

Intersection Mesh and Need for Mesh Rebase

Pavel Solin

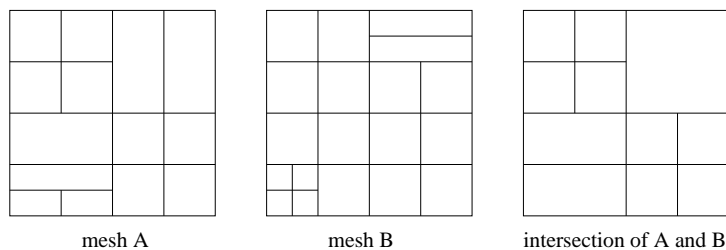
hp-FEM group, University of Nevada, Reno
Academy of Sciences, Prague, Czech Republic
<http://hpfem.org>
September 2009

Abstract

We describe what an intersection mesh means in the context of multimesh *hp*-FEM and why it is needed. We also explain why we need to develop a new functionality in the multimesh *hp*-FEM that we may call "mesh rebase".

1 Intersection mesh

Let us have a mesh M and a finite number of possible element refinement operations. Let A and B be two meshes obtained from M via independent sequences of these refinements. We say that M is a *submesh* of A and B . By *intersection of A and B* we mean the finest common submesh of A and B . This is illustrated in the following figure:



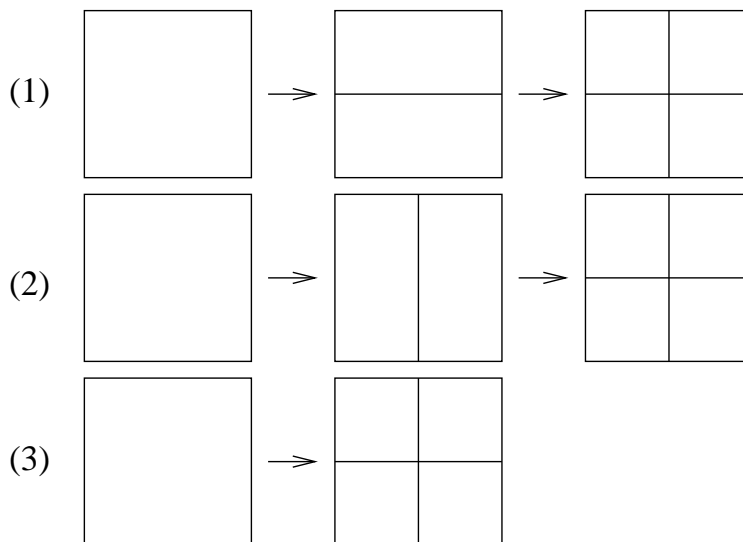
On the algorithmic side, the intersection mesh is obtained when descending simultaneously in the trees of the meshes A and B , and stopping whenever we are at the end of a branch in either mesh.

The motivation for creating the intersection mesh is in incompressible Navier-Stokes equations: When all velocity components are approximated using different meshes, and the corresponding intersection mesh is used for the pressure, then the Babuska-Brezzi (inf-sup) condition will be satisfied.

The construction of the intersection mesh is straightforward when only isotropic element refinements are done. Otherwise, there is a problem that we need to address. This is described in the following paragraph.

2 Incompatible refinement histories and the need for mesh rebase

In order to create the intersection mesh of meshes A and B, we need to be able to descend simultaneously the pairwise-corresponding branches in the refinement trees of A and B, always up to a point where the branches differ. This is no problem if only isotropic refinements are allowed, since in such case the refinement tree is unique. However, this is not the case when anisotropic refinements are used – we can have a variety of different trees representing the same mesh, as illustrated in the following figure:



Here (1), (2) and (3) represent different refinement histories leading to the same resulting mesh. Unfortunately, the nonuniqueness is a problem for the construction of the intersection mesh:

For example, denote by A and B the final meshes in the cases (1) and (2), respectively. Evidently, A and B are the same and thus the corresponding intersection mesh is the same as A and B. However, when descending the refinement trees, we have to stop already at the beginning, because there is no common branch in these two trees. In other words, the algorithm would deliver the initial mesh instead of the correct intersection mesh.

Hence we need to find a mechanism to collapse non-unique branches of mesh refinement histories. For example, the refinement trees (1) and (2) above would collapse to (3) which is a unique tree for the given final mesh.

Another reason for implementing the mesh rebase algorithm is that with a unique refinement history for each mesh in the system, we would be able to unrefine anisotropically-refined meshes in time-dependent problems, which we currently cannot do.