Subject: A new puzzle for histogram
Posted by gknoke on Fri, 15 Sep 2006 22:07:02 GMT

So, I've got this piece of code which is horribly horribly inefficient.
I know the solution lies in a clever application of the histogram function, but being Friday afternoon my brain isn't seeing it. Anyone else have any insight on how I might approach it?

This particular routine is mapping a piece of data from polar to cartesian coordinates. Currently the code generates sin/cos angle tables and calculates the x,y coordinates in meters for each range/angle, and then converts that to an x,y coordinate in terms of pixels from the center. I realize the calculation of the x,y coordinates can be replaced with a simple vector operation, but I can't see how to turn the separate resulting arrays for x and y into a single array I can use the histogram function to match to the mapped grid.

```
;Setup the output grid
map = fltarr(xysize, xysize)
count = intarr(xysize, xysize)

cos_table = cos(angles)
sin_table = sin(angles)

for j_theta = 0, n_elements(angles)-1 do begin
  for i_range = 0, n_range-1 do begin
    ;Calculate x and y in meters
    x = r_pts(i_range)*cos_table(j_theta)
    y = r_pts(i_range)*sin_table(j_theta)

    ;Find corresponding pixel on mapped grid
    ix = round((x-x0)/cellsize)+xysize/2
    jy = round((y-y0)/cellsize)+xysize/2

    ;If the pixel coord is inside the image put the data point there
    if(ix ge 0 and ix le xysize-1) then begin
      if(jy ge 0 and jy le xysize-1) then begin
        map(ix,jy)=map(ix,jy) + data(i_range,j_theta)
        count(ix,jy)=count(ix,jy)+1
      endif
    endif
  endfor
endfor ;End of nearest neighbor loops
```

Thanks,

--Greg

Subject: Re: A new puzzle for histogram
Posted by gknoke on Fri, 15 Sep 2006 23:41:04 GMT
View Forum Message <> Reply to Message

R.G. Stockwell wrote:
> Check out hist_nd
>
> http://www.dfanning.com/programs/hist_nd.pro
>
>
> Cheers,
> bob

I'm having a bit of an issue getting hist_nd to work. I guess I'm a little slow. So here's the hist_2d call I am using:

```
count = hist_2d(ix, jy, max1=(xysize-1),max2=(xysize-1),min1=0,min2=0)
```

I noticed hist_nd requires a single array organized as n_dim*n_points, so I did the following:

```
nx = n_elements(ix)
ix = reform(ix, nx)
jy = reform(jy, nx)
xy = intarr(2, nx)
xy[0] = ix & xy[1] = jy
count = hist_nd(xy, 1, min=0, max=39)
```

But I get drastically different results from what I had with hist_2d.
Any insight on what I'm doing wrong?

--Greg

Subject: Re: A new puzzle for histogram
Posted by JD Smith on Tue, 19 Sep 2006 17:50:27 GMT
View Forum Message <> Reply to Message

On Fri, 15 Sep 2006 16:41:04 -0700, gknoke wrote:

> R.G. Stockwell wrote:
>> Check out hist_nd
>>
>> http://www.dfanning.com/programs/hist_nd.pro
>>
>>
>> Cheers,
>> bob
>

> I'm having a bit of an issue getting hist_nd to work.  I guess I'm a
> little slow.  So here's the hist_2d call I am using:
>
> count = hist_2d(ix, jy, max1=(xysize-1),max2=(xysize-1),min1=0,min2=0)
>
> I noticed hist_nd requires a single array organized as n_dim*n_points,
> so I did the following:
> nx = n_elements(ix)
> ix = reform(ix, nx)
> jy = reform(jy, nx)
> xy = intarr(2, nx)
> xy[0] = ix & xy[1] = jy
> count = hist_nd(xy, 1, min=0, max=39)
>
> But I get drastically different results from what I had with hist_2d.
> Any insight on what I'm doing wrong?

Because xy[0]=ix only sets the first element.  This "trick" (using the
first in a range of indices when assigning a sub-array) only works when
the adjacent data are in memory order (along rows).  Perhaps:

```
 xy=[reform(ix,1,nx),reform(jy,1,nx)]
```

would work better.

JD