Subject: Nearest Neighbor ... again!
Posted by Fabzi on Thu, 08 Jul 2010 13:11:37 GMT
View Forum Message <> Reply to Message

Hi everybody,

You probably have been asked many times, but once again this apparently simple problem is driving me crazy. The problem is famous:

I have a 2D grid defined by x1 (dim 2 array, for example lons) and y1 (dim2 array, for example lats). And I want to fit it to a second grid x2, y2. More precisely, I want to know the indexes in GRID1 that are the closest to each of my points in GRID2. The output of my function has then the same dimension as GRID2.

After using a long time the (very) inefficient "for loop":

```
-----
n2 = N_ELEMENTS(x2)
for i = 0l, n2 - 1 do begin
        quad = (x2[i] - x1)^2 + (y2[i]- y1)^2
        minquad = min(quad, p)
        if N_ELEMENTS(p) gt 1 then p = p[0] ; it happens.....
        out[i] = p
 endfor
---
```

I finaly found the best solution:

```
---
  n1 = n_elements(ilon)
  triangulate, x1, y1, c ; Compute Delaunay triangulation
  out = GRIDDATA(x1,y1, LINDGEN(n1), XOUT=x2, YOUT=y2, /NEAREST_N,
TRIANGLES =c)
---
```

Which is very fast.

However, I cannot find a quick solution to get the FOUR nearest points in my GRID1.

The stupid solution is (sorry for the very very ugly code):

```
---
      for i = 0l, n2 - 1 do begin
        quad = (x2[i] - x1)^2 + (y2[i]- y1)^2
        for j=0, 3 do begin
          minquad = min(quad, p)
```

```
      if N_ELEMENTS(p) gt 1 then p = p[0] ; it happens.....
      out[j,i] = p
      quad[p] = max(quad) * 2. ;dummy large distance
    endfor
  endfor
```

But I just cannot find a cleverer solution with triangulation...
Someone clever than me to help ?

Thanks a lot!

Fabz
