Curvilinear GMSH Reader

Aleksejs Fomins

1 Implementation

Curvilinear GMSH Reader Documentation - Principle of function:

1. Read all nodes from .msh file
2. Read all elements from .msh file

   - each element has *elm_type* which describes what sort of element it is, including the polynomial order

   - has a *physical_entity* tag, which describes its belonging to certain physical object

   - has other tags $\{element\_tag, partition\_tag\}$

   - Has set of DoF, which are the numbers of nodes corresponding to the element. Number of DoF's depends on *elm_type*
3. All DoF's are renumbered from GMSH convention to Dune convention

   - GMSH convention is nodes $\rightarrow$ edges $\rightarrow$ faces $\rightarrow$ element, all recursive

   - DUNE convention is $loop\ over\ z(\ loop\ over\ y(\ loop\ over\ x(\ point(x, y, z))))$
4. For each tetrahedron, define interpolation class, which gives its global curved coordinates based on local parametric coordinates and interpolation points.

   - the interpolation points are passed only once when initialising the class
5. parametrically Loop over all surfaces of the tetrahedron

   - Shrink the surfaces by a little bit for better visibility

   - Calculate a fixed number of equally spaced points on each face using square mesh, add all points to a map, which notes which points have already been added. This can be done either by using the interpolation class to get the interpolated points, or simply re-using the interpolation points not to invoke the interpolation process at all thus testing it.

- Split each square into 2 triangles and add them to the triangle list. If the square is at the edge, it produces only 1 triangle.

6. Write a .VTK file which has all triangles

1.1 Optimal Sampling Rate

Here we would like to discuss how many sampling points are required to represent the bounded curve using linear interpolation. This is necessary, for example, for visualisation. For a line segment, interpolating the curve bounded by 2 points, the analytic error is bounded by

$$\epsilon = |f(x) - g(x)| \leq \frac{1}{8} \max_{z \in [x_a, x_b]} \{g''(z)\}(x_a - x_b)^2 \tag{1}$$

where $f(x)$ is the line segment, $g(x)$ is the original function. x_a and x_b are arguments of the endpoints of the interval.

Given a large interval L which we split into N equal segments such that $x_b - x_a = L/N$, the maximal error over all interpolated intervals is bounded by

$$\epsilon_{\max} \leq \frac{L^2}{8N^2} \max_{z \in L} \{g''(z)\} \tag{2}$$

This can be rewritten to obtain the necessary interval number per edge

$$N = \left\lceil \alpha \sqrt{\max_{z \in L} \{g''(z)\}} \right\rceil \tag{3}$$

where α depends on precision and length of the interval. We observe that there is a problem with this formula, namely that if $g''(z) = 0$ we get 0 intervals, and if $g''(z)$ is small, we get 1 interval, and we would like 2, because there is at least some curvature, and it should be emphasized that the case is different from just straight line. Therefore we empirically modify the formula

$$N = \left\lceil \alpha \sqrt{\max_{z \in L} \{g''(z)\}} + 1 \right\rceil \tag{4}$$

Finally, since this equation is not exactly accurate for surfaces, and it is rather hard to make sense of optimal relative error, it is much easier to choose α empirically. We will generally operate with unit length edges ($L = 1$), and would like to, for example, have 5 points per edge with standard quadratic curvature $g(x) = x^2$. We obtain that $\alpha \approx 2.8$.

As a rule of thumb, we can calculate the expected number of intervals for a standard function $g(x) = \sum_i x^i$ using this formula. The first 5 orders will produce edge intervals $\{1, 5, 9, 17, 35\}$, which will correspond to the number of vertices per edge $\{2, 6, 10, 18, 36\}$, and thus the number of surface vertices $(i(i+1)/2)$ will be given by $\{3, 21, 55, 171, 666\}$.