Subject: Re: vector layer comparison in IDL
Posted by Mark Hadfield on Thu, 03 Mar 2005 21:08:10 GMT
View Forum Message <> Reply to Message

yp wrote:
> Mark Hadfield wrote:
>> Are you interested, perhaps, in the maximum distance between
>> the two "coastlines"?
>
> You guessed it right. I got to calculate the absolute error
> (distance) assuming that one is *true* and the other is
> *estimated* from other sources.

Ok, so we have 2 polylines, the "true" one defined by vectors xt and yt, both dimensioned [n], and the "estimated" one defined by vectors xe and ye, both dimensioned [m]. We want to calculate the maximum distance of the "estimated polyline" from the "true" one. Here's a naive approach:

```
dmax = 0.
for i=0,m-1 do begin
   for j=1,n-1 do begin
      d = magic_distance_function(xe[i], ye[i], $
                                  xt[j-1], yt[j-1], xt[j], yt[j])
      if j eq 1 then dmin = d else dim = dmin < d
   endfor
   dmax = dmax > dmin
endfor
```

(I think I've got the logic right there. We're looping through the "estimated" vertices, for each one calculating the distance to the closest line segment on the "true" polyline.)

Now you might object that this solution is a little vague in the line where magic_distance_function is invoked. That's a good point. So here's an attempt at this function

```
function magic_distance_function, x, y, x0, y0, x1, y1

   compile_opt DEFINT32
   compile_opt STRICTARR
   compile_opt STRICTARRSUBS
   compile_opt LOGICAL_PREDICATE

   d0 = sqrt((x-x0)^2+(y-y0)^2)
   d1 = sqrt((x-x1)^2+(y-y1)^2)
   dp = 2.0*poly_area([x,x0,x1],[y,y0,y1]) / $
       sqrt((x1-x0)^2+(y1-y0)^2)
```

```
    return, dp > (d0 < d1)

end
```

This uses three relevant distances between point [x,y] and line segment [x0,y0] -> [x1,y1]. D0 and d1 are distances to the end points; dp is the perpendicular distance between [x,y] and the line through [x0,y0] & [x1,y1], extending the line as far as necessary. I'm too lazy to look up the expression for perpendicular distance, so I've taken advantage of the relationship between this distance and the area of the triangle formed by the 3 points, calculated by IDL function POLY_AREA.

Applying this to the surface of the earth is left as an exercise, as is testing.

You might want to think about the case where the "estimated" polyline follows the "true" polyline closely, but has extra vertices at one end or other. The above algorithm sees this as an error.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)