ri) then $
     ret[0]=histogram(h,min=0,max=total_bins-1,REVERSE_INDICES=ri ) $
  else $
    ret[0]=histogram(h,min=0,max=total_bins-1)
  return,ret
end
```

File Attachments

1) hist_nd.pro, downloaded 140 times