Geosci. Model Dev. Discuss.,
doi:10.5194/gmd-2016-140-RC2, 2016

Geoscientific Model Development Discussions

Open Access

Interactive comment on "A numbering algorithm for finite elements on extruded meshes which avoids the unstructured mesh penalty" *by* Gheorghe-Teodor Bercea et al.

Anonymous Referee #2

Received and published: 22 August 2016

1 Context

The article presents a numbering strategy to take advantage of the vertically structured nature of meshes used in most geophysical applications, while using a unstructured "base mesh" in the horizontal. The idea is to amortize the cost of indirect addressing in the unstructured horizontal direction through direct addressing in the vertical in the innermost loop. Although the idea is not new, as was proposed in Macdonald et al. (2011) and others, the added value is the application to a class of higher order continuous and discontinuous finite element methods A careful analysis is provided on performance for first and second order schemes, as to how many vertical levels are required to amortize the cost of the indirect addressing.

2 Questions and suggestions

- I assume that λ defined as "number of extruded layers" stands for the number of vertices of a vertical mesh, rather than the number of segments in the vertical.

  In this case, looking at Equation 5: d_2 can be either 0 (for vertices) or 1 (for segments). Then the statement $0 \leq l \leq \lambda - d_2$ will let λ go out of bounds, unless the second $\leq$ symbol is replased by $<$. Similar observation for l-loop in Algorithm 3.

  In case this assumption is wrong, it should be made more clear what is meant with layers.

- The title suggests that besides a memory layout for function spaces, also a numbering strategy in the horizontal would be discussed.

- Looking at Figure 3., how is the numbering of $n+\#$ related to $m+\#$, and possibly nodes internal to the triangle? Are $m + \#$ and $n + \#$ related to different function spaces?

- Given the title I would have expected to see discussed what would be the impact (on e.g. performance) to number $n + \#$ (as in Figure 3) first in vertical fashion (zig-zag up-down, rather than right-left).

- It would help to see a visualisation of a practical numbering for a mesh of a few triangles and few levels, for a few function function space configurations, besides Algorithm 1.

- Algorithm 1 :

- First "$\mathsf{dofs}_{\mathsf{fs}}$" is assigned with round brackets, later with square brackets.
- The second last line makes reference to l, which is invalid outside the vertical loop.
- Perhaps exchanging loops l and d_2 can avoid the last 2 lines by looping l in $\{0, 1, ..., \lambda - 1 - d_2\}$? (possibly without changing the resulting order)

- Algorithm 3 :
  - Can I assume that for a single DG-DG cell-entity, $(\mathsf{dof}_0, \mathsf{dof}_1, ..., \mathsf{dof}_{k-1})$ are contiguous?
  - subscript fs in $L_{\mathsf{fs}}(v)$ missing
  - d_2 unassigned, should be referenced as subscript of V?

- Figure 5: Almost none of the results have reached as mentioned the "performance plateau" with 100 layers for the badly ordered mesh. It would be interesting to see how many layers are required for all discretizations to reach this plateau.

  Is there an expected benefit in going higher order to reach the plateau with less layers?