Subject: Re: I need some histogram magic - gridding very large dataset
Posted by John Correira on Thu, 02 May 2013 14:42:57 GMT
View Forum Message <> Reply to Message

On 05/02/2013 10:10 AM, Craig Markwardt wrote:
> I've run into similar problems. Perhaps the attached GEOSPLAT()
> function will do what you want. It does some trickery to avoid that
> extra bin at the end. It asks for a LON and LAT array, and also an
> EXPOsure array which can be any additive weighting for your points. In
> my field it's the amount of exposure time in seconds, but it can be
> anything, or just an array of 1's in order to compute a straight
> average. It also returns the total exposure array and a sample
> variance array. Craig

Below was my attempt at solving this problem. It allows for an arbitrary function to be applied to each bin. It's a work in progress. I haven't looked at it in awhile so I can't remember how well it worked the last time I tried to use it :-). Suggested improvements welcome.

Regards,

John

%%%%%%%%%%%%%%%%

```
FUNCTION cpi_2d_histobin, x, y, value, funct, $
  funct_arguments=funct_arguments, $
  BINSIZE=binsize, $
  MIN=mn, $
  MAX=mx, $
  NBINS=nbins, $
  emptybinvalue=emptybinvalue, $
  hdata=hdata, $
  x_size = x_size, $
  y_size = y_size, $
  debug=debug

  COMPILE_OPT IDL2, STRICTARRSUBS

  if ~KEYWORD_SET(debug) THEN BEGIN
    CATCH, theError
    IF theError NE 0 THEN BEGIN
      CATCH, /CANCEL
      print, '% '+!ERR_STRING
      return, -1
    ENDIF
  ENDIF
```

```
  if N_ELEMENTS(emptybinvalue) eq 0 then begin
    tname = size(value, /TNAME)
    if tname eq 'FLOAT' then emptybinvalue = !values.f_nan
    if tname eq 'DOUBLE' then emptybinvalue = !values.d_nan
  endif

  data = transpose([[x],[y]])

  imx = max(data,DIMENSION=2,MIN=imn)

  if n_elements(mx) eq 0 then mx=imx
  if n_elements(mn) eq 0 then mn=imn

  if ((N_ELEMENTS(binsize) GT 0) AND (N_ELEMENTS(nbins) GT 0)) THEN $
    print, ' BINSIZE and NBINS keywords conflicting. Ignoring NBINS.'

;  if (N_ELEMENTS(binsize) eq 0) then binsize = (mx-mn)/nbins
  h =
 hist_nd(data,binsize*(1+(MACHAR()).eps),MIN=mn,MAX=mx+binsiz e,REVERSE_INDICES=ri)


  nx = (size(h))[1]
  ny = (size(h))[2]

  x_size = (mx-mn+1)[0]/float(nx)
  y_size = (mx-mn+1)[1]/float(ny)

  val = h
  val *= emptybinvalue

  if N_ELEMENTS(funct_arguments) gt 0 then begin
    for i=0L, nx-1 do begin
      for j=0L, ny-1 do begin
        if h[i,j] gt 0 then begin
          ind = i+nx*j
          val[i,j] = CALL_FUNCTION(funct,value[ri[ri[ind]:ri[ind+1]-1]], $
            _STRICT_EXTRA=funct_arguments)
        endif else continue
      endfor
    endfor
  endif else if N_ELEMENTS(funct_arguments) eq 0 then begin
     for i=0L, nx-1 do begin
      for j=0L, ny-1 do begin
        if h[i,j] gt 0 then begin
          ind = i+nx*j
          val[i,j] = CALL_FUNCTION(funct,value[ri[ri[ind]:ri[ind+1]-1]])
          endif else continue
      endfor
```

```
  endfor
endif

val =  val[0:n_elements(val[*,0])-2,indgen(N_ELEMENTS(val[0,*])-1)]

if arg_present(hdata) then hdata = h

return, val

END
```