Subject: Re: triangulating over undefined space in irregular grids
Posted by bjelley on Thu, 27 Mar 2008 17:26:38 GMT
View Forum Message <> Reply to Message

On Mar 26, 3:10 pm, "ben.bighair" <ben.bigh...@gmail.com> wrote:
> On Mar 26, 3:57 pm, bjel...@worldwindsinc.com wrote:
>
>
>
>
>
>> On Mar 26, 11:48 am, Kenneth Bowman <k-bow...@tamu.edu> wrote:
>
>>> In article < 419932f8-47d6-4822-aa67-6f6e235ef...@n77g2000hse.googlegroup s.com >,
>
>>> bjel...@worldwindsinc.com wrote:
>>>> The setup: I am trying to contour plot a number of variables for a
>>>> storm surge grid that has up to 10 km resolution in the open Atlantic,
>>>> yet has 8 meter resolution in New Orleans area waterways. The grid
>>>> obviously includes most anything at or below sea level for the
>>>> Northwest Atlantic domain, but also includes many (200k or so) land-
>>>> based grid points (or it would be useless as a storm surge model). I
>>>> mention the land-based nodes to express that a coastline mask will not
>>>> fix the problem.
>
>>>> The problem: When plotting the results using the triangles returned,
>>>> the result includes vast areas that are outside of the model domain.
>>>> For example, the plot shows data in trangles running from central
>>>> Louisiana to New England, yet the grid does not include any land with
>>>> an elevation greater than 20 meters. So the grid does include some
>>>> land for a range of distances from the coast, from ~5 to ~100 km
>>>> inland based on elevation.
>
>>> I think your main problem is that TRIANGULATE computes the convex hull of
>>> the set of points. That is, there are no "bays" or concave regions in the
>>> resulting set of triangles. This produces the long narrow triangles that
>>> you see across the SE U.S.
>
>>>> Also of importance is that the grid is numbered and ordered in a
>>>> counter-clockwise fashion, not in any order of south to north and the
>>>> like. This is native to the model which uses a finite element method
>>>> for calculations allowing us to successfully resolve conservation of
>>>> momentum, velocity, etc at very high resolutions in areas of interest
>>>> while maintaining the ability to carry out computation at lower
>>>> resolutions over larger areas where mass conservation is
>>>> required...such as the Gulf of Mexico. Could I reorder for purposes of
>>>> plotting? Sure, but do I need to and where would it get me?
>

>>>> I have been through Dr. Bowman's book, Liam Gumley's book, David
>>>> Fanning's site, the astro site, the online help, and the German
>>>> library of IDL routines with no light shed on the solution.
>
>>>> I have fiddled with the tolerance variable passed to triangulate which
>>>> has not changed anything until I make it too large at which time...
>>>> % TRIANGULATE: Points are co-linear, no solution.
>
>>> You might also have some round-off problems, as your grid spacing spans
>>> 4 orders of magnitude 10 to 10^4 m. Are your inputs double precision?
>
>>> Your best bet may be to learn how to use object graphics polygon objects,
>>> but I'm afraid I can't help you there.
>
>>> Ken Bowman- Hide quoted text -
>
>>> - Show quoted text -- Hide quoted text -
>
>>> - Show quoted text -
>
>> Yes, I think that is correct. TRIANGULATE is producing the "convex
>> hull of the set of points"...in your words ;)
>
>> The inputs are all double precision and the data in the files are
>> carried out to enough places to avoid round off errors.
>> Example definition for node 1:" 1 -90.2350725000 30.4780242000
>> -7.9830000000"
>> [node long lat bathymetry]
>> Of course this is far beyond the precision of any measuring system,
>> but that does not matter at the moment.
>
>> The nodes at the corners of the triangles shown in the grid plot
>> (above) are defined in the grid file. I am going to try using that to
>> look for any side of a triangle that is not repeated, which means it
>> is on a boundary, and classify the nodes on either end of the line
>> (side of triangle) as a memebr of an array descibing FAULT_POLYGONS
>> for use in GRIDDATA.
>
>> I'll share how that goes.
>
> Hi,
>
> Wow! What a great problem!
>
> I think it is worth pursuing the FAULT_POLYGONS feature of GRIDDATA.
> I suggest that you run your points through GRID_INPUT first - these
> can be calculated once and then saved for reuse just like the
> triangulation can be saved.

>
> Out of enterprising ignorance I have occasionally introduced sample
> locations over undefined regions like land. To these locations I
> assign whatever the "missing" value is (like NaN). The nice thing
> about that is that the triangulation doesn't have to draw line
> segments that cross undefined regions (like Florida). But I never did
> that for a problem this scale.
>
> It will be very interesting to know how you resolve this - so I am
> glad you will share.
>
> Cheers,
> Ben- Hide quoted text -
>
> - Show quoted text -

An update:

I am working on the fault_polygons and fault_xy arrays. Some of the challenge is in connecting the beginning and end of some of the polygons...the islands are easy. Also working on perfecting the generation of polygons that do not erroneously connect islands to mainland. As much as the populace might wish for one, Cuba does not have a landbridge to Florida ;-)

Still perfecting all of that. A little polygon question: I can see that the fault_polygons can be a concatinated array describing points in x,y detailed in a 2-by-n array called fault_xy with n = number of total points in polygons. The question is, can fault_xy be the same concatinated array, or does it simply need to have all of the points in the polygons consecutively. In the online help, fault_xy is described as 2-by-n with n = number of points "that define the polygon". I am going to have dozens of ploygons and need hundreds more points in fault_xy than any single polygon contains.

I currently have 6000+ polygon points on 61 polygons. A plot and link forthcoming.

I must credit Jim P of ITT VIS for the first suggestion of the FAULT_POLYGONS in GRIDDATA method via an email response to this posting.

I expect that as soon as I figure this all out David Fanning will come on and show us an elegant 2 lines to handle it all. I like his map_gshhs_shoreline routine utilizing some of the logic needed here.

Last question for the moment: Is there a way to dictate a polygon of inclusion (the mainland and open Atlantic boundary) and also numerous

polygons of exclusion (islands)? That would be useful.