

How To Transfer Functions Between Meshes via Orthogonal Projections

Pavel Solin, *hp*-FEM group
University of Nevada, Reno
Academy of Sciences, Prague, Czech Republic
<http://hpfem.math.unr.edu>
September 2008

Assume a pair of meshes τ_1 and τ_2 which are covering the same computational domain Ω and moreover are the descendants of the same master mesh (in the sense of the multimesh *hp*-FEM). By V_1 and V_2 let us denote the finite element spaces defined by the meshes τ_1 and τ_2 , respectively. By N_1 and N_2 we denote the dimensions of these two spaces (the dimension means the number of degrees of freedom, or yet in other words the size of the stiffness matrix on that mesh). On τ_1 , we have some approximation U_1 which we want to transfer to τ_2 in the best possible way (so that as little information as possible is lost). Note that the transfer only can be lossless if $V_1 \subset V_2$, but typically this is not the case. Usually, $V_1 \not\subset V_2$ and thus by transferring functions we lose information.

This loss can be minimized if we use orthogonal projection of U_1 onto the space V_2 . This is done as follows: Let v_1, v_2, \dots, v_{N_2} be the finite element basis in V_2 (basis functions on the mesh τ_2). By U_2 let us denote the unknown best representant of U_1 in the space V_2 . Since U_2 lives in V_2 , we can express it as a linear combination of the basis functions in V_2 with unknown coefficients:

$$U_2 = \sum_{j=1}^{N_2} y_j v_j. \quad (1)$$

The minimum distance condition says that the difference $U_1 - U_2$ must be orthogonal to all basis functions of V_2 in the inner product of the space V_2 ,

$$(U_1 - U_2, v_i)_{V_2} = 0 \quad \text{for all } i = 1, 2, \dots, N_2. \quad (2)$$

(See Solin's Wiley book, Appendix A, Def. A.47 on page 402. Lemma A.39 on page 404 explains why orthogonal projection gives the best approximation in the target space.) Equation (2) can be written using the expansion (1) as

$$\left(U_1 - \sum_{j=1}^{N_2} y_j v_j, v_i \right)_{V_2} = 0 \quad \text{for all } i = 1, 2, \dots, N_2. \quad (3)$$

This is just

$$\sum_{j=1}^{N_2} y_j \underbrace{(v_j, v_i)_{V_2}}_{m_{ij}} = (U_1, v_i)_{V_2} \quad \text{for all } i = 1, 2, \dots, N_2, \quad (4)$$

where m_{ij} are the entries of the mass matrix M . Thus we obtained a matrix system $MY = F$, where the entries of the vector F have the form $f_i = (U_1, v_i)_{V_2}$. The solution of this matrix system are the expansion coefficients of the function U_2 on the target mesh τ_2 .

Note that the $N_2 \times N_2$ matrix M always is symmetric and positive definite (SPD), and thus easy to solve using iterative solvers such as preconditioned conjugate gradients. The inner product depends on the space where we are. For example, in $H^1(\Omega)$ we have

$$(u, v)_{V_2} = \int_{\Omega} \nabla u \cdot \nabla v + uv \, d\mathbf{x}.$$

The reason why we will need the multimesh functionality is that the functions U_1 and v_i on the right-hand side of (4) are defined on different meshes τ_1 and τ_2 , respectively.