Subject: a behemoth bubble sort
Posted by fischertc13 on Mon, 29 Oct 2012 21:24:07 GMT
View Forum Message <> Reply to Message

Hi all,

I am currently frustrated trying to convert information from one data cube into another and could use some direction.

I have a 3-d modeling program which creates a spatial geometry datacube where each voxel in the geometry contains a velocity. Voxels outside the geometry are assigned an artificially high velocity that is set to be transparent in the modeling program. What I would like to do is to create a datacube with dimensions of x, y, and velocity from the spatial data cube of x,y, and z.

Unfortunately, the bubble sort I've employed in this code needs to run
through 20 billion+ data points for the program to complete, which is of
course impossible. Is there some way to simplify this? Also, is
there some way to select out the 'good' portion of the initial data cube
to apply the conversion to instead of the entire thing? You have posted on
array-juggling similar to this, though after reading the article I was not
able to apply the technique to my own problem. Any help would be much
appreciated!

Cheers,
Travis

My current codes is as follows:

----------

pro slice_run

restore,'nifscube.dat' ; restores spatial datacube 'nifs'

size = size(nifs,/dimensions)

nifs = long(nifs) ; turns velocities to integers

max = max(nifs) ; bad voxels are preset to artificially high maximum velocity

xsize = size[0]
ysize = size[1]
zsize = size[2]

good = where(nifs ne max, complement = bad)
; find where all true velocity data points exist,
; this is not employed elsewhere yet

```
nifs(bad) = 0 ; set bad pixels to zero velocity

min = min(nifs,max=max) ; find max/min true velocities

nifs(bad) = max+1 ; reset bad pixels out of velocity range

vsize = max-min ; set velocity space range

flux = fltarr(xsize,ysize,vsize) ; create new velocity data cube where z =
velocity

;giant for-loop that looks at each individual voxel at each velocity
;step and places the velocity at that voxel into the new cube's v-dimension.

for v = min,max-1 do begin ; v = velocity step

for x = 0,xsize-1 do begin
   for y = 0,ysize-1 do begin
      for z =0,zsize-1 do begin
         if (nifs(x,y,z) eq v) then begin ;if voxel has vth velocity step
            flux(x,y,v-min) = v ;places v at the vth plane of flux cube
         endif
      endfor
   endfor
endfor
endfor

end
-----
```

Cheers,
Travis