Subject: Find all points within a set of polygons
Posted by Maarten[1] on Wed, 30 Nov 2005 15:14:15 GMT
View Forum Message <> Reply to Message

Hi,

I'm trying to compare satellite images from two different polar sun-synchronous satellites. One dataset is high resolution (1 by 1 km, from MODIS/Aqua, if you're curious), the other has a lower spatial resolution (though still pretty high at about 15 by 15 km, from OMI/Aura).

I'd like to attach a number to each MODIS pixel telling me in which "low resolution" box it is located - allowing me to continue with histogram and its reverse indices to extract relevant averages from the MODIS data.

I have some code which tells me if a point falls within a specific box, but that would require a loop over all boxes and all pixels until a matching box is found - with ~ 99000 boxes in an orbit, things get tedious and slow very fast. I /think/ I can adapt this code to do either loop in one go, but not both loops at the same time - at least I have no idea where (no pun intended) to start.

The single point version of the poly-tester, returns true if the point [pnt_x,pnt_y] lies within the polygon described by the (x,y) pairs in poly_x and poly_y:

```
function POINT_IN_POLY, pnt_x, pnt_y, poly_x, poly_y
 compile_opt defint32, strictarr, strictarrsubs

 ii=0
 jj=n_elements(poly_x)-1
 cc=0B

 for ii=0,n_elements(poly_x)-1 do begin
  if ((poly_y[ii] le pnt_y) && (pnt_y lt poly_y[jj])) || $
   ((poly_y[jj] le pnt_y) && (pnt_y lt poly_y[ii])) $
  then $
   if pnt_x lt (poly_x[jj] - poly_x[ii]) * $
    (pnt_y - poly_y[ii]) / (poly_y[jj] - poly_y[ii]) + poly_x[ii] $
   then $
    cc = ~ cc
  jj = ii
 endfor
 return, cc
end
```

(taken from the C-version available here:
http://www.ecse.rpi.edu/Homepages/wrf/Research/Short_Notes/p npoly.html)

In my case all polygons are trapezoids, i.e. four points. Replacing the if statement with a where statement, I think I can get rid of the explicit loop over the pixels, but not the boxes. (I did try something, but it doesn't work quite yet).

I think I have enough memory to do it all at once for a pair over known overlapping data-sets.

Of course there are some simplifications in that both data-sets are ordered south to north (but not neatly or aligned in any way, it is just that the instruments fly in the same direction). And there are caveats: the coordinates are in (lon,lat) pairs, so the poles will probably go off the scale at some point, and the box-computation doesn't quite work over the dateline, but for now I don't care about that.

Any pointers or ideas for improvement are welcome.

Maarten
