Subject: Re: Hist_nd 3D +1 gridding / binning data
Posted by Markus Schmassmann on Wed, 08 Mar 2017 16:26:46 GMT
View Forum Message <> Reply to Message

On 03/06/2017 11:42 PM, clement.feller@obspm.fr wrote:
> From images, I have assembled a large table (4 columns of
> single-precision floats and about 160 millions lines) - 3 independants
> variables and 1 quantity - which I will later use to perform the
> inversion of a radiative transfer model through MPFIT.
> Given hardware limitations, I sought to bin/resample/grid the data.
> Hence the following lines:
>
> density = hist_nd([col1, col2, col3, col4], nbins=50, $
> reverse_indices=ri) ;size(col1, /dimension) = [1,P]
> index = where(density ne 0, cts) ;finding non-empty bins
>
> newcol1 = fltarr(cts) ; a better way to allocate memory than density*0.
> newcol2 = newcol1
> newcol3 = newcol1
> newcol4 = newcol1
>
> for ijk=0L, (cts-1L) do begin
> init = ri[index[ijk]]
> stop = ri[index[ijk]+1L]-1L
> newcol1[ijk] = mean(col1[ri[init:stop]])
> newcol2[ijk] = mean(col2[ri[init:stop]])
> newcol3[ijk] = mean(col3[ri[init:stop]])
> newcol4[ijk] = mean(col4[ri[init:stop]])
> endfor
> ..... save data and move on to the next task
>
> It takes about 15-20 secs to do the hist_nd task using 4 threads on
> a Intel Core i5-3230M CPU (3rd gen) @ 2.60GHz, which is pretty awesome.
> But the averaging takes on a few hours, burning through all the cpu
> reserves.
>
> Since my initial data are images, I binned them down to a 512x512
> size (a fourfold reduction) and ended up with a table of 8.5 million lines
> instead.
> In this case, hist_nd takes less than a second and the averaging
> takes about 15 minutes.
>
> Do you have any advice, or have you ever tried to do that kind of
> taskin a different way ?

Hi Clement,

 for i=0,n-1 do SOMETHING

is faster than
```
   for i=0,n-1 do begin
      SOMETHING
   endfor
```

therefore you could try whether the following is faster:

```
newcol = fltarr(4,cts)
for ijk=0L, (cts-1L) do newcol[0,ijk]=$
        [ [mean(col1[ri[ri[index[ijk]]:ri[index[ijk]+1L]-1L]]),$
          [mean(col2[ri[ri[index[ijk]]:ri[index[ijk]+1L]-1L]]),$
          [mean(col3[ri[ri[index[ijk]]:ri[index[ijk]+1L]-1L]]),$
          [mean(col4[ri[ri[index[ijk]]:ri[index[ijk]+1L]-1L]]) ]
```

I hope that speeds up your task, but I don't know.
Good luck, Markus
