Comput Visual Sci 6: 163–170 (2004)
Digital Object Identifier (DOI) 10.1007/s00791-003-0120-1

Computing and Visualization in Science

Regular article

MooNMD – a program package based on mapped finite element methods

Volker John[1], **Gunar Matthies**[2]

[1] Otto-von-Guericke-Universität Magdeburg, Institut für Analysis und Numerik, PF 4120, 39016 Magdeburg, Germany
(e-mail: john@mathematik.uni-magdeburg.de)
[2] Ruhr-Universität Bochum, Fakultät für Mathematik, Universitätsstr. 150, 44780 Bochum, Germany
(e-mail: Gunar.Matthies@ruhr-uni-bochum.de)

Received: 20 July 2002 / Accepted: 23 March 2003
Published online: 23 January 2004 – © Springer-Verlag 2004

Communicated by: M.S. Espedal, A. Quarteroni, A. Sequeira

Abstract. The basis of mapped finite element methods are reference elements where the components of a local finite element are defined. The local finite element on an arbitrary mesh cell will be given by a map from the reference mesh cell. This paper describes some concepts of the implementation of mapped finite element methods. From the definition of mapped finite elements, only local degrees of freedom are available. These local degrees of freedom have to be assigned to the global degrees of freedom which define the finite element space. We will present an algorithm which computes this assignment. The second part of the paper shows examples of algorithms which are implemented with the help of mapped finite elements. In particular, we explain how the evaluation of integrals and the transfer between arbitrary finite element spaces can be implemented easily and computed efficiently.

1 Introduction

Finite element methods are one of the most popular discretisation techniques for many classes of partial differential equations. There are two different ways to define finite elements - mapped and unmapped.

Mapped finite elements are closely connected to a standard way of analysing finite element methods. This standard analysis consists of three steps:

1.) Map an arbitrary mesh cell K to a reference mesh cell $\hat{K}$.
2.) Prove the desired properties on $\hat{K}$.
3.) Map the reference mesh cell $\hat{K}$ back to K to get the final result.

Step 2 is the core of this analysis. The two main features of this approach are the followings:

1.) All considerations have to be done on $\hat{K}$ only. (1)
2.) There are no information necessary and available about neighbour mesh cells of K. (2)

The unmapped finite element approach treats directly the mesh cell K. The definition of a reference mesh cell is not necessary. This approach is, to our knowledge, much more common in the implementation of finite element methods, e.g., [1].

Mapped and unmapped finite element methods posses the same analytical properties if the reference map $F_K : \overline{\hat{K}} \to \overline{K}$ is an affine map for every mesh cell K of the given triangulation. In the case of non-affine maps, occurring, e.g., for triangulations consisting of arbitrary quadrilateral or hexahedral mesh cells, mapped and unmapped finite elements might be different. The proof of the same analytical property may require different analytical tools, e.g., the proof of the inf-sup property for the so-called unmapped Q_2/P_1^{disc} pair of finite elements can be found already in [7] whereas the same property for the mapped Q_2/P_1^{disc} finite element has been established only recently in [3, 19].

The advantages of using mapped finite elements in computations arise from (1). All components which are necessary to define a certain finite element have to be implemented for a reference mesh cell only. These are, e.g., basis functions with their derivatives, positions of local degrees of freedom (d.o.f.), local nodal functionals or quadrature rules. During the execution of the program, these information are available from a data base. The challenge in the implementation of mapped finite elements comes from (2). There are many global finite element spaces whose functions have to fulfil continuity conditions across faces of mesh cells. This continuity will be given if the local degrees of freedom in each mesh cell K are associated appropriately to the global degrees of freedom which define the finite element space. We will present in Sect. 2 an algorithm which computes such a map from the local degrees of freedom to the global ones. This algorithm will be called d.o.f.-manager. It is part of the program package MooNMD (*M*athematics and *o*bject-*o*riented *N*umerics in *M*ag*D*eburg[1]) written in C++. One main feature of this program package is the strict separation of geometry and finite element data. Furthermore, all relevant mathematical objects like basis functions and nodal functionals are represented in MooNMD by C++ classes. This

[1] *MD* is the abbreviation of Magdeburg on the license plate of German cars.

program package has been successfully applied, e.g., in the solution of the incompressible steady state and time dependent Navier–Stokes equations with higher order finite element methods [9, 10, 16], for the large eddy simulation of turbulent flows [8, 11, 13], and for free boundary value problems with capillary surfaces [17, 18]. Some of its features are adaptive grid refinement [6, 15], isoparametric finite elements [10, 16] or mortar finite element methods [2].

Section 3 illustrates how mapped finite element methods can be used in the implementation of algorithms like the evaluation of integrals or the grid transfer between arbitrary finite element spaces. It is shown that these algorithms can be implemented easily and executed efficiently. Since nearly all computations in these algorithms are traced back to $\hat{K}$, a data base providing necessary information on $\hat{K}$ can be used. This data base contains both, information which are part of the code, like quadrature rules, as well as information which are computed during the run time, like local prolongation matrices.

2 Finite elements, nodal functionals, degrees of freedom and the d.o.f.-manager

The definition of finite elements, nodal functionals and degrees of freedom follows [4, 5]. A finite element in $\mathbb{R}^d$ is a triple (K, V_K, Σ_K). The set K is an open subset of $\mathbb{R}^d$ with Lipschitz-continuous boundary. The function space V_K has the finite dimension m. The set Σ_K which consists of m linear functionals N_i^K, $i = 1, \ldots, m$, defined on V_K is assumed to be V_K-unisolvent, i.e., for any given real numbers α_i, $i = 1, \ldots, m$, there exists a uniquely determined function $\varphi \in V_K$ which satisfies $N_i^K(\varphi) = \alpha_i$, $i = 1, \ldots, m$. The linear functionals N_i^K, $i = 1, \ldots, m$, are called local nodal functionals. Note that this definition of a finite element is similar to that given in [4, 5]. The main difference is that in [4, 5] the set K is considered to be a closed subset of $\mathbb{R}^d$. Defining finite elements on open mesh cells has the advantage that the finite element space can be defined on each cell separately without taking into account the definition on neighbouring cells and there are no conflicts in defining function values on cell boundaries. Furthermore, one can define one-sided limits which are used for jumps and averages on cell boundaries in an easy way.

Remark 1. Examples of local nodal functionals. For $k \geq 0$, the finite element spaces like the spaces P_k of piecewise polynomials of degree less than or equal to k and the spaces Q_k of mapped or unmapped piecewise polynomials of degrees less than or equal to k in each variable separately are equipped with local nodal functionals which use point values

$$N_i^K\left(v^h\right) = v^h|_K\left(\boldsymbol{x}_i\right),$$

where $\boldsymbol{x}_i \in \overline{K}$ is a given point. The local nodal functionals for the non-conforming, piecewise linear element P_1^{nc} and the two dimensional mean value oriented, rotated bilinear element Q_1^{rot} are given by integral mean values on the faces of K

$$N_i^K\left(v^h\right) = \frac{1}{|\partial K_i|}\int\limits_{\partial K_i} v^h|_K ds,$$

where ∂K_i is a face of K with $(d-1)$-dimensional measure $|\partial K_i|$. For the finite element spaces P_k^{disc}, $k \geq 1$, consisting of discontinuous functions which are mapped or unmapped piecewise polynomials of degree less than or equal to k, the local nodal functionals are given by weighted integral mean values on K.

One main ingredient in defining a finite element space is the correlation between local and global nodal functionals. Let N_i^K and $N_j^{K'}$ be two local nodal functionals which are associated with the cells K and K', respectively. Both local nodal functionals belong to the same global nodal functional if and only if

$$N_i^K(\varphi|_K) = N_j^{K'}(\varphi|_{K'}) \quad \forall\, \varphi \in C^\infty(U),$$

where U is an open subset of $\mathbb{R}^d$ with $\overline{K \cup K'} \subset U$.

Let $\mathcal{T}$ be a shape regular triangulation. A local degree of freedom on K will be denoted by (K, i) where i runs from 1 to the total number $\mathcal{N}(K)$ of local degrees of freedom. Furthermore, let

$$\mathcal{M}(K) := \{(K, i) : i = 1, \ldots, \mathcal{N}(K)\}$$

the set of all local degrees of freedom on K. The union

$$\mathcal{M} := \bigcup_{K \in \mathcal{T}} \mathcal{M}(K)$$

is the set of all local degrees of freedom.

With these notations, we can formulate the aim of the d.o.f.-manager in the following way:

Find a mapping $\mathcal{F} : \mathcal{M} \to \mathbb{N}$ such that

$$\mathcal{F}((K_1, i_1)) = \mathcal{F}((K_2, i_2))$$

if and only if (K_1, i_1) and (K_2, i_2) belong to same global degree of freedom.

Algorithm. Computation of the map which assigns the local degrees of freedom to the global degrees of freedom. Given a triangulation $\mathcal{T}$ with mesh cells $\{K\}$. The mesh cells are numbered by ascending integers. The identification number of K is denoted by $id(K)$.

```
1. for all mesh cells K ∈ T
2.   determine M(K)
3.   for all neighbours K' of K
4.     if id(K') < id(K)
5.       continue
6.     determine M(K')
7.     find partitioning of M(K) ∪ M(K')
8.     update the partitioning of M
9.   endfor
10. endfor
11. assign the partition members to increasing integers
```

This algorithm will be illustrated with the following example. The example shows in particular that Step 7 can be performed easily and efficiently.

Example 1. We consider a 2×2 mesh consisting of Q_1-finite elements, see Fig. 1. First, Steps 6 – 8 of the algorithm

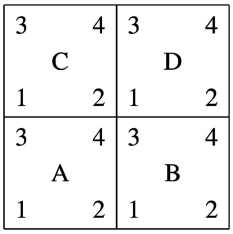

Fig. 1. 2×2 mesh consisting of Q_1-finite elements, initial local numbering

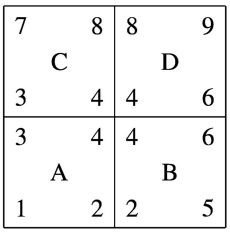

Fig. 2. 2×2 mesh consisting of Q_1-finite elements, final global numbering as result of the map $\mathcal{F}$

are described in detail. In the following, the notation $(Ki) \sim (Lj)$ means that the local degrees (Ki) and (Lj) belong to the same global degree of freedom.

1. *cell A with neighbour B*
   - partitioning of $\mathcal{M}(A) \cup \mathcal{M}(B)$:
     (A1),(A2) $\sim$ (B1),(A3),(A4) $\sim$ (B3),(B2),(B4)
   - partitioning of $\mathcal{M}$:
     (A1),(A2) $\sim$ (B1),(A3),(A4) $\sim$ (B3),(B2),(B4),
     (C1),(C2),(C3),(C4),(D1),(D2),(D3),(D4)
2. *cell A with neighbour C*
   - partitioning of $\mathcal{M}(A) \cup \mathcal{M}(C)$:
     (A1), (A2), (A3) $\sim$ (C1), (A4) $\sim$ (C2), (C3), (C4)
   - partitioning of $\mathcal{M}$:
     (A1), (A2) $\sim$ (B1), (A3) $\sim$ (C1), (A4) $\sim$ (B3) $\sim$ (C2),
     (B2), (B4), (C3), (C4), (D1), (D2), (D3), (D4)
3. *cell B with neighbour D*
   - partitioning of $\mathcal{M}(B) \cup \mathcal{M}(D)$:
     (B1), (B2), (B3) $\sim$ (D1), (B4) $\sim$ (D2), (D3), (D4)
   - partitioning of $\mathcal{M}$:
     (A1), (A2) $\sim$ (B1), (A3) $\sim$ (C1), (A4) $\sim$ (B3) $\sim$ (C2)
     $\sim$ (D1), (B2), (B4) $\sim$ (D2), (C3), (C4), (D3), (D4)
4. *cell C with neighbour D*
   - partitioning of $\mathcal{M}(C) \cup \mathcal{M}(D)$:
     (C1), (C2) $\sim$ (D1), (C3), (C4) $\sim$ (D3), (D2), (D4)
   - partitioning of $\mathcal{M}$:
     (A1), (A2) $\sim$ (B1), (A3) $\sim$ (C1), (A4) $\sim$ (B3) $\sim$ (C2)
     $\sim$ (D1), (B2), (B4) $\sim$ (D2), (C3), (C4) $\sim$ (D3), (D4)

Finally, Step 11 is performed, yielding

$$\mathcal{F}(A1) = 1,$$
$$\mathcal{F}(A2) = \mathcal{F}(B1) = 2,$$
$$\mathcal{F}(A3) = \mathcal{F}(C1) = 3,$$
$$\mathcal{F}(A4) = \mathcal{F}(B3) = \mathcal{F}(C2) = \mathcal{F}(D1) = 4,$$
$$\mathcal{F}(B2) = 5,$$
$$\mathcal{F}(B4) = \mathcal{F}(D2) = 6,$$
$$\mathcal{F}(C3) = 7,$$
$$\mathcal{F}(C4) = \mathcal{F}(D3) = 8,$$
$$\mathcal{F}(D4) = 9.$$

The result of the map $\mathcal{F}$ is illustrated in Fig. 2

Note that this algorithm does not provide a numbering which results in matrices with optimal bandwidth. However, there is no numerical algorithm implemented in the program package MooNMD which relies on a minimal bandwidth. In particular, linear systems of equations are solved by iterative schemes like Krylov subspace methods with multilevel preconditioners.

One fundamental object class in MooNMD is `TFESpace` which stores all necessary information for a certain finite element space. The description of a finite element space consists mainly of two arrays, `GlobalNumbers` and `BeginIndex`, which are generated by the d.o.f.-manager such that the global number $\mathcal{F}(K, i)$ of a local degree of freedom (K, i) is given by

$$\mathcal{F}(K, i) = \texttt{GlobalNumbers[BeginIndex[id(K)]+i]}.$$

For each cell K, the global numbers of the local degrees of freedom are stored in the array `GlobalNumbers` at consecutive entries starting with the index `BeginIndex[id(K)]`. Thus, the interplay of both arrays allows the evaluation of the mapping $\mathcal{F}$ between the local and global degrees of freedom in an efficient way.

Furthermore, an object of class `TFESpace` assigns the local function space V_K and the set Σ_K of local degrees of freedom to each cell K.

3 Implementation of some algorithms in the framework of mapped finite element methods

The reference cubes in MooNMD are $\hat{K} = (-1, 1)^d$, $d = 2, 3$, the reference triangle has the vertices (0, 0), (1, 0) and (0, 1) and the reference tetrahedron possesses the vertices (0, 0, 0), (1, 0, 0), (0, 1, 0) and (0, 0, 1). For the evaluation of integrals on edges of two-dimensional mesh cells, a one-dimensional reference mesh cell is needed, too. This is in MooNMD the interval $(-1, 1)$.

3.1 Evaluation of integrals

The numerical evaluation of integrals is a frequently used routine in finite element codes. It appears, e.g., in the assembling of stiffness matrices or in the computation of residual based a posteriori error estimators.

Let ϕ^h be a function whose integral on Ω has to be computed. The general approach to compute this integral starts by splitting it into a sum of integrals on the mesh cells. Next, the integral on each mesh cell is transformed to the reference mesh cell. Last, the integrals on the reference mesh cell are approximated by a quadrature rule. Let $J_K\left(\hat{\boldsymbol{x}}\right)$ be the Jacobian of the reference map F_K, and $\left(\hat{\boldsymbol{x}}_l, \theta_l\right), l = 1, \ldots, n$, a quadrature rule on the reference mesh cell with quadrature points $\hat{\boldsymbol{x}}_l$ and weights θ_l. Then, the computation of the integral of ϕ^h on Ω reads as follows

$$\int_{\Omega} \phi^h(\boldsymbol{x})\, d\boldsymbol{x} = \sum_{K\in\mathcal{T}_h} \int_K \phi^h(\boldsymbol{x})\, d\boldsymbol{x}$$
$$= \sum_{K\in\mathcal{T}_h} \int_{\hat{K}} \hat{\phi}^h(\hat{\boldsymbol{x}}) \left|\det J_K(\hat{\boldsymbol{x}})\right| d\hat{\boldsymbol{x}}$$
$$\approx \sum_{K\in\mathcal{T}_h} \left(\sum_{l=1}^{n} \theta_l \hat{\phi}^h(\hat{\boldsymbol{x}}_l) \left|\det J_K(\hat{\boldsymbol{x}}_l)\right| \right).$$

This approach requires the implementation of quadrature rules for reference mesh cells only. The quadrature rule on $\hat{K}$ is always chosen such that integrals of finite element functions over cells K which arise from affine transformations are evaluated exactly. Note that in this case the integrand on $\hat{K}$ is a polynomial if $\hat{\phi}^h$ is a polynomial.

Non-homogeneous Neumann boundary conditions, slip with friction and penetration with resistance boundary conditions, [12], or jump terms in residual based a posteriori error estimators require the computation of integrals on $(d-1)$-dimensional faces of mesh cells. Using the same way as above, these integrals are transformed to a $(d-1)$-dimensional reference cell and sufficiently accurate quadrature rules on the reference mesh cell are used for their evaluation.

3.2 Grid transfer in non-nested multilevel methods

The last example deals with multilevel methods where the discrete spaces might be non-nested. The main application of such a multilevel method in MooNMD is the multiple discretisation multilevel method for higher order finite element discretisations, see Fig. 3. Often, it is possible to compute more accurate results with less degrees of freedom and in less time if higher order finite element discretisations are used instead of low order ones. But the arising discrete systems are in general considerably harder to solve for higher order discretisations. The multiple discretisation multilevel method is a solver (or preconditioner in a Krylov subspace method) for systems coming from higher order finite element discretisations which exploits the efficiency of multigrid solvers for lowest order finite element discretisations. The main feature of this method is the use of two (or more) finite element discretisations on the finest geometric grid. One of them is the discretisation of interest which forms the highest level of the multilevel hierarchy. The other one is a lowest order discretisation. On each coarser geometric level, only a lowest order discretisation is applied. If necessary, all discretisations are stabilised, e.g., for convection dominated problems.

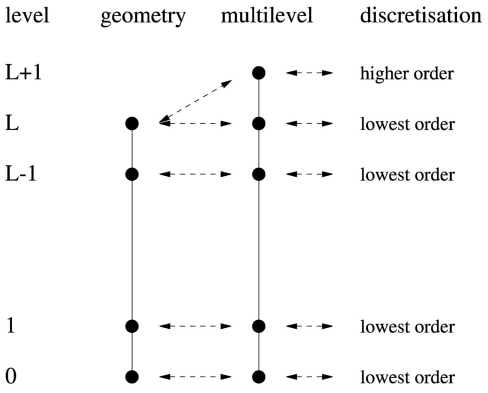

Fig. 3. The multiple discretisation multilevel method

The efficiency of the multiple discretisation multilevel method has been demonstrated in computations for the 2d and 3d Navier–Stokes equations, [9, 10, 16], and the convergence for the W-cycle in the solution of the Stokes equations has been proved in [14]. For such saddle point problems, it is advisable to choose lowest order non-conforming spaces for the velocity since these spaces fulfil the inf-sup condition with piecewise constant pressure approximations. But, one obtains a hierarchy of non-nested velocity spaces. One has to define a grid transfer between the non-conforming spaces on level l and $l+1$, $0 \le l < L$ and between a non-conforming space on level L and an arbitrary space on level $L+1$ which are defined on the same triangulation.

3.2.1 The function prolongation. We will describe in detail the implementation of a prolongation operator in MooNMD, [20], which allows the prolongation between almost arbitrary finite element spaces.

We consider the transfer (prolongation) from a finite element space V_{l-1}^h to a finite element space V_l^h. Let $\mathcal{T}_{l-1}$ and $\mathcal{T}_l$ be the corresponding triangulations of the domain Ω such that $\mathcal{T}_l$ originates either from a refinement of $\mathcal{T}_{l-1}$ or $\mathcal{T}_{l-1} = \mathcal{T}_l$. The second case is relevant in the multiple discretisation multilevel method for $l = L+1$, see Fig. 3.

Let Σ_l^h be a discontinuous finite element space defined on $\mathcal{T}_l$

$$\Sigma_l^h = \left\{ w \in L^2(\Omega) : w|_K \in S_l^h(K)\ \forall K \in \mathcal{T}_l \right\}.$$

The choice of the local spaces $S_l^h(K)$ depends on V_{l-1}^h and V_l^h. It has to be done such that the inclusion

$$V_{l-1}^h + V_l^h \subset \Sigma_l^h \tag{3}$$

holds. From the practical point of view, the spaces $S_l^h(K)$ are not needed for implementing the transfer operator. From the theoretical point of view, it can be proved that appropriate spaces $S_l^h(K)$ always exist for triangulations consisting of simplices, see [14].

The transfer operator is based on the concept of nodal functionals. For each mesh cell $K \in \mathcal{T}_l$ and for the finite element space Σ_l^h there exist a local finite element basis $\left\{ \psi_{l,j}^h|_K \right\}$ and a dual basis $\left\{ N_{l,j}^K \right\}$ of local nodal functionals such that

$$N_{l,j}^K\left(\psi_{l,i}^h|_K\right) = \delta_{ij}, \quad 0 \le i, j \le \dim\left(S_l^h(K)\right),$$

where δ_{ij} is the Kronecker delta.

Let $\left\{\varphi_{l,j}^h\right\}$ be a finite element basis of V_l^h. The indices j are called nodes or degrees of freedom. The set of all nodes of V_l^h is denoted by $I_l\left(V_l^h\right)$. The set of local nodes with respect to the mesh cell K is given by

$$I_l\left(K, V_l^h\right) = \left\{ i \in I_l\left(V_l^h\right) : \operatorname{supp}\left(\varphi_{l,i}^h\right) \cap K \neq \emptyset \right\}. \tag{4}$$

Furthermore, we define for any node $j \in I_l\left(V_l^h\right)$

$$\mathcal{T}_{l,j} = \left\{ K \in \mathcal{T}_l : j \in I_l\left(K, \Sigma_l^h\right) \right\},$$

the set of all mesh cells which are connected to the node j. Then, the global nodal functional which is associated with a node $j \in I_l\left(V_l^h\right)$ and whose argument is a function $w^h \in \Sigma_l^h$ is defined by the arithmetic mean of local nodal functionals

$$N_{l,j}\left(w^h\right) = \frac{1}{\operatorname{card}\left(\mathcal{T}_{l,j}\right)} \sum_{K \in \mathcal{T}_{l,j}} N_{l,j}^K\left(w^h|_K\right), \quad w^h \in \Sigma_l^h,$$

where $\operatorname{card}\left(\mathcal{T}_{l,j}\right)$ denotes the number of mesh cells in $\mathcal{T}_{l,j}$. The transfer operator for the prolongation is defined with the help of the global nodal functionals:

$$P_{l-1}^l : \Sigma_l^h \to V_l^h, \quad P_{l-1}^l\left(v^h\right) = \sum_{i=1}^{\dim\left(V_l^h\right)} N_{l,i}\left(v^h\right) \varphi_{l,i}^h. \tag{5}$$

From the inclusion (3) follows that this operator is defined especially for functions from V_{l-1}^h.

Let $\left\{\varphi_{l-1,i}^h\right\}$ be a finite element basis of V_{l-1}^h and

$$w_{l-1}^h = \sum_{i=1}^{\dim\left(V_{l-1}^h\right)} w_{l-1,i} \varphi_{l-1,i}^h \in V_{l-1}^h.$$

For evaluating the coefficient of $\varphi_{l,i}^h$ for the prolongated function, one has to compute

$$\begin{aligned} & N_{l,i}\left(w_{l-1}^h\right) \\ &= \frac{1}{\operatorname{card}\left(\mathcal{T}_{l,i}\right)} \sum_{K \in \mathcal{T}_{l,i}} N_{l,i}^K\left(w_{l-1}^h|_K\right) \\ &= \frac{1}{\operatorname{card}\left(\mathcal{T}_{l,i}\right)} \sum_{K \in \mathcal{T}_{l,i}} \sum_{j=1}^{\dim\left(V_{l-1}^h\right)} w_{l-1,j} N_{l,i}^K\left(\varphi_{l-1,j}^h|_K\right). \end{aligned}$$

We will give now some concrete examples how to compute the local nodal functionals. For simplicity of presentation, we restrict ourselves to two-dimensional finite elements. Let $\mathcal{P}(K) \in \mathcal{T}_{l-1}$ be the parent mesh cell of $K \in \mathcal{T}_l$. The local degrees of freedom of $\mathcal{P}(K)$ are represented by balls in the following figures and the local degrees of freedom of K by squares.

• *Red refined triangles,* $P_1(\mathcal{P}(K)) \to P_1(K)$. We consider the two situations given in Fig. 4. For this finite element, the local nodal functionals are point values, i.e. $N_{l,i}^K\left(\varphi_{l-1,j}^h|_K\right)$ is the value of $\varphi_{l-1,j}^h$ at the position of the local degree of freedom i in K. One obtains the following values

Fig. 4, left	$\varphi_{l-1,0}^h\|_K$	$\varphi_{l-1,1}^h\|_K$	$\varphi_{l-1,2}^h\|_K$
$N_{l,0}^K$	1	0	0
$N_{l,1}^K$	0.5	0.5	0
$N_{l,2}^K$	0.5	0	0.5

Fig. 4, right	$\varphi_{l-1,0}^h\|_K$	$\varphi_{l-1,1}^h\|_K$	$\varphi_{l-1,2}^h\|_K$
$N_{l,0}^K$	0.5	0.5	0
$N_{l,1}^K$	0	0.5	0.5
$N_{l,2}^K$	0.5	0	0.5

It turns out for the prolongation that this is just the standard inclusion.

• *Red refined triangles,* $P_1^{nc}(\mathcal{P}(K)) \to P_1^{nc}(K)$. We consider again two situations, see Fig. 5. In this case, the local nodal functionals are given by integral mean values at the edges of K, e.g.,

$$N_{l,0}^K\left(\varphi_{l-1,j}^h|_K\right) = \frac{1}{\|\boldsymbol{x}_0 - \boldsymbol{x}_1\|} \int_{\boldsymbol{x}_0}^{\boldsymbol{x}_1} \varphi_{l-1,j}^h|_K ds.$$

One obtains for the local nodal functionals

Fig. 5, left	$\varphi_{l-1,0}^h\|_K$	$\varphi_{l-1,1}^h\|_K$	$\varphi_{l-1,2}^h\|_K$
$N_{l,0}^K$	1	−0.5	0.5
$N_{l,1}^K$	0.5	0	0.5
$N_{l,2}^K$	0.5	−0.5	1

Fig. 5, right	$\varphi_{l-1,0}^h\|_K$	$\varphi_{l-1,1}^h\|_K$	$\varphi_{l-1,2}^h\|_K$
$N_{l,0}^K$	0.5	0.5	0
$N_{l,1}^K$	0	0.5	0.5
$N_{l,2}^K$	0.5	0	0.5

Applying these local nodal functionals in the prolongation operator (5), one gets a standard averaging operator, see Fig. 6. In this figure, the square denotes the degree of freedom of V_l^h whose value has to be computed and the balls stand for the nodes of V_{l-1}^h. The numbers give the weights which have to be applied to the coefficients of the function from V_{l-1}^h corresponding to these nodes. It is easy to see that the prolon-

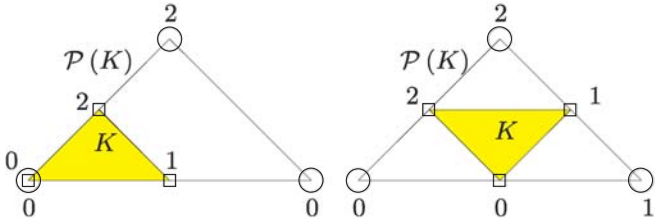

Fig. 4. Red refined triangles, $P_1(\mathcal{P}(K)) \to P_1(K)$

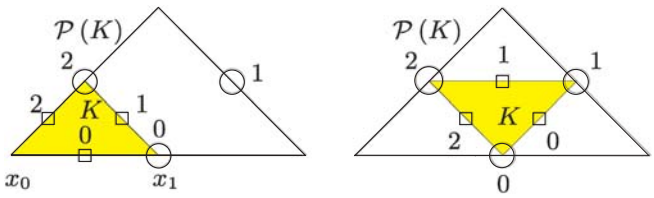

Fig. 5. Red refined triangles, $P_1^{nc}(\mathcal{P}(K)) \to P_1^{nc}(K)$

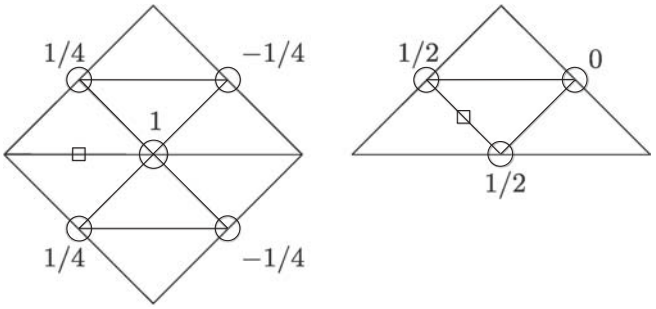

Fig. 6. Red refined triangles, the weights in the prolongation for $P_1^{nc}(\mathcal{P}(K)) \to P_1^{nc}(K)$

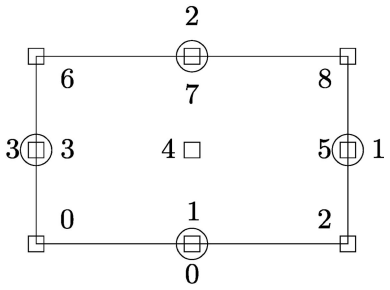

Fig. 7. No refined mapped quadrilaterals, $Q_1^{rot}(\mathcal{P}(K)) \to Q_2(K)$

gated value in the left picture of Fig. 6 is just the average in this point of the values of the finite element function of V_{l-1}^h restricted to the triangles in $\mathcal{T}_{l-1}$.

• *No refined mapped quadrilaterals,* $Q_1^{rot}(\mathcal{P}(K)) \to Q_2(K)$. In this case, we have $\mathcal{P}(K) = K$. A basis of $Q_1^{rot}\left(\hat{K}\right)$ is given by

$$\begin{aligned}&\left\{\hat{\varphi}_{l-1,0}^h, \hat{\varphi}_{l-1,1}^h, \hat{\varphi}_{l-1,2}^h, \hat{\varphi}_{l-1,3}^h\right\}\\ &= \left\{-\frac{3}{8}\left(\hat{x}_1^2 - \hat{x}_2^2\right) - \frac{1}{2}\hat{x}_2 + \frac{1}{4}, \frac{3}{8}\left(\hat{x}_1^2 - \hat{x}_2^2\right) + \frac{1}{2}\hat{x}_1 + \frac{1}{4},\right.\\ &\quad \left.-\frac{3}{8}\left(\hat{x}_1^2 - \hat{x}_2^2\right) + \frac{1}{2}\hat{x}_2 + \frac{1}{4}, \frac{3}{8}\left(\hat{x}_1^2 - \hat{x}_2^2\right) - \frac{1}{2}\hat{x}_1 + \frac{1}{4}\right\}.\end{aligned}$$

It is straightforward to check that

$$\frac{1}{|\hat{E}_i|}\int_{\hat{E}_i} \hat{\varphi}_{l-1,j}^h ds = \delta_{ij},$$

where the edges $\hat{E}_i$ of $\hat{K}$ are numbered counter clockwise, starting with the bottom edge. Let K be an arbitrary mesh cell and let $\left(\hat{x}_1, \hat{x}_2\right) \in \overline{\hat{K}}$ be transformed to $(x_1, x_2) \in \overline{K}$ such that $\hat{\varphi}^h\left(\hat{x}_1, \hat{x}_2\right) = \varphi^h(x_1, x_2)$. Thus, the values of the transformed basis functions can be easily computed by values of the reference basis functions in $\overline{\hat{K}}$. The reference transformation leads to a situation as presented in Fig. 7. The local nodal functionals of $Q_2(K)$ are defined as point values of the local basis functions of $Q_1^{rot}(K)$. As pointed out, the evaluation of these point values can be done in $\overline{\hat{K}}$ which gives, independent of K,

	$\varphi_{l-1,0}^h\|_K$	$\varphi_{l-1,1}^h\|_K$	$\varphi_{l-1,2}^h\|_K$	$\varphi_{l-1,3}^h\|_K$
$N_{l,0}^K$	3/4	−1/4	−1/4	3/4
$N_{l,1}^K$	9/8	−1/8	1/8	−1/8
$N_{l,2}^K$	3/4	3/4	−1/4	−1/4
$N_{l,3}^K$	−1/8	1/8	−1/8	9/8
$N_{l,4}^K$	1/4	1/4	1/4	1/4
$N_{l,5}^K$	−1/8	9/8	−1/8	1/8
$N_{l,6}^K$	−1/4	−1/4	3/4	3/4
$N_{l,7}^K$	1/8	−1/8	9/8	−1/8
$N_{l,8}^K$	−1/4	3/4	3/4	−1/4

These examples show that the values of the local nodal functionals $N_{l,i}^K\left(\varphi_{l-1,j}^h|_K\right)$ are, in general, the same for a large number of mesh cells. The values can be computed in a preprocessing step and stored in the data base. In computing the prolongation, only local matrix-vector products have to be performed with these values. This strategy is used to accelerate the computation of the prolongation (5).

An algorithm for computing the prolongation (5) for a function $w_{l-1} \in V_{l-1}^h$ looks as follows.

Algorithm. Prolongation. Given the coefficient vector $\mathtt{w}_{l-1}$ of the finite element function $w_{l-1}^h \in V_{l-1}^h$, determine the coefficient vector $\mathtt{w}_l$ of $P_{l-1}^l(w_{l-1}^h) \in V_l^h$.

```
1. for (i := 0; i < dim(V_1^h); i++)
2.   w_1(i) = 0
3.   card(i) = 0
4. endfor
5. for K ∈ T_1
6.   for i ∈ I_1(K, V_1^h)
7.     for (j := 0; j < dim(V_{1-1}^h); j++)
8.       if supp(φ_{1-1,j})^h|_K ∩ K = ∅
9.         continue
10.      w_1(i) := w_1(i) + w_{1-1}(j) N_{1,i}^K (φ_{1-1,j}^h|_K)
11.      card(i) := card(i) + 1
12.    endfor
13.  endfor
14. endfor
15. for (i := 0; i < dim(V_1^h); i++)
16.   w_1(i) := w_1(i) / card(i)
17. endfor
```

3.2.2 The defect restriction. The definition of the operator for the defect restriction $R_l^{*,l-1} : \left(V_l^h\right)^* \to \left(V_{l-1}^h\right)^*$ uses the prolongation operator given in (5). Let $d_l \in \left(V_l^h\right)^*$ be a given defect functional, its restriction to $\left(V_{l-1}^h\right)^*$ is defined by

$$\int_\Omega R_l^{*,l-1}(d_l)\,\varphi_{l-1}^h d\boldsymbol{x} = \int_\Omega d_l P_{l-1}^l\left(\varphi_{l-1}^h\right) d\boldsymbol{x}$$

for all $\varphi_{l-1}^h \in V_{l-1}^h$. This is the standard definition and since the prolongation operator turns out to be standard in many situations, the same holds for the defect restriction operator.

3.2.3 The function restriction. The iterative solution of nonlinear equations leads to linear problems which involve the current finite element approximation of the solution as parameter. Solving these problems with multilevel methods requires

the assembling of stiffness matrices on coarse levels where the current finite element approximation of the solution is needed. To this end, this finite element function has to be restricted to the coarse levels.

We will define a restriction operator $R_l^{l-1} : V_l^h \to V_{l-1}^h$ which maps a finite element function from the finite element space connected to level l in the multilevel hierarchy to a finite element function connected to level $l-1$. This operator is based on local L^2-projections and averaging.

The bases of V_l^h and V_{l-1}^h are again denoted by $\left\{\varphi_{l,j}^h\right\}$ and $\left\{\varphi_{l-1,i}^h\right\}$, respectively. Let $v_l^h \in V_l^h$ with

$$v_l^h = \sum_{i=1}^{\dim\left(V_l^h\right)} w_{l,i}\varphi_{l,i}^h$$

be given. The goal is to compute a function

$$R_l^{l-1}\left(v_l^h\right) = \sum_{i=1}^{\dim\left(V_{l-1}^h\right)} w_{l-1,i}\varphi_{l-1,i}^h.$$

We consider a mesh cell K on the geometric grid which is connected with V_{l-1}^h and assume that K possesses an affine reference transformation. Local values of the unknown coefficients $w_{l-1,i}$ are determined by the local L^2-projection

$$\sum_{i=1}^{\mathrm{card}\left(I_l\left(K,V_l^h\right)\right)} w_{l,i}|_K \left(\varphi_{l,i}^h, \varphi_{l-1,j}^h\right)_K = \sum_{i=1}^{\mathrm{card}\left(I_{l-1}\left(K,V_{l-1}^h\right)\right)} w_{l-1,i}|_K \left(\varphi_{l-1,i}^h, \varphi_{l-1,j}^h\right)_K$$

for all $j \in I_{l-1}\left(K, V_{l-1}^h\right)$, where $I_l\left(K, V_l^h\right)$ is defined in (4). The transformation to the reference cell $\hat{K}$ gives

$$\sum_{i=1}^{\mathrm{card}\left(I_l\left(K,V_l^h\right)\right)} w_{l,i}|_K \int_{\hat{K}} \hat{\varphi}_{l,i}^h \hat{\varphi}_{l-1,j}^h \left|\det J_K\left(\hat{\boldsymbol{x}}\right)\right| d\hat{\boldsymbol{x}} = \sum_{i=1}^{\mathrm{card}\left(I_{l-1}\left(K,V_{l-1}^h\right)\right)} w_{l-1,i}|_K \int_{\hat{K}} \hat{\varphi}_{l-1,i}^h \hat{\varphi}_{l-1,j}^h \left|\det J_K\left(\hat{\boldsymbol{x}}\right)\right| d\hat{\boldsymbol{x}}$$

for all $j \in I_{l-1}\left(K, V_{l-1}^h\right)$. Since $\left|\det J_K\left(\hat{\boldsymbol{x}}\right)\right|$ is constant, this relation simplifies to

$$\sum_{i=1}^{\mathrm{card}\left(I_l\left(K,V_l^h\right)\right)} w_{l,i}|_K \int_{\hat{K}} \hat{\varphi}_{l,i}^h \hat{\varphi}_{l-1,j}^h d\hat{\boldsymbol{x}} = \sum_{i=1}^{\mathrm{card}\left(I_{l-1}\left(K,V_{l-1}^h\right)\right)} w_{l-1,i}|_K \int_{\hat{K}} \hat{\varphi}_{l-1,i}^h \hat{\varphi}_{l-1,j}^h d\hat{\boldsymbol{x}}$$

for all $j \in I_{l-1}\left(K, V_{l-1}^h\right)$. This is a linear system of the form

$$G w_l|_K = M w_{l-1}|_K.$$

Thus, the local values of the unknown coefficients are given by

$$w_{l-1}|_K = M^{-1} G w_l|_K = R w_l|_K. \tag{6}$$

The matrix R is independent of K. That means, for all other mesh cells whose basis on the reference mesh cell has the same form as for K, one also needs the matrix R. This will be the case very often. E.g., if the grids are refined uniformly and the same finite element space is used on every level, the matrix R is needed for each mesh cell on each level! This matrix R will be computed once and then stored in the data base. Then, only a local matrix-vector product has to be computed in (6) which leads to a very fast algorithm. The final restriction is computed by an averaging

$$w_{l-1,i} = \frac{1}{\mathrm{card}\left(\mathcal{T}_{l-1,i}\right)} \sum_{K \in \mathcal{T}_{l-1,i}} w_{l-1,i}|_K.$$

For mesh cells with a non-affine reference transformation, we use for simplicity also (6) such that they are handled in the same way as mesh cells with an affine transformation. One can consider this approach also as a function restriction which is a local L^2-projection on the reference mesh cell and which is an approximation of a L^2-projection on the original mesh cell. A deterioration of the efficiency of multilevel solvers for the steady state Navier–Stokes equations on non-affine quadrilateral and hexahedral meshes using this function restriction was not observed in [10, 16].

Acknowledgements. The second author was partly supported by the Deutsche Forschungsgemeinschaft under Grant FOR 301.

References

1. Bastian, P., Birken, K., Johannsen, K., Lang, S., Neuß, N., Rentz-Reichert, H., Wieners, C.: UG – a flexible software toolbox for solving partial differential equations. Comput. Visual. Sci. 1, 27–40 (1997)
2. Behns, V.: Mortar–Techniken zur Behandlung von Grenzschichtproblemen. PhD thesis, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 2001
3. Boffi, D., Gastaldi, L.: On the quadrilateral Q_2–P_1 element for the Stokes problem. Int. J. Num. Meth. Fluids, 39(11), 1001–1011 (2002)
4. Brenner, S. C., Scott, L. R.: The Mathematical Theory of Finite Element Methods. Vol. 15. Texts in Applied Mathematics. New York: Springer–Verlag 1994
5. Ciarlet, P. G.: Basic error estimates for elliptic problems. In P.G. Ciarlet and J.L. Lions, (eds.) Handbook of Numerical Analysis II, pp. 19–351. Amsterdam, New York, Oxford, Tokyo: North–Holland 1991
6. Dunca, A., John, V., Layton, W. J.: Approximating local averages of fluid velocities: the equilibrium Navier–Stokes equations. Appl. Numer. Math., to appear, 2003
7. Girault, V., Raviart, P.-A.: Finite Element Methods for Navier–Stokes equations. Berlin-Heidelberg-New York: Springer-Verlag 1986
8. Iliescu, T., John, V., Layton, W. J., Matthies, G., Tobiska, L.: A numerical study of a class of LES models. Int. J. Comput. Fluid Dyn. 17(1), 75–85 (2003)
9. John, V.: Reference values for drag and lift of a two-dimensional time dependent flow around a cylinder. submitted to Int. J. Num. Meth. Fluids
10. John, V.: Higher order finite element methods and multigrid solvers in a benchmark problem for the 3D Navier–Stokes equations. Int. J. Num. Meth. Fluids 40, 775–798 (2002)
11. John, V.: Large Eddy Simulation of Turbulent Incompressible Flows. Analytical and Numerical Results for a Class of LES Models. Lecture Notes in Computational Science and Engineering 34. Berlin, Heidelberg, New York: Springer-Verlag 2003

12. John, V.: Slip with friction and penetration with resistance boundary conditions for the Navier–Stokes equations – numerical tests and aspects of the implementation. J. Comp. Appl. Math. 147, 287–300 (2002)
13. John, V.: The behaviour of the rational LES model in a two–dimensional mixing layer problem. Preprint 28, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 2002
14. John, V., Knobloch, P., Matthies, G., Tobiska, L.: Non-nested multilevel solvers for finite element discretizations of mixed problems. Computing, 68, 313–341 (2002)
15. John, V., Layton, W.J.: Approximating local averages of fluid velocities: The Stokes problem. Computing, 66, 269–287 (2001)
16. John, V., Matthies, G.: Higher order finite element discretizations in a benchmark problem for incompressible flows. Int. J. Num. Meth. Fluids, 37, 885–903 (2001)
17. Lavrova, O., Matthies, G., Mitkova, T., Polevikov, V., Tobiska, L.: Finite element methods for coupled problems in ferrohydrodynamics. In E. Bänsch (ed.) Challenges in Scientific Computing – CISC 2002, Lecture Notes in Computational Science and Engineering 35. Berlin, Heidelberg, New York: Springer-Verlag, 160–183, 2003
18. Matthies, G.: Finite element methods for free boundary value problems with capillary surfaces. Shaker Verlag, Aachen, 2002. PhD thesis, Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg
19. Matthies, G., Tobiska, L.: The inf-sup condition for the mapped Q_k/P_{k-1}^{disc} element in arbitrary space dimensions. Computing, 69(2), 119–139 (2002)
20. Schieweck, F.: A general transfer operator for arbitrary finite element spaces. Preprint 25, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 2000