Geoscientific Model Development Discussions

Geosci. Model Dev. Discuss., referee comment RC1
https://doi.org/10.5194/gmd-2021-349-RC1, 2021

Comment on gmd-2021-349

Anonymous Referee #1

Referee comment on "TriCCo v1.1.0 – a cubulation-based method for computing connected components on triangular grids" by Aiko Voigt et al., Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2021-349-RC1, 2021

General comments:

This is a well-written manuscript with beautiful figures. I enjoyed reading it and learning about cubulation.

One major issue puzzles me. I apologize if I've just missed something obvious. The content of this review focuses on this issue. If I've missed something obvious, simply explain and that will be sufficient. If my point is substantial, then I'd like the revised manuscript to state clearly at the outset that alternative, much more efficient, methods exist to solve the problem of finding connected components of a triangulation.

Again, the text itself is well written, and even if I'm right that the particular problem of finding connected components is much more efficiently done using standard graph-theoretic means, I suggest the text be published to document this cubulation procedure in case it is of use in other contexts.

Specific comments:

1. According to Table 2, the most expensive part of the workflow by a large factor is the cubulation: hours vs seconds or minutes for the other steps. Given this cost, why is it better to cubulate the triangulation and then run a connected-components algorithm on the cubulation than simply to compute the connected components of the triangulation directly, using any of a number of standard approaches?

A standard approach is something like this, with details dependent on setting. Represent the triangulation by

* vertex positions: an nvertex x ndim matrix P and
* triangles: an ntriangle x 3 matrix T of indices into P.

From this representation, one can construct a triangle-triangle adjacency matrix G = (V,E) in time linear in the number of objects, with edges E determined by an adjacency rule, e.g. edge or vertex adjacency. (In the edge-rule case, the resulting graph is the dual graph defined in Defn. 2.1.) I want to emphasize that because this procedure has linear time, computing adjacency data from the raw triangulation is extremely fast, likely about as fast as the fastest entries in Table 2.

Now consider the cloud segmentation task. Each node v in V corresponds to a triangle. In the cloud segmentation task, remove v and its edges from G if the cloud fraction is below the threshold. One can then use a standard algorithm, such as breadth-first search, to find components in the resulting pruned graph. Each of these operations is again linear in the number of objects. Thus, here again this operation would be about as fast as the fastest entries in Table 2. See, e.g., https://en.wikipedia.org/wiki/Component_(graph_theory).

It is not clear to me what the computational complexity of your cubulation implementation is, nor what the optimal complexity is. But for the former, we can estimate from Table 2 as follows. The cost factor increase when going from 20km to 10km (4x more triangles) is 180min/12min = 15, just short of the 4^2 = 16 that would imply quadratic increase in cost. On the other hand, the factor is 9.6 when going from 40km to 20km: still superlinear (i.e., > 4) but less than quadratic. It's possible a constant cost dominates at lower resolution, and the 15x in the 20->10 jump shows that an asymptotic quadratic cost is eventually exposed.

In any case, it is clear that cubulation is expensive.

Thus, I pose to you this question: Why should we prefer cubulation followed by computing components rather than computing the components directly from the triangle-triangle adjacency graph?

2. As a smaller issue, the figures and text imply the triangulation needs to be geometrically regular so that the index scheme works. Is this a true constraint? That is, will this method not work for geometrically unstructured triangular grids? The graph-based method described above does not require regular triangulations, nor for that matter a fixed type of polygon.

Typos and grammar:
50: "we first introduce...and [-to] rigorously define"
Fig. 2: "in [+the] same colors"
300: "their cube coordinates differ[-s]"