THREE MATLAB IMPLEMENTATIONS OF THE LOWEST-ORDER RAVIART-THOMAS MFEM WITH A POSTERIORI ERROR CONTROL

C. BAHRIAWATI AND C. CARSTENSEN

ABSTRACT. The numerical approximation of the Laplace equation with inhomogeneous mixed boundary conditions in 2D with lowest-order Raviart-Thomas mixed finite elements is realized in three flexible and short MATLAB programs. The first, hybrid, implementation (LMmfem) assumes that the discrete function $p_h(x)$ equals $a + bx$ for x with unknowns $a \in \mathbb{R}^2$ and $b \in \mathbb{R}$ on each element and then enforces $p_h \in H(\mathrm{div}, \Omega)$ through Lagrange multipliers. The second, direct, approach (EBmfem) utilizes edge-basis functions $(\psi_E : E \in \mathcal{E})$ as an explicit basis of RT_0 with the edge-wise constant flux normal $p_h \cdot \nu_E$ as a degree of freedom. The third ansatz (CRmfem) utilizes the $P1$ nonconforming finite element method due to Crouzeix and Raviart and then postprocesses the discrete flux via a technique due to Marini. It is the aim of this paper to derive, document, illustrate, and validate the three MATLAB implementations EBmfem, LMmfem, and CRmfem for further use and modification in education and research. A posteriori error control with a reliable and efficient averaging technique is included to monitor the discretization error. Therein, emphasis is on the correct treatment of mixed boundary conditions. Numerical examples illustrate some applications of the provided software and the quality of the error estimation.

1. INTRODUCTION

This paper provides three short Matlab implementations of the lowest-order Raviart-Thomas mixed finite elements for the numerical solution of a Laplace equation with mixed Dirichlet and Neumann boundary conditions and their reliable error control through averaging techniques.

Section 2 presents details on the model boundary value problem, its weak, and its discrete mixed formulation. Three essentially equivalent implementations EBmfem, LMmfem, and CRmfem yield the three linear systems of equations (1.1)-(1.3) discussed below. The direct realization with an edge-oriented basis of $RT_0(\mathcal{T})$ from Section 4 in the Matlab program EBmfem leads to a linear system of the form

$$\begin{pmatrix} B & C \\ C^T & 0 \end{pmatrix} \begin{pmatrix} x_\psi \\ x_u \end{pmatrix} = \begin{pmatrix} b_D \\ b_f \end{pmatrix} \tag{1.1}$$

for given b_D and b_f which reflect inhomogeneous Dirichlet boundary conditions and volume forces, and for unknowns x_ψ, the normal components of the flux $p_h \cdot \nu_E$, which correspond to the basis of $M_{h,0}$, and the elementwise constant displacements with components x_u.

The continuity condition $p_h \in H(\mathrm{div}, \Omega)$ is directly satisfied by the edge-basis functions $(\psi_E : E \in \mathcal{E})$ of Section 4. In Section 5, it is enforced in the Matlab program LMmfem via Lagrange

Key words and phrases. Matlab, implementation, mixed finite element method, Raviart-Thomas finite element method, Crouzeix-Raviart finite element method, nonconforming finite element method.

multipliers and results in a linear system

$$\begin{pmatrix} B & C & D & F \\ C^T & 0 & 0 & 0 \\ D^T & 0 & 0 & 0 \\ F^T & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_\psi \\ x_u \\ x_{\lambda_M} \\ x_{\lambda_N} \end{pmatrix} = \begin{pmatrix} b_D \\ b_f \\ 0 \\ b_g \end{pmatrix} \tag{1.2}$$

with given b_D, b_f, b_g and unknowns $x_\psi, x_u, x_{\lambda_M}, x_{\lambda_N}$. For each element $T \in \mathcal{T}$ with center of gravity x_T, the three components a_1, a_2, b of $p_h(x) = a + b(x - x_T)$ are gathered in x_ψ while x_u denote the elementwise constant displacement approximations. The unknown x_{λ_N} are the Lagrange multipliers for the flux boundary conditions on Γ_N while the unknown x_{λ_M} are the Lagrange multipliers to the side restriction that the jump of the normal flux components $[p_h \cdot \nu_E]$ vanishes along interior edges. The components b_D, b_f, and b_g reflect inhomogeneous Dirichlet boundary conditions, volume forces, and applied surface forces.

The nonconforming Crouzeix-Raviart finite element method is implemented in the Matlab program CRmfem in Section 7 and leads to a linear system

$$Ax = b \tag{1.3}$$

for given b and unknown x associated with the volume force and the edge-oriented basis of the nonconforming finite element space $\mathcal{S}_D^{1,NC}(\mathcal{T})$. A result of [M, AC] then allows a modification to compute the discrete flux $p_h(x) = \nabla_T u - \frac{1}{2} f_h(x - x_T)$ of the Raviart-Thomas finite element discretization where f_h is the piecewise constant approximation of the right-hand side f. We give a more direct proof of that in Theorem 7.1 for a more general situation than in [M].

It is the aim of this paper to give a clear algorithmic description of the computation of the matrices A, B, C, D, and F in (1.1)-(1.3) and corresponding Matlab programs documented in Section 4, 5, and 7.

In Section 8, a posteriori error control is performed by an averaging technique. Therein, the error estimator is based on a smoother approximation, e.g. in $\mathcal{S}^1(\mathcal{T})^d$, the continuous $\mathcal{T}$-piecewise affine functions [C1, C2, CBa], to the discrete flux p_h obtained by an averaging operator $A : P_h \to \mathcal{S}^1(\mathcal{T})^d$ to p_h. That is, for each node $z \in \mathcal{N}$ and its patch ω_z, $(Ap_h)(z) := A_z p_h$ where $A_z := \pi_z \circ M_z$ is the composition of a continuous averaging $M_z : P_1(\mathcal{T}_z)^d \to \mathbb{R}^d$ and the orthogonal projection $\pi_z : \mathbb{R}^d \to \mathbb{R}^d$ onto the affine subspace $\mathcal{A}_z \subset \mathbb{R}^d$ that carries proper boundary conditions (cf. (8.9) below for details).

Then, Ap_h is defined by interpolation with first-order nodal basis functions $(\varphi_z : z \in \mathcal{N})$,

$$Ap_h = \sum_{z \in \mathcal{N}} A_z(p_h|_{\omega_z})\varphi_z.$$

The resulting averaging error estimator defined by $\eta_A := ||p_h - Ap_h||_{L^2(\Omega)}$ [C2, CBa] is reliable and efficient in the sense that

$$C_{\text{eff}}\eta_A - \text{h.o.t.} \le ||p - p_h||_{L^2(\Omega)} + \text{h.o.t.} \le C_{\text{rel}}\eta_A + \text{h.o.t.} \tag{1.4}$$

The remaining part of the paper is organized as follows. The model problem in its weak and discrete formulation is described in Section 2. The triangulation $\mathcal{T}$ and geometric data structures, which lie in the heart of the contribution, are presented in Section 3.

In Section 4, we define an edge-oriented basis $(\psi_E : E \in \mathcal{E})$ for $RT_0(\mathcal{T})$ where $\mathcal{E}$ is the set of all edges in the triangulation $\mathcal{T}$. Section 5 describes the Lagrange multiplier technique

to enforce continuity of the normal flux along interior edges $E \in \mathcal{E}_\Omega$. The Matlab realization of right-hand sides and boundary conditions is established in Section 6. Section 7 explains the flux and displacement approximation, p_h and u_h, via Crouzeix-Raviart finite element methods due to [M, AB, AW] for mixed boundary conditions.

The implementation of a posteriori error control, based on an averaging technique [C1, C2, CBa, CBK, V], is presented in Section 8. Numerical examples illustrate the documented software in Section 9 and the a posteriori error control via the averaging error estimate η_A. Post-processing routines of the display of the numerical solution are documented in the collected algorithm.

The collected algorithm gives the full listing of `EBmfem.m`, `LMmfem.m`, `CRmfem.m`, postprocessing (see `postproc.m`), and `Aposteriori.m`.

The Matlab programs run under Matlab 6 and require the main program files `EBmfem.m`, `LMmfem.m`, `CRmfem.m`, `StemaFEM.m`, `StemaNC.m`, `ph_OnRTElement.m`, and `main_uh.m` plus `postproc.m`, for each particular problem at hand, the user-specified files `coordinate.dat`, `element.dat`, `Dirichlet.dat`, and `Neumann.dat` as well as the user-specified Matlab functions `f.m`, `g.m`, and `u_D.m`. The graphical representation is performed with the function `ShowDisplacement.m` and `ShowFlux.m`; the a posteriori estimator is provided in the function `Aposteriori.m`. The complete listing can be downloaded from `http://www.math.tuwien.ac.at/~carsten/` under the item Software.

2. Model problem in its weak and discrete formulation

This section is devoted to details on the model example at hand in a strong and weak mixed formulation as well as in a first discrete formulation.

Let Ω be a bounded Lipschitz domain in the plane with outer unit normal ν on the polygonal boundary $\Gamma = \Gamma_D \cup \Gamma_N$ split into a relatively open Neumann boundary Γ_N and a closed Dirichlet boundary $\Gamma_D := \Gamma \setminus \Gamma_N$ of positive surface measure. Given $f \in L^2(\Omega)$, $g \in L^2(\Gamma_N)$, and $u_D \in H^1(\Omega) \cap C(\overline{\Omega})$, seek $u \in H^1(\Omega)$ such that

$$\Delta u + f = 0 \quad \text{in } \Omega, \qquad u = u_{\rm D} \quad \text{on } \Gamma_D, \qquad \nabla u \cdot \nu = g \quad \text{on } \Gamma_N. \tag{2.1}$$

Here and throughout, we use standard notation for Lebesgue and Sobolev spaces $L^2(\Omega)$ and $H^1(\Omega)$, respectively; $C(\overline{\Omega})$ denotes the set of continuous functions on $\overline{\Omega}$.

The second-order equation (2.1a) is split into two equations

$$\operatorname{div} p + f = 0 \quad \text{and} \quad p = \nabla u \quad \text{in } \Omega \tag{2.2}$$

for unknown $u \in H^1(\Omega)$ and $p \in L^2(\Omega)^2$ with $\operatorname{div} p \in L^2(\Omega)$. The standard functional analytical framework [BF] for (2.2), called *dual mixed formulation*, involves the function spaces

$$\begin{aligned} H(\operatorname{div},\Omega) &:= \{q \in L^2(\Omega)^2 : \operatorname{div} q \in L^2(\Omega)\}, \\ H_{0,N}(\operatorname{div},\Omega) &:= \{q \in H(\operatorname{div},\Omega) : q \cdot \nu = 0 \text{ on } \Gamma_N\}, \\ H_{g,N}(\operatorname{div},\Omega) &:= \{q \in H(\operatorname{div},\Omega) : q \cdot \nu = g \text{ on } \Gamma_N\}. \end{aligned}$$

Then, the weak formulation of (2.2) reads: Given $f \in L^2(\Omega)$, $g \in L^2(\Gamma_N)$, and $u_D \in H^1(\Omega) \cap C(\overline{\Omega})$, seek $p \in H_{g,N}(\mathrm{div}, \Omega)$ and $u \in L^2(\Omega)$ such that

$$\int_\Omega p \cdot q \, dx + \int_\Omega u \ \mathrm{div}\, q \, dx = \int_{\Gamma_D} u_D \, q \cdot \nu \, ds \qquad \text{for all } q \in H_{0,N}(\mathrm{div}, \Omega), \tag{2.3}$$

$$\int_\Omega v \ \mathrm{div}\, p \, dx = -\int_\Omega v f \, dx \qquad \text{for all } v \in L^2(\Omega). \tag{2.4}$$

The existence and uniqueness of the solution (p, u) of system (2.3)-(2.4) and its equivalence with (2.1) are well established (cf., e.g. [BF, § II, Thm. 1.2]).

For the discretisation of the flux p we consider the lowest-order Raviart-Thomas space

$$\begin{aligned} RT_0(\mathcal{T}) := \{q \in L^2(T) : \forall T \in \mathcal{T} \ \exists a \in \mathbb{R}^2 \, \exists b \in \mathbb{R} \, \forall x \in T, q(x) = a + bx \\ \text{and } \forall E \in \mathcal{E}_\Omega, [q]_E \cdot \nu_E = 0\}, \end{aligned}$$

where $\mathcal{T}$ is a regular triangulation (cf. Section 3), $\mathcal{E}_\Omega$ is the set of all interior edges, and $[q]_E := q|_{T_+} - q_{T_-}$ along E denotes the jump of q across the edge $E = T_+ \cap T_-$ shared by the two neighbouring elements T_+ and T_- in $\mathcal{T}$.

The continuity of the normal components on the boundaries reflects the conformity $RT_0(\mathcal{T}) \subset H(\mathrm{div}, \Omega)$ (as defined below). For the second approach with EBmfem, this continuity is built in the shape function ψ_E along $E \in \mathcal{E}_\Omega$. In the hybrid formulation, the continuity of $E \in \mathcal{E}_\Omega$ is enforced via the Lagrange multiplier technique.

With the $\mathcal{E}_N$-piecewise constant approximation g_h of g, $g_h|_E = \int_E g \, ds/|E|$ for each $E \in \mathcal{E}_N$ of length $|E|$, the discrete spaces read

$$\begin{aligned} M_{h,g} &:= \{q_h \in RT_0(\mathcal{T}) : q_h \cdot \nu = g_h \text{ on } \Gamma_N\}, \\ M_h &:= M_{h,0} = RT_0(\mathcal{T}) \cap H_{0,N}(\mathrm{div}, \Omega), \\ L_h = P_0(\mathcal{T}) &:= \{v_h \in L_2(\Omega) : T \in \mathcal{T}, \ v_h|_T \in P_0(T)\}. \end{aligned}$$

The discrete problem reads: seek $(u_h, p_h) \in L_h \times M_{h,g}$ with

$$\int_\Omega p_h \cdot q_h \, dx + \int_\Omega u_h \, \mathrm{div}\, q_h \, dx = \int_{\Gamma_D} u_D \, q_h \cdot \nu \, ds \qquad \text{for all } q_h \in M_{h,0}, \tag{2.5}$$

$$\int_\Omega v_h \, \mathrm{div}\, p_h \, dx = -\int_\Omega v_h \, f \, dx \qquad \text{for all } v_h \in L_h. \tag{2.6}$$

The system (2.5)-(2.6) admits a unique solution (u_h, p_h) (cf., e.g. [AB], [BF, § IV.1, Prop. 1.1]).

In Section 4 we will define an edge-oriented basis $(\psi_j : j = 1, \ldots, N)$ of $RT_0(\mathcal{T})$ with $M_{h,0} = \mathrm{span}\{\psi_1, \ldots, \psi_M\} \subseteq RT_0(\mathcal{T})$. With respect to this basis (possibly in a different order of the indices), the components $x_\psi = (x_1, \ldots, x_N)$ of $p_h = \sum_{k=1}^N x_k \psi_k \in M_{h,g}$ and $x_u = (x_{N+1}, \ldots, x_{N+L})$ of $u_h|_{T_\ell} = x_{N+\ell}$ for $\ell = 1, \ldots, L$ and for an enumeration $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_L\}$ of the $L = \mathrm{card}(\mathcal{T})$ elements. Then, (2.5)-(2.6) are recast into the linear system of equations for the

unknown $(x_1, \ldots, x_M)$ and $(x_{N+1}, \ldots, x_{N+L})$

(2.7)

$$\sum_{k=1}^{M} x_k \int_\Omega \psi_j \cdot \psi_k \, dx + \sum_{\ell=1}^{L} x_{N+\ell} \int_{T_\ell} \operatorname{div} \psi_j \, dx = \int_{\Gamma_D} u_D \, \psi_j \cdot \nu \, ds - \sum_{m=M+1}^{N} g_h|_{E_m} \int_\Omega \psi_j \cdot \psi_m \, dx,$$

(2.8)

$$\sum_{k=1}^{M} x_k \int_{T_\ell} \operatorname{div} \psi_k \, dx = - \int_{T_\ell} f \, dx - \sum_{m=M+1}^{N} g_h|_{E_m} \int_{T_\ell} \operatorname{div} \psi_k \, dx$$

for $j = 1, \ldots, M$, $\ell = 1, \ldots, L$ and the known $(x_{M+1}, \ldots, x_N) := (g_h|_{E_m}, m = M+1, \ldots, N)$. For the case of the presentation, it is assumed in Section 2 that the Neumann edges have the numbers $M+1, \ldots, N$ while this will be defined on the Matlab realization below. The enumeration $\{E_1, \ldots, E_N\} = \mathcal{E}_\Omega \cup \mathcal{E}_N$ of the interior edges $\mathcal{E}_\Omega = \{E_1, \ldots, E_M\}$ and the edges $\mathcal{E}_N = \{E_{M+1}, \ldots, E_N\}$ on the Neumann boundary is explained in the subsequent section.

3. Triangulation and geometric data structures

To describe further the edge-basis $(\psi_j : j = 1, \ldots, N)$, this section provides notation on the triangulation $\mathcal{T}$ and the edges $\mathcal{E}$ and their data representation.

3.1. **Geometric Description.** Suppose the domain Ω with the polygonal boundary $\Gamma = \Gamma_D \cup \Gamma_N$ is covered by a regular triangulation $\mathcal{T}$, in the sense of Ciarlet [Ci, BS], into triangles. That is, $\mathcal{T}$ is a set of closed triangles $T = \operatorname{conv}\{a, b, c\}$ of positive area with vertices a, b, c, called nodes, such that, the union $\cup\mathcal{T} = \overline{\Omega}$ of the triangulation covers $\overline{\Omega}$ exactly and any non-empty intersection of two distinct triangles of $\mathcal{T}$ equals one common edge $E = \operatorname{conv}\{a, b\}$ or a node $\{a\}$ shared by the two triangles. The set of all edges and nodes is abbreviated by $\mathcal{E}$ and $\mathcal{N}$, respectively. The set

$$\mathcal{E} := \mathcal{E}_\Omega \cup \mathcal{E}_D \cup \mathcal{E}_N$$

of all edges in $\mathcal{T}$ is partitioned into edges on the Dirichlet boundary $\mathcal{E}_D := \{E \in \mathcal{E} : E \subset \Gamma_D\}$, on the Neumann boundary $\mathcal{E}_N := \{E \in \mathcal{E} : E \subset \overline{\Gamma}_N\}$, and the set of all interior element edges $\mathcal{E}_\Omega$. The skeleton of all points which belong to some element's boundary is the union of all edges and read $\cup\mathcal{E} = \cup_{E \in \mathcal{E}} E = \{x \in \overline{\Omega} : \exists T \in \mathcal{T}, x \in \partial T\}$.

All the Matlab programs employ the following data representation of $\mathcal{T}, \mathcal{N}$ and $\mathcal{E}$. The $n := \operatorname{card}(\mathcal{N})$ nodes $\mathcal{N} = \{z_1, \ldots, z_n\}$ with its coordinates $z_j = (x_j, y_j) \in \mathbb{R}^2$ are stored in the user-defined file `coordinate.dat`, where the row number j contains the two coordinates x_j, y_j. The element $T_j = \operatorname{conv}(z_k, z_\ell, z_m)$ is the convex hull of its three vertices z_k, z_ℓ, z_m in $\mathcal{N}$ described by the global numbers k, ℓ, m stored in the row number j of the file `element.dat`. It is a convention in all data structures [ACF, ACFK, CK] that the enumeration k, ℓ, m of the three nodes is counterclockwise. The information of the edge $E = \operatorname{conv}(z_k, z_\ell)$ of $\mathcal{E}_D$ and $\mathcal{E}_N$ is stored in the data files `Dirichlet.dat` and `Neumann.dat`, respectively, represent $E = \operatorname{conv}\{z_k, z_\ell\}$ in row j by the two (global) numbers k, ℓ. It is a convention that the tangential unit vector τ_E along E points from z_k to z_ℓ and that the outer normal ν_E points to the right. Figure 1 and 2 display a triangulation $\mathcal{T}$ and its data representation in `coordinate.dat`, `element.dat`, and `Dirichlet.dat`.

The initialization of `coordinate.dat`, `element.dat`, `Dirichlet.dat`, and Neumann is performed by the simple Matlab commands `load coordinate.dat; load element.dat; load Dirichlet.dat; load Neumann.dat`.

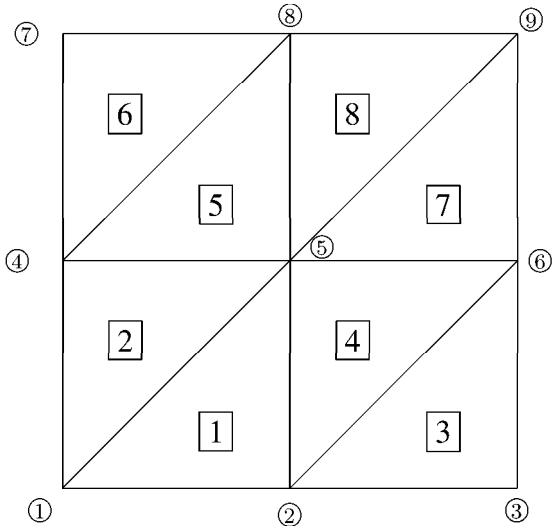

FIGURE 1. Triangulation of the unit square in eight congruent triangles with enumeration of nodes (numbers in circles) and an enumeration of triangles (numbers in boxes).

`coordinates.dat`

0	0
0.5	0
1	0
0	0.5
0.5	0.5
1	0.5
0	1
0.5	1
1	1

`element.dat`

1	2	5
1	5	4
2	3	6
2	6	5
4	5	8
4	8	7
5	6	9
5	9	8

`Dirichlet.dat`

1	2
2	3
3	6
6	9
9	8
8	7
7	4
4	1

FIGURE 2. Data files `coordinate.dat`, `element.dat`, and `Dirichlet.dat` for the triangulation displayed in Figure 1.

With the geometric information from Figure 1 and 2 and with $f := 1$, $g := 0$, and `u_D=0`, the mixed finite element approximation shows the approximate displacement u_h and flux p_h in Figure 3.

Definition 3.1 (Normals and jumps on edges)**.** For each $E \in \mathcal{E}$, let ν_E be a unit normal which coincides with the exterior unit normal $\nu = \nu_E$ along Γ if $E \in \mathcal{E}_D \cup \mathcal{E}_N$. Given any $E \in \mathcal{E}_\Omega$ and any $\mathcal{T}$-piecewise continuous function $\rho \in L^2(\Omega; \mathbb{R}^2)$, let $J_E := [\rho \cdot \nu_E]$ denote the jump of ρ across E in the direction ν_E on $E \in \mathcal{E}_\Omega$ defined by

$$[\rho \cdot \nu_E] = (\rho|_{T_+} - \rho|_{T_-}) \cdot \nu_E \quad \text{if } E = T_+ \cap T_- \tag{3.1}$$

for $T_+, T_- \in \mathcal{T}$ such that ν_E points from T_- into T_+. Set $J_E := 0$ for $E \in \mathcal{E} \setminus \mathcal{E}_\Omega$. [This convention is found useful in the treatment of boundary edges but *not* standard in the literature.]

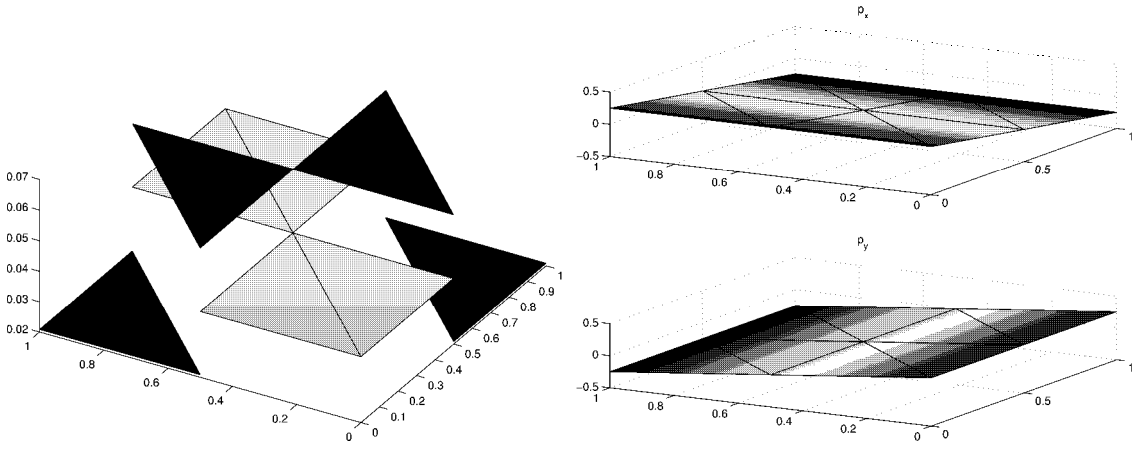

FIGURE 3. Discrete solution of the problem (2.1) for the prescribed data of Figure 1 with $f := 1$, $g := 0$, and `u_D=0`. The displacement u_h is shown left, the flux components p_{h_x} (top) and p_{h_y} (bottom) are shown right.

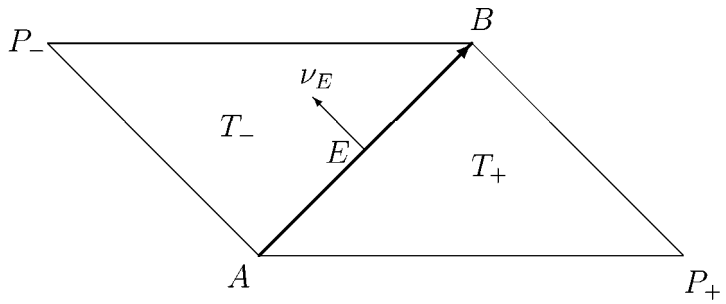

FIGURE 4. Two neighbouring triangles T_+ and T_- that share the edge $E = \partial T_+ \cap \mathcal{T}_-$ with initial node A and end node B and unit normal ν_E. The orientation of ν_E is such that it equals the outer normal of T_+ (and hence points into T_-).

3.2. **Edge enumeration.** The degrees of freedom in the flux variable of mixed formulation are edge-oriented. The underlying edge enumeration in EBmfem, LMmfem, and CRmfem connect all edges with geometric information of the triangulation. Three main groups of data structures built in the three matrices called `nodes2edge`, `nodes2element`, and `edges2element` are computed in the function `edge.m`

```
function [nodes2element,nodes2edge,noedges,edge2element,...
          interioredge]=edge(element,coordinate)
```

The corresponding data structures and computation of the three matrices `nodes2edge`, `nodes2element`, and `edges2element` are given in Subsection 3.2.1-3.2.3.

3.2.1. *Matrix nodes2element.* The quadratic sparse matrix `nodes2element` of dimension card($\mathcal{N}$) describes the number of an element as a function of its two vertices

$$\texttt{nodes2element}(k,\ell) = \begin{cases} j & \text{if } (k,\ell) \text{ are numbers of nodes of element number } j; \\ 0 & \text{otherwise.} \end{cases}$$

Notice carefully that the two neighbouring elements T_+ and T_- as depicted in Figure 4 that share one common edge conv(A, B) with endpoints A = coordinate(k) and B = coordinate(ℓ) have the number j_+:=nodes2element(k,ℓ) and j_-:=nodes2element(ℓ,k); i.e. $T_\pm$ has number $j_\pm$. The computation of the matrix nodes2element in Matlab reads

```
% Matrix nodes2element
nodes2element=sparse(size(coordinate,1),size(coordinate,1));
for j=1:size(element,1)
    nodes2element(element(j,:),element(j,[2 3 1]))= ...
    nodes2element(element(j,:),element(j,[2 3 1]))+j*eye(3,3);
end
```

3.2.2. *Matrix nodes2edge.* The symmetric sparse matrix nodes2edge of dimension card($\mathcal{N}$) describes number of edges given by

$$\texttt{nodes2edge}(k,\ell) = \begin{cases} j & \text{if edge } E_j = \text{conv}\{z_k, z_\ell\} \text{ number } j \text{ belongs to nodes with number } k, \ell; \\ 0 & \text{otherwise.} \end{cases}$$

The computation of the matrix nodes2edge in Matlab reads

```
% Matrix nodes2edge
B=nodes2element+nodes2element';
[I,J]=find(triu(B));
nodes2edge=sparse(I,J,1:size(I,1),size(coordinate,1), ...
           size(coordinate,1));
nodes2edge=nodes2edge+nodes2edge';
```

Therein, I denotes non-zero indices of the upper triangular part of the matrix B. The number of edges is abbreviated by noedges=size(I,1).

3.2.3. *Matrix edge2element.* The (noedges$\times 4$) matrix edges2element represents the initial node k and the end node ℓ of the edge of row number j and the number n, m of elements T_+, T_- that share the edge. The line number j of the matrix edges2element contains the four components

$$k \quad \ell \quad m \quad n$$

Therein, the neighbouring elements T_+, T_- with number m, n are specified with respect to the convention of Figure 4. The entry edge2element(j,3) defines T_+ and hence the orientation of the normal ν_E of the edge $E = T_+ \cap T_-$ number j throughout all the Matlab programs. In case of an exterior edge $E \in E_D \cup \mathcal{E}_N$ the fourth entry is zero [thereby, ν_E is exterior to Ω],

$$\texttt{edge2element}(j,[3,4]) = \begin{cases} [m,n] & \text{if a common edge } j \text{ belongs to elements } m, n; \\ [m,0] & \text{if an edge } j \text{ belongs to an element } m. \end{cases}$$

The computation of the matrix edge2element in Matlab reads

```
edge2element=zeros(noedges,4);
for m=1:size(element,1)
 for k=1:3
 initial_node=element(m,k);
 end_node=element(m,rem(k,3)+1);
 p=nodes2edge(element(m,k),element(m,rem(k,3)+1));
 if edge2element(p,1)==0
   edge2element(p,:)=[initial_node end_node ...
                      nodes2element(initial_node,end_node) ...
```

```
                    nodes2element(end_node,initial_node)];
  end
 end
end
```

Using this structure one can immediately compute a list `find(edge2element(:,4))` of the numbers of the interior edges and the list `find(edge2element(:,4)==0)` of the exterior edge. Figure 5 displays the matrix `edge2element` computed from the data of Figure 2.

1	2	1	0
2	3	3	0
4	1	2	0
5	1	1	2
2	5	1	4
5	4	2	5
6	2	3	4
3	6	3	0
6	5	4	7
7	4	6	0
8	4	5	6
5	8	5	8
8	7	6	0
9	5	7	8
6	9	7	0
9	8	8	0

FIGURE 5. Matrix `edge2element` generated by the function `edge.m` from the data of Figure 2 for the triangulation depicted in Figure 1.

4. EDGE-BASIS FUNCTIONS AND STIFFNESS MATRICES IN EBMFEM

This section is devoted to the edge-basis functions for the lowest order Raviart-Thomas finite elements employed in the Matlab program EBmfem that realizes (1.1). Figure 6 displays the notation adapted for one typical triangle throughout this section.

4.1. Construction of edge-basis function ψ_E.

This subsection is devoted to the (local) definition of the edge-basis function for a triangle depicted in Figure 6.

Definition 4.1 (Local definition of ψ_E)**.** Let E_1, E_2, E_3 be the edges of a triangle T opposite to its vertices P_1, P_2, P_3, respectively, and let ν_{E_j} denote the unit normal vector of E_j chosen with a global fixed orientation while ν_j denotes the outer unit normal of T along E_j. Define

$$\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T|}(x - P_j) \qquad \text{for } j = 1, 2, 3 \text{ and } x \in T, \tag{4.1}$$

where $\sigma_j = \nu_j \cdot \nu_{E_j}$ is $+1$ if ν_{E_j} points outward and otherwise -1; $|E_j|$ is the length of E_j, and $|T|$ is the area of T,

$$2|T| = \det\,(P_2 - P_1, P_3 - P_1) = \det \begin{pmatrix} P_1 & P_2 & P_3 \\ 1 & 1 & 1 \end{pmatrix} \tag{4.2}$$

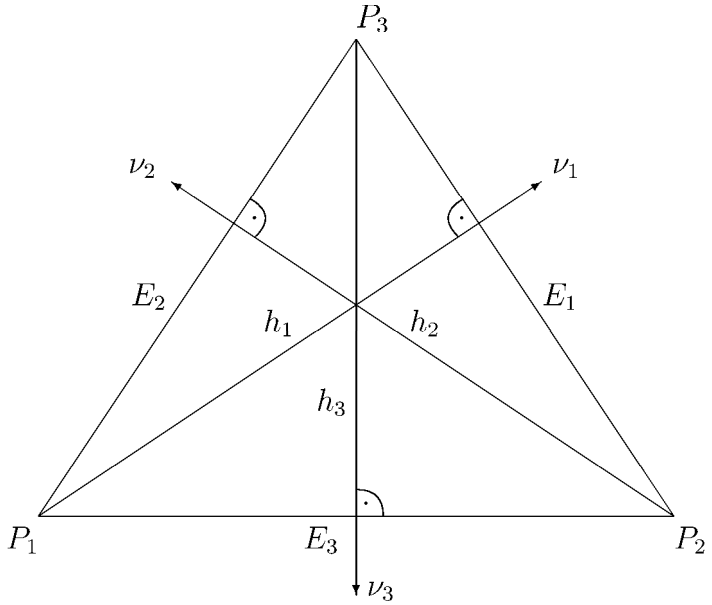

FIGURE 6. Triangle T = conv$\{P_1, P_2, P_3\}$ with vertices P_1, P_2, P_3 (ordered counterclockwise) and opposite edges E_1 = conv$\{P_2, P_3\}$, E_2 = conv$\{P_1, P_3\}$, E_3 = conv$\{P_1, P_2\}$ of lengths $|E_1|$, $|E_2|$, $|E_3|$, respectively. The area $|T|$ satisfies Equation (4.2) with a plus sign in front of the determinant; with the heights h_1, h_2, h_3 depicted, there holds $2|T| = |E_j|h_j$ for $j = 1, 2, 3$.

(with the 3×3-matrix that consists of the 2×3 matrix of the three vectors $P_1, P_2, P_3 \in \mathbb{R}^2$ plus three ones in the last row).

Definition 4.2 (Notation for elements that share an edge E)**.** Let $T_\pm = \text{conv}(E \cup \{P_\pm\})$ for the vertex $P_\pm$ opposite to E of $T_\pm$ such that the edge $E = \text{conv}\{A, B\}$ orients from A to B. Then ν_E points outward from T_+ to T_- (cf. Figure 4) with a positive sign. If E is an exterior edge $E \in \mathcal{E}_D \cup \mathcal{E}_N$, then $\nu = \nu_E$ is the exterior normal and $E \subset \partial T_+$ defines T_+ (and T_- is undefined).

Note that the normal direction changes if we reverse the orientation of the edge $E = \text{conv}\{A, B\}$.

Definition 4.3 (Global definition of ψ_E)**.** Given an edge $E \in \mathcal{E}$ there are either two elements T_+ and T_- in $\mathcal{T}$ with the joint edge $E = \partial T_+ \cap \partial T_-$ or there is exactly one element T_+ in $\mathcal{T}$ with $E \subset \partial T_+$. Then if $T_\pm = \text{conv}(E \cup \{P_\pm\})$ for the vertex $P_\pm$ opposite to E of $T_\pm$ set

$$\psi_E(x) := \begin{cases} \pm\frac{|E|}{2|T_\pm|}(x - P_\pm) & \text{for } x \in T_\pm, \\ 0 & \text{elsewhere.} \end{cases} \tag{4.3}$$

Lemma 4.4. *There hold*

(a) $\psi_E \cdot \nu_E = \begin{cases} 0 & \text{along } (\cup\mathcal{E}) \setminus E, \\ 1 & \text{along } E; \end{cases}$

(b) $\psi_E \in H(\text{div}, \Omega)$*;*

(c) $(\psi_E : E \in \mathcal{E})$ *is a basis of* $RT_0(\mathcal{T})$*;*

(d) $\text{div}\, \psi_E = \begin{cases} \pm\frac{|E|}{|T_\pm|} & \text{on } T_\pm, \\ 0 & \text{elsewhere.} \end{cases}$

Proof. (a) Consider $T_\pm = \mathrm{conv}\{P_\pm, A, B\}$ with $E = T_+ \cap T_-$ shown in Figure 4 and denote $\omega_E := \mathrm{int}(T_+ \cup T_-)$. Let $F \in \mathcal{E} \setminus \{E\}$ be an edge different from E. Clearly, $\psi_E \cdot \nu_F$ vanishes for $F \not\subset \partial\omega_E$. For $x \in F \subset \partial\omega_E$ the vector $x - P_\pm$ is tangential to $\partial\omega_E$ and hence $\psi_E(x) \cdot \nu_F = 0$ as well. For $x \in E = F$, $(x - P_\pm) \cdot \nu_E$ is the height of the triangle $T_\pm$ (cf. Figure 6). Hence it is constant and equals $2|T_\pm|/|E|$. The factor in Definition 4.3 then yields $\psi_E(x) \cdot \nu_E = 1$.

(b) Obviously, $\psi_E \in L^2(\Omega)$ and $\psi|_{T_+}$ equals

$$\psi_E(x) = \pm\frac{|E|}{2T_+}P_\pm \pm \frac{|E|}{2T_+}x \qquad \text{for all } x \in T_\pm.$$

For any $F \in \mathcal{E}_\Omega$ there follows $[\psi_E]_F \cdot \nu_F = 0$ from (a). Hence $\psi_E \in RT_0(\mathcal{T}) \subseteq H(\mathrm{div}, \Omega)$.

(c) The functions $(\psi_E : E \in \mathcal{E})$ are (uniquely) determined by $\psi_E \in RT_0(\mathcal{T})$ and $\psi_E \cdot \nu_F = 1$ for $E = F \in \mathcal{E}$ and $\psi_E \cdot \nu_F = 0$ for $E \neq F$ in $\mathcal{E}$. Given any $q_h \in RT_0(\mathcal{T})$ notice that $q_h \cdot \nu_E$ is constant on $E \in \mathcal{E}$ and define

$$p_h := q_h - \sum_{E\in\mathcal{E}}(q_h \cdot \nu_E)\psi_E \in RT_0(\mathcal{T}).$$

Then, (a) implies $p_h \cdot \nu_E = 0$ for all $E \in \mathcal{E}$. On $T \in \mathcal{T}$ with edges E_1, E_2, E_3 as in Figure 6, there holds

$$p_h(P_j) \cdot \nu_{E_k} = 0 \qquad \text{for } k = \{1,2,3\} \setminus \{j\}$$

at the vertex $x = P_j$ opposite to E_j. Since $p_h|_T$ is affine, this proves $p_h \equiv 0$ on $T \in \mathcal{T}$. Consequently, $RT_0(\mathcal{T}) \subseteq \mathrm{span}\{\psi_E : E \in \mathcal{E}\}$. It remains to verify that $(\psi_E : E \in \mathcal{E})$ is linear independent: Given real coefficients $(x_E : E \in \mathcal{E})$ with

$$p_h = \sum_{E\in\mathcal{E}} x_E\,\psi_E \equiv 0 \quad \text{in } \Omega,$$

we deduce with (a) that $0 = p_h|_F \cdot \nu_F = x_F$.

(d) This is immediate from (4.3). □

4.2. **Local stiffness matrices.** In this subsection, we recall notation from Figure 6 for a triangle T with edges E_1, E_2, and E_3 with (local) number $j = 1, 2, 3$ and abbreviate $\psi_j := \psi_{E_j}$.

Definition 4.5 (Local Stiffness Matrices)**.** Let the local stiffness matrices $B_T, C_T \in \mathbb{R}^{3\times3}$ be defined by

$$(B_T)_{jk} := \int_T \psi_j \cdot \psi_k \, dx \quad \text{for } j,k = 1,2,3, \tag{4.4}$$

$$C_T := \mathrm{diag}\Big(\int_T \mathrm{div}\ \psi_1\, dx, \int_T \mathrm{div}\ \psi_2\, dx, \int_T \mathrm{div}\ \psi_3\, dx\Big). \tag{4.5}$$

Lemma 4.6. *Given (4.4)-(4.5) and the matrices*

$$M := \begin{pmatrix} 2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \\ 1 & 0 & 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 2 \end{pmatrix} \in \mathbb{R}^{6\times6} \text{ and } N := \begin{pmatrix} 0 & P_1 - P_2 & P_1 - P_3 \\ P_2 - P_1 & 0 & P_2 - P_3 \\ P_3 - P_1 & P_3 - P_2 & 0 \end{pmatrix} \in \mathbb{R}^{6\times3},$$

there holds

$$B_T = \frac{1}{48|T|} C_T^T\, N^T\, M\, N\, C_T. \tag{4.6}$$

Proof. Let $\lambda_1, \lambda_2, \lambda_3$ denote the barycentric coordinates in the triangle T of Figure 3. Then an affine function (4.1) reads $\psi_E(x) = \sigma_{E_j}|E_j|/(2|T|)(\lambda_1(x)(P_1-P_j)+\lambda_2(x)(P_2-P_j)+\lambda_3(x)(P_3-P_j))$. Hence one calculates

$$\begin{aligned} B_{jk} &= \int_T \psi_j \cdot \psi_k \, dx = \sigma_{E_j}\sigma_{E_k} \frac{|E_j||E_k|}{4\,|T|^2} \sum_{\substack{\ell=1\\ m=1}}^{3} \int_T \lambda_\ell (P_\ell - P_j) \cdot \lambda_m (P_m - P_k)\, dx \\ &= \frac{\sigma_{E_j}|E_j|\sigma_{E_k}|E_k|}{4|T|^2} \sum_{\substack{\ell=1\\ m=1}}^{3} (P_\ell - P_j)\cdot(P_m - P_k) \int_T \lambda_\ell \lambda_m \, dx. \end{aligned}$$

Since $\int_T \lambda_\ell \lambda_m \, dx = \frac{|T|}{12}(1+\delta_{\ell m})$, this yields

$$\begin{aligned} (B_T)_{jk} &= \frac{\sigma_{E_j}|E_j|\sigma_{E_k}|E_k|}{4|T|^2} \sum_{\substack{\ell=1\\ m=1}}^{3} (P_\ell - P_j)\cdot(P_m - P_k)\Big(\frac{|T|}{12}(1+\delta_{\ell m})\Big) \\ &= \frac{\sigma_{E_j}|E_j|}{48|T|}\Bigg(\Big(\sum_{\ell=1}^{3}(P_\ell - P_j)\Big)\cdot\Big(\sum_{m=1}^{3}(P_m - P_k)\Big) \\ &\quad + \sum_{\ell=1}^{3}(P_\ell - P_j)\cdot(P_\ell - P_k)\Bigg)\sigma_{E_k}|E_k|. \end{aligned}$$

Then, direct calculations for each $(B_T)_{jk}$, $j,k = 1,2,3$ and $P_{j,k} = (P_{jk,x}, P_{jk,y})^T$ verify (4.6). □

The Matlab realization for the computation of the local stiffness matrix B_T in EBmfem via Lemma 4.6 reads

```
function B=stimaB(coord);
N=coord(:)*ones(1,3)-repmat(coord,3,1);
C=diag([norm(N([5,6],2)),norm(N([1,2],3)),norm(N([1,2],2))]);
M=spdiags([ones(6,1),ones(6,1),2*ones(6,1),ones(6,1),ones(6,1)],...
          [-4,-2,0,2,4],6,6);
B=C*N'*M*N*C/(24*det([1,1,1;coord]));
```

Therein, the matrices B and C equal B_T and C_T from (4.5) up to the global signs $\sigma_{E_1}, \sigma_{E_2}$, and σ_{E_3} cooperated with the assembling described in the subsequent subsection.

4.3. **Assembling the global stiffness matrices.** The global stiffness matrix A consists of matrices $B \in \mathbb{R}^{N\times N}$ and $C \in \mathbb{R}^{N\times L}$ computed from the local stiffness matrix B_T and C_T; recall $N = \text{card}(\mathcal{E})$ and $L = \text{card}(\mathcal{T})$. Given any element $T \in \mathcal{T}$ of number j, the command `I=nodes2edge(element(j,[2 3 1]),element(j,[3 1 2]))` gives the vector (m_1, m_2, m_3) of global edge numbers. The sign $\sigma_{E_{m_k}}$ of the global edge number m_k with respect to the current element T with outer unit normal ν_T is $\sigma_{E_{m_k}} = \nu_{E_{m_k}} \cdot \nu_T$ for $k = 1,2,3$. The sign

$\sigma_{E_{m_k}} = \pm 1$ is negative if and only if j=`edge2element(`m_k`,4)`. In a formal way, the assembling

$$B = \sum_{T \in \mathcal{T}} B(T) \quad \text{and} \quad C = \sum_{T \in \mathcal{T}} C(T)$$

requires the concept of the $N \times N$ matrix $B(T)$ defined by the entries

$$B(T)\begin{pmatrix} m_1, m_2, m_3 \\ m_1, m_2, m_3 \end{pmatrix} = \text{diag}\Big(\sigma_{E_{m_1}}, \sigma_{E_{m_2}}, \sigma_{E_{m_3}}\Big)\ \text{stimaB(coord)}\ \text{diag}\Big(\sigma_{E_{m_1}}, \sigma_{E_{m_2}}, \sigma_{E_{m_3}}\Big)$$

in the components of m_1, m_2, and m_3; similar formulae hold for $C(T)$. The Matlab routines for the assembling of the global stiffness matrix B and matrix C read

```
B=sparse(noedges,noedges);
C=sparse(noedges,size(element,1));
for j=1:size(element,1)
  coord=coordinate(element(j,:),:)';
  I=diag(nodes2edge(element(j,[2 3 1]),element(j,[3 1 2])));
  signum=ones(1,3);
  signum(find(j==edge2element(I,4)))=-1;
  n=coord(:,[3 1 2])-coord(:,[2 3 1]);
  B(I,I)=B(I,I)+diag(signum)*stimaB(coord)*diag(signum);
  C(I,j)=diag(signum)*[norm(n(:,1)) norm(n(:,2)) norm(n(:,3))]';
end
```

and the global matrix A from (1.1) is generated by

```
A=sparse(noedges+size(element,1),noedges+size(element,1));
A=[B,C;C',sparse(size(C,2),size(C,2))];
```

5. Lagrange Multiplier Technique and Stiffness Matrices in LMmfem

This section is devoted to the hybrid mixed finite element realization LMmfem that realizes (1.2). Lagrange multipliers are introduced on the interfaces $\cup\mathcal{E}_\Omega$ and on the Neumann boundary Γ_N to relax the continuity required on the normal component. Set

$$\Lambda_h := P_0(\mathcal{E}_\Omega)^2 := \{\lambda \in L^\infty(\cup\mathcal{E}) : \forall E \in \mathcal{E}, \lambda|_E := \lambda_E \in \mathbb{R}^2 \text{ and } \lambda_{\partial\Omega} \equiv 0\}, \tag{5.1}$$

$$N_h := P_0(\mathcal{E}_N)^2. \tag{5.2}$$

Then the discrete problem reads [BF]: Find $(u_h, p_h, \lambda_h, \ell_h) \in L_h \times M_h \times \Lambda_h \times N_h$ such that there holds for all $(v_h, q_h, \mu_h, m_h) \in L_h \times M_h \times \Lambda_h \times N_h$

(5.3)

$$\int_\Omega p_h \cdot q_h\, dx + \int_\Omega u \cdot \operatorname{div} q_h\, dx - \sum_{E \in \cup\mathcal{E}} \int_E [q_h \cdot \nu_E]\lambda_h\, ds - \int_{\Gamma_N} (q_h \cdot \nu) \cdot \ell_h\, ds = \int_{\Gamma_D} u_D\, q_h \cdot \nu\, ds,$$

$$\int_\Omega v_h \operatorname{div} p_h\, dx = -\int_\Omega v_h f\, dx, \tag{5.4}$$

$$-\sum_{E \in \cup\mathcal{E}} \int_E [p_h \cdot \nu_E]\, \mu_h\, ds = 0, \tag{5.5}$$

$$-\int_{\Gamma_N} [p_h \cdot \nu] \cdot m_h\, ds = -\int_{\Gamma_N} g \cdot m_h\, ds. \tag{5.6}$$

Throughout this section, given any triangle T with the center of gravity (x_T, y_T) we consider the shape functions

$$\psi_1 = (1,0), \quad \psi_2 = (0,1), \quad \psi_3 = (x - x_T, y - y_T) \qquad \text{for } x \in T. \tag{5.7}$$

The evaluation of the integrals in (5.3)-(5.6) for (5.7) yields the linear system (1.2).

Definition 5.1 (Local stiffness matrices)**.** For each element T, the local stiffness matrices $B_T \in \mathbb{R}^{3\times 3}$ and $C_T \in \mathbb{R}^{3\times 1}$ are given by

$$(B_T)_{jk} := \int_T \psi_j \cdot \psi_k dx \qquad \text{for } j,k = 1,2,3, \tag{5.8}$$

$$(C_T)_j := \int_T \operatorname{div} \psi_j \, dx \qquad \text{for } j = 1,2,3. \tag{5.9}$$

Lemma 5.2. *Given an element T with vertices $z_j = (x_j, y_j) \in \mathbb{R}^{3\times 3}$, $j = 1,2,3$, and area $|T|$, define $s := |z_2 - z_1|^2 + |z_3 - z_2|^2 + |z_3 - z_1|^2$. Then there holds*

$$B = |T| \operatorname{diag}\big(1,1,s/36\big) \quad \textit{and} \quad C_T = \big(0,0,2|T|\big).$$

Proof. The direct calculation of $(B_T)_{jk}$ in (5.8) shows that $(B_T)_{jk}$ are zero except the diagonal entries

$$(B_T)_{11} = (B_T)_{22} = |T|, \qquad (B_T)_{33} = \int_T \Big((x - x_T)^2 + (y - y_T)^2\Big)\, dy = 1/36\,|\,T|\,s.$$

The calculation of $(C_T)_j$ for (5.9) results in

$$(C_T)_j = \int_T \operatorname{div}\, \psi_j \, dx = 0 \quad \text{for } j = 1,2, \quad \text{and } (C_T)_3 = 2|T|. \qquad \square$$

The Matlab realization for the local stiffness matrices B_T and C_T in LMmfem via Lemma 5.2 reads

```
B=sparse(3*size(element,1),3*size(element,1));
C=sparse(3*size(element,1),size(element,1));
for j=1:size(element,1)
  s=sum(sum((coordinate(element (j,[2 3 1]),:)- ...
            coordinate(element (j,[1 2 3]),:)).^2));
  B(3*(j-1)+[1 2 3],3*(j-1)+[1 2 3])=det([1 1 1; ...
            coordinate(element(j,:),:)'])/2*diag([1 1 s/36]);
  C(3*(j-1)+[1 2 3],j)=[0;0;det([1 1 1;coordinate(element(j,:),:)'])];
end
```

The global stiffness matrix B of dimension $3L$ and the global matrix C of dimension $3L \times L$ are block diagonal matrices,

$$\begin{pmatrix} B_1 & 0 & \dots & 0 \\ 0 & B_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & B_{\mathrm{card}(\mathcal{T})} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} C_1 & 0 & \dots & 0 \\ 0 & C_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & C_{\mathrm{card}(\mathcal{T})} \end{pmatrix}. \tag{5.10}$$

Definition 5.3. For each $E_k \in \mathcal{E}_\Omega$, the local vector $D_k \in \mathbb{R}^{6\times 1}$ for the Lagrange multiplier technique is defined by

$$D_{kj} = -\int_{E_k} \psi_j \cdot \nu_{E_k}\, ds \quad \text{for } j = 1,\dots,6, \tag{5.11}$$

where ψ_1, ψ_2, ψ_3 and ψ_4, ψ_5, ψ_6 denote the three shape functions in $RT_0(T_+)$ and $RT_0(T_-)$, respectively.

Recall the notation and orientation of Figure 4 and define

$$EV(k) := (EV(k)_x, EV(k)_y) := B - A \in \mathbb{R}^2 \quad \text{for the edge } E_k = \text{conv}\{A, B\} \tag{5.12}$$

with nodes A and B and the tangential unit vector $\tau_k := (B-A)/|B-A|$ and the unit normal $\nu_k = (EV(k)_y, -EV(k)_x)/|E_k| \in \mathbb{R}^2$ (assume that A, B are ordered counterclockwise).

Lemma 5.4. *For each $E_k \in \mathcal{E}_\Omega$, $k = 1, 2 \dots, M$, let $h := (y_T - y_0, x_0 - x_T) \cdot EV(k)$ with a node (x_0, y_0) of the edge E_k. Then there holds*

$$D_k = \big(-EV(k)_y, EV(k)_x, -h, EV(k)_y, -EV(k)_x, h\big)^T. \tag{5.13}$$

Proof. Direct calculations of D_{kj} in (5.11) result in

$$\begin{aligned}
-D_{k1} = D_{k4} &= \int_{E_k} \psi_1 \cdot \nu_k\, ds = \int_{E_k} \nu_x(k)\, ds = -\nu_x(k)|E_k| = EV(k)_y, \\
-D_{k2} = D_{k5} &= \int_{E_k} \psi_2 \cdot \nu_k\, ds = \int_{E_k} \nu_y(k)\, ds = -\nu_y(k)|E_k| = EV(k)_x, \\
-D_{k3} = D_{k6} &= \int_{E_k} \psi_3 \cdot \nu_k\, ds = \int_{E_k} (y_T - y, x - x_T) \cdot EV(k)/|E_k|\, ds =: h. \qquad \square
\end{aligned}$$

Given an interior edge $E = T_+ \cap T_-$ shared by the elements $T_\pm \in \mathcal{T}$ of global numbers $j_\pm$ and with (5.13), the global matrix $D \in \mathbb{R}^{L\times M}$ reads

$$D\begin{pmatrix} & & & k & & \\ j_+ & j_+ + 1 & j_+ + 2 & j_- & j_- + 1 & j_- + 2 \end{pmatrix} = D_k. \tag{5.14}$$

With the sublist I of `edge2element` of all interior edges, the Matlab realization of the global matrix D reads

```
I=edge2element(find(edge2element(:,4)))
D=sparse(3*size(element,1),size(I));
MidPoint=reshape(sum(reshape(coordinate(element',:),3,...
              2*size(element,1))),size(element,1),2)/3;
EV=coordinate(I(:,2),:)-coordinate(I(:,1),:);
for k=1:size(I,1)
  h=(coordinate(I(k,[1 1]),:)-...
     MidPoint(I(k,3:4),:))*[EV(k,2);-EV(k,1)];
  D([3*(I(k,3)-1)+[1 2 3],3*(I(k,4)-1)+...
    [1 2 3]],k)=[-EV(k,2);EV(k,1);-h(1);EV(k,2);-EV(k,1);h(2)];
end
```

The global matrix A from (1.2) is assembled by its blocks B, C, D.

6. MATLAB REALIZATION OF RIGHT-HAND SIDES AND BOUNDARY CONDITIONS

This section is devoted to the computation of the right-hand sides in (1.1)-(1.3). The ingredi-ences of which include numerical integration over elements and edges of the given functions `f.m`, `g.m`, and `u_D.m`.

6.1. **Computing b_f.** The given volume force is provided by the user-specified function `f.m`. The integrals for each element T_ℓ with centre of gravity z_{T_ℓ}, namely

$$-\int_{T_\ell} f(x)\,dx \quad \text{for } \ell = 1, \ldots, L, \tag{6.1}$$

form the vector b_f on the right-hand sides in (1.1)-(1.3). In the simplest choice, the numerical realization involves a one-point numerical quadrature rule

$$b_{f_\ell} := -|T_\ell|\, f(z_{T_\ell}) \tag{6.2}$$

which in Matlab reads

```
-det([1 1 1;coordinate(element(l,:),:)']) * ...
          f(sum(coordinate(element(l,:),:))/3)/6
```

The global vector b (of length $N+L$ or $4*L+M$ or N) on the right-hand side has a different structure in each linear system of equations (1.1)-(1.3). For (1.1) the entry (6.2) equals `b(noedges+`ℓ`)` and for (1.2) the entry (6.2) equals `b(4 * size(element) +` ℓ`)` for $\ell = 1, \ldots, L$. [The right-hand side of (1.3) is stored differently according to an enumeration of edges.]

6.2. **Computing b_D.**

6.2.1. *Dirichlet condition for EBmfem.* Since the normal components of the test functions are zero or equal one along the edge $E \in \mathcal{E}_D$ of number j with mid-point (x_M, y_M), a simple one-point integration reads

$$b_D := u_D(x_M, y_M)\,|E| \approx \int_E u_D\, ds. \tag{6.3}$$

Given the values of `u_D` in a user-specified function `u_D.m`, the Matlab realization of (6.3) reads

```
for k=1:size(Dirichlet,1)
  b(nodes2edge(Dirichlet(k,1),Dirichlet(k,2)))=...
    norm(coordinate(Dirichlet(k,1),:)-coordinate(Dirichlet(k,2),:))*...
    u_D(sum(coordinate(Dirichlet(k,:),:))/2);
end
```

6.2.2. *Dirichlet condition for LMmfem.* Since the normal components of the test functions are zero or equal one along the edge $E \in \mathcal{E}_D$ with mid-point (x_M, y_M), a simple one-point integration reads

$$\begin{aligned} b_D &:= \sum_{k=1}^{\mathrm{card}(\mathcal{E}_D)} u_D(x_M, y_M)\Big(EV(k)_y, -EV(k)_x, (y_T - y_0, x_0 - x_T)\cdot EV(k)\Big) \\ &\approx \sum_{k=1}^{\mathrm{card}(\mathcal{E}_D)} \int_{E_k} u_D \psi_j \cdot \nu_k\, ds \end{aligned} \tag{6.4}$$

with components $EV(k)_x$ and $EV(k)_y$ of $EV(k)$ in (5.12). The Matlab realization of (6.4) reads

```
EV=coordinate(Dirichlet(:,2),:)-coordinate(Dirichlet(:,1),:);
for k=1:size(Dirichlet,1)
  h=(coordinate(Dirichlet(k,1),:)-...
     MidPoint(nodes2element(Dirichlet(k,1),Dirichlet(k,2)),:))*...
     [EV(k,2);-EV(k,1)];
  b(3*nodes2element(Dirichlet(k,1),Dirichlet(k,2))-[2 1 0])=...
  b(3*nodes2element(Dirichlet(k,1),Dirichlet(k,2))-[2 1 0]) + ...
  u_D(sum(coordinate(Dirichlet(k,:),:))/2)*[EV(k,2);-EV(k,1);h];
end
```

6.3. **Incorporating Neumann conditions.**

6.3.1. *Neumann conditions for EBmfem.* Let $B_1 := B\binom{1,\dots,M}{1,\dots,M}$, $B_2 := B\binom{M+1,\dots,N}{1,\dots,M}$, and $B_3 := B\binom{M+1,\dots,N}{M+1,\dots,N}$ be a partition of B so that the system of linear equations resulting from the construction described in (2.7)-(2.8) can be written as

$$\begin{pmatrix} B_1 & B_2 & C \\ B_2^T & B_3 & 0 \\ C^T & 0 & 0 \end{pmatrix} \begin{pmatrix} x_\psi(1,\dots,M) \\ x_\psi(M+1,\dots,N) \\ x_u \end{pmatrix} = \begin{pmatrix} b_D(1,\dots,N) \\ b_f \end{pmatrix}. \tag{6.5}$$

Therein, $x_\psi(1,\dots,M)$ is the vector of the unknowns at the free edges $\mathcal{E}_\Omega$ to be determined and $x_\psi(M+1,\dots,N)$ are the given values at the edges which are on the Neumann boundary. Hence, the first and second blocks of equations can be rewritten as

$$\begin{pmatrix} B_1 & C \\ C^T & 0 \end{pmatrix} \begin{pmatrix} x_\psi(1,\dots,M) \\ x_u \end{pmatrix} = \begin{pmatrix} b_D(1,\dots,M) \\ b_f \end{pmatrix} - \begin{pmatrix} B_2 x_\psi(M+1,\dots,N) \\ 0 \end{pmatrix},$$

where the values $x_\psi(\ell) = g_h|_{E_\ell}$ for $\ell = M+1,\dots,N$. In fact, this is the formulation of (2.7)-(2.8) with $g = 0$ at non-Neumann nodes. The Matlab realization reads

```
if ~isempty(Neumann)
tmp=zeros(noedges+size(element,1),1);
tmp(diag(nodes2edge(Neumann(:,1),Neumann(:,2))))=...
    ones(size(diag(nodes2edge(Neumann(:,1),Neumann(:,2))),1),1);
FreeEdge=find(~tmp);
x=zeros(noedges+size(element,1),1);
CN=coordinate(Neumann(:,2),:)-coordinate(Neumann(:,1),:);
for j=1:size(Neumann,1)
 x(nodes2edge(Neumann(j,1),Neumann(j,2)))=...
 g(sum(coordinate(Neumann(j,:),:))/2,CN(j,:)*[0,-1;1,0]/norm(CN(j,:)));
end
b=b-A*x;
```

and the solution $x \in \mathbb{R}^N$ is computed via

```
x(FreeEdge)=A(FreeEdge,FreeEdge)\b(FreeEdge)
```

6.3.2. *Neumann conditions for LMmfem.* For each $E_k \in \mathcal{E}_N$, define

$$F_j = \int_{E_k} \psi_j \cdot \nu_k \, ds \quad \text{for } j = 1, 2, 3. \tag{6.6}$$

Using notation in (5.12), let $EV(k)_x$ and $EV(k)_y$ denote the components of $EV(k)$ with respect to x- and y- coordinates along the Neumann boundary. Direct calculations result in

$$\begin{aligned} F_1 &= \int_{E_k} \psi_1 \cdot \nu_k \, ds = \int_{E_k} \nu_x(k) \, ds = \nu_x(k)|E_k| = EV(k)_y, \\ F_2 &= \int_{E_k} \psi_2 \cdot \nu_k \, ds = \int_{E_k} \nu_y(k)|E_k| \, ds = \nu_y(k)|E_k| = -EV(k)_x, \\ F_3 &= \int_{E_k} \psi_3 \cdot \nu_k \, ds = \frac{1}{|E_k|} \int_{E_k} (y_T - y, x - x_T) \cdot EV(k) \, ds \\ &= (y_T - y, x - x_T) \cdot EV(k) =: h, \quad \text{where } (x,y) \in E_k \in \mathcal{E}_N. \end{aligned}$$

Hence $F_T = \Big(EV(k)_y, -EV(k)_x, h\Big)^T$ and its Matlab realization reads

```
if ~isempty(Neumann)
  F=sparse(3*size(element,1),size(Neumann,1));
  CN=coordinate(Neumann(:,2),:)-coordinate(Neumann(:,1),:);
  for k=1:size(Neumann,1)
    h=(coordinate(Neumann(k,1),:)-...
       MidPoint(nodes2element(Neumann(k,1),Neumann(k,2)),:))...
       *[CN(k,2);-CN(k,1)];
       F([3*(nodes2element(Neumann(k,1),Neumann(k,2))-1)+...
       [1 2 3]],k)=[CN(k,2);-CN(k,1);h];
  end
  F=[F;sparse(size(A,1)-size(F,1),size(F,2))];
  A=[A,F;F',sparse(size(F,2),size(F,2))];
  % Right-hand side
  b=[b;sparse(size(Neumann,1),1)];
  for j=1:size(Neumann,1)
    b(4*size(element,1)+size(I,1)+j)= ...
    b(4*size(element,1)+size(I,1)+j)+norm(CN(j,:))*...
    g(sum(coordinate(Neumann(j,:),:))/2,CN(j,:)*[0,-1;1,0]/norm(CN(j,:)));
  end
end
```

The user-defined Matlab function `g.m` specifies the values of g in its first argument; its second argument is the outward normal vector. For the example in Subsection 9.1, the function `g.m` reads

```
function val=g(x,n);
[a,r]=cart2pol(x(:,1),x(:,2));
ind=find(a<0);
a(ind)=a(ind)+2*pi*ones(size(ind));
val=(2/3*r.^(-1/3).*[-sin(a/3),cos(a/3)])*n';
```

7. Nonconforming Crouzeix-Raviart Finite Element Method

This section is devoted to the flux approximation of the Raviart-Thomas MFEM obtained from nonconforming Crouzeix-Raviart finite elements. The idea is to employ the identity $p_h(x) = \nabla_T u_{NC} - f_h(x - x_T)/2$ for the discrete flux p_h from lowest-order Raviart-Thomas MFEM and the discrete flux ∇u_{NC} from the P1 nonconforming FE approximation and the piecewise constant

function $f_h|_T := \int_T f(x)\,dx/|T|$ for $T \in \mathcal{T}$. Therein, x_T denote the centre of gravity of the triangle T and $x \in T$. Set

$$
\begin{aligned}
P_1(\mathcal{T}) &= \{f \in L^2(\Omega) : \forall T \in \mathcal{T}, f|_T \in P_1(T)\},\\
\mathcal{S}^{1,NC}(\mathcal{T}) &= \{v \in P_1(\mathcal{T}) :\ v \text{ are continuous in all midpoints } z_E \text{ of edges } E \in \mathcal{E}_\Omega\},\\
\mathcal{S}^{1,NC}_D(\mathcal{T}) &= \{v \in \mathcal{S}^{1,NC}(\mathcal{T}) :\ v(z_E) = 0 \text{ for all } E \in \mathcal{E}_D\}.
\end{aligned} \tag{7.1}
$$

The discrete problem reads: Find $u_{NC} \in \mathcal{S}^{1,NC}(\mathcal{T})$ with $u_{NC}(z) = u_D(z)$ for all midpoints z of edges in $\mathcal{E}_D$ such that

$$
\int_\Omega \nabla_T u_{NC} \cdot \nabla_T v_h = \int_\Omega f_h v_h\,dx + \int_{\Gamma_N} g_h v_h\,ds \quad \text{for all } v_h \in \mathcal{S}^{1,NC}_D(\mathcal{T}). \tag{7.2}
$$

The subsequent theorem holds in any dimensions n, i.e. $\Omega \subset \mathbb{R}^n$, while the rest of paper focuses on $n = 2$.

Theorem 7.1. *Let $p_h \in RT_0(\mathcal{T})$ solve the mixed system with the right-hand side $f \in L^2(\Omega)$ and $p_h \cdot \nu = g_h$ on Γ_N where $g_h := \fint_E g(x)\,ds/|E|$ for all $E \in \mathcal{E}_N$. Let $u_{NC} \in \mathcal{S}^{1,NC}_D(\mathcal{T})$ solve (7.2) with $f_h = -\operatorname{div} p_h$ and g_h. Set $b_h(x) = f_T(x - x_T)/n$ for $x \in T \in \mathcal{T}$. Then there holds*

$$
p_h = \nabla_T u_{NC} + b_h. \tag{7.3}
$$

Proof. For some constants $a_T \in \mathbb{R}^n$ and $b_T \in \mathbb{R}$, $T \in \mathcal{T}$, there holds

$$
p_h|_T = a_T + b_T(x - x_T) \quad \text{for all } x \in T
$$

and for the centre of inertia x_T of T. The MFEM flux approximation $p_h \in H(\operatorname{div}, \Omega)$ satisfies $\int_T \operatorname{div} p_h\,dx = -\int_T f\,dx$ and so [with $\operatorname{div} p_h = b_T n$],

$$
f_T := \fint_T f(x)\,dx = -b_T\,n \quad \text{whence} \quad b_T = -f_T/n.
$$

Let ψ_E be a basis function of an interior edge E. Since $p_h \cdot \nu_E$ is constant and the jump $[\psi_E]$ of ψ_E on each edge in $\mathcal{E}_\Omega$ has integral zero, there holds

$$
\int_{\cup\mathcal{E}} p_h \cdot \nu [\psi_E]\,ds = 0.
$$

For an exterior edge $E \in \mathcal{E}_N$, $\psi_E\, p_h \cdot \nu_E = g_h|_E := \fint_E g\,ds$ and so

$$
\int_E g_h\,ds = \int_{\cup\mathcal{E}} p_h \cdot \nu [\psi_E]\,ds.
$$

There we follow the convention $[\psi_E] \equiv \psi_E$ on $E \cap \Gamma_N$. An elementwise integration by parts shows

$$
\begin{aligned}
\int_{\Gamma_N} g_h \psi_E\,ds &= \int_{\cup\mathcal{E}} p_h \cdot \nu [\psi_E]\,ds\\
&= \int_\Omega \psi_E \operatorname{div} p_h\,dx + \int_\Omega p_h \cdot \nabla_T \psi_E\,dx.
\end{aligned}
$$

Since $\operatorname{div} p_h = -f_h$ in Ω it follows that

$$
\int_\Omega p_h \cdot \nabla_T \psi_E\,dx = \int_\Omega f_h \psi_E\,dx + \int_{\Gamma_N} g_h \psi_E\,ds.
$$

Notice that $\nabla_{\mathcal{T}}\psi_E$ is constant on each T and so

$$\int_\Omega p_h \cdot \nabla_{\mathcal{T}}\psi_E \, dx = \sum_{T\in\mathcal{T}} |T| \, a_T \cdot \nabla_{\mathcal{T}}\psi_E|_T.$$

Let a_h denote the $\mathcal{T}$-piecewise constant values of $(a_T : T \in \mathcal{T})$, i.e. $a_h|_T := a_T$ for all $T \in \mathcal{T}$. Since $u_{NC} \in \mathcal{S}_D^{1,NC}(\mathcal{T})$ solves (7.2) there holds

$$\int_\Omega (a_n - \nabla_{\mathcal{T}} u_{NC}) \cdot \nabla_{\mathcal{T}} v_h \, dx = 0 \quad \text{for all } v_h \in \mathcal{S}_D^{1,NC}(\mathcal{T}).$$

Define $a_h - \nabla_{\mathcal{T}} u_{NC} =: c_h \in P_0(\mathcal{T})^n$. Then

$$\begin{aligned} 0 = \int_\Omega c_h \cdot \nabla_{\mathcal{T}}\psi_E \, dx &= \int_{\cup\mathcal{E}} [c_h \cdot \nu_{\mathcal{E}} \, \psi_E] \, ds \\ &= \int_E \nu_E \cdot [c_h] \, ds + \int_{(\cup\mathcal{E})\setminus E} [c_h \cdot \nu_{\mathcal{E}} \psi_E] \, ds. \end{aligned}$$

Since the second integral vanishes $[c_h] \cdot \nu = 0$ on E. This holds for all $E \in \mathcal{E}_\Omega$. Thus $c_h \in H(\mathrm{div}, \Omega)$ and $c_h \cdot \nu = 0$ on Γ_N. Hence c_h is a proper test function in the first equation for p_h, namely

$$\int_\Omega p_h \cdot c_h \, dx + \int_\Omega \mathrm{div}\, c_h \, p_h dx = 0.$$

Observe that $\mathrm{div}\, c_h = 0$ as $c_h \in H(\mathrm{div}, \Omega) \cap P_0(T)^n$. Therefore,

$$0 = \int_\Omega p_h \cdot c_h \, ds = \int_\Omega a_h \cdot c_h \, dx.$$

Notice that $v_h = u_{NC}$ is possible above and hence

$$\int_\Omega c_h \cdot \nabla_{\mathcal{T}} u_{NC} \, dx = 0.$$

Since $c_h = a_h - \nabla_{\mathcal{T}} u_{NC}$ this shows $0 = \int_\Omega c_h \cdot c_h \, dx = 0$, i.e. $a_h \equiv \nabla_{\mathcal{T}} u_{NC}$. □

The following lemma is devoted to the local stiffness matrix of the $P1$ nonconforming finite element method in the spirit of [ACF].

Lemma 7.2. *For any $T \in \mathcal{T}$ denote by M^{NC} and M the local stiffness matrix of $P1$ nonconforming finite element and $P1$ conforming finite element, respectively. Then there holds*

$$M^{NC} = 4\, M. \tag{7.4}$$

Proof. For a triangle $T \in \mathcal{T}$ set $P_j := (x_j, y_j)$ for $1 \le j \le 3$ and let $z := (\overline{x}_j, \overline{y}_j) = (x_{j+1} + x_{j+2}, y_{j+1} + y_{j+2})/2$ be the mid points on edges. Here the indices are modulo 3. Then

$$\begin{pmatrix} \overline{y}_j - \overline{y}_{j+1} \\ \overline{x}_{j+1} - \overline{x}_j \end{pmatrix} = \frac{1}{2} \begin{pmatrix} y_{j+1} + y_{j+2} - y_{j+2} - y_{j+3} \\ x_{j+2} + x_{j+1} - x_{j+3} - x_{j+2} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} y_j - y_{j+1} \\ x_{j+1} - x_j \end{pmatrix}. \tag{7.5}$$

Since $T = \mathrm{conv}(P_j, z_j)$ for $1 \le j \le 3$, $|T| := |T|_{\mathrm{conv}(P_j, z_j)} = 4|T|_{\mathrm{conv}(z_1, z_2, z_3)}$. With this, (7.5) and by [ACF] it follows that $M^{NC} = 4M$. □

The Matlab realization of the stiffness matrix M^{NC} in (7.4) via a result from [ACF] reads

```
 function M_NC=StemaNC(vertices)
G=[ones(1,3);vertices']\[zeros(1,2);eye(2)];
M_NC=4*det([ones(1,3);vertices'])*G*G';
```

The complete Matlab program for the solution u_{NC} is provided in function `CR_FEM.m` displayed in the collected algorithm.

The remaining part of this section concerns the computation of (u_h, p_h) from the solution u_{NC}. Given $u_{NC} \in \mathcal{S}^{1,NC}(\mathcal{T})$ from `CRmfem.m`, Theorem 7.1 allows the computation pf $p_h(x) := \nabla u_{NC} - f_h(x - x_T)/2$ for $x \in T \in \mathcal{T}$. Let $(\varphi_E : E \in \mathcal{E})$ be the edge-oriented first-order nonconforming basis functions of $\mathcal{S}^{1,NC}(\mathcal{T})$ and let $(\overline{x}_j, \overline{y}_j)$ be the midpoint of edge E_j for a local enumeration E_1, E_2, E_3 of the edges of T. Based on

$$\begin{pmatrix} \nabla\varphi_{E_1} \\ \nabla\varphi_{E_2} \\ \nabla\varphi_{E_3} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ \overline{x}_1 & \overline{x}_2 & \overline{x}_3 \\ \overline{y}_1 & \overline{y}_2 & \overline{y}_3 \end{pmatrix}^{-1} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{7.6}$$

the Matlab realization for the computation of p_h from (7.3) reads

```
function ph=ph_OnRTElement(element,coordinate,nodes2edge,noedges,...
                           edge2element,uNC)
MidPoint=reshape(sum(reshape(coordinate(element',:),3,...
                             2*size(element,1))),size(element,1),2)/3;
ph=zeros(3*size(element,1),2);
for j=1:size(element,1)
  I=diag(nodes2edge(element(j,[2 3 1]),element(j,[3 1 2])));
  gradUNC=([-1,1,1;1,-1,1;1,1,-1]*uNC(I))'*...
          ([1,1,1;coordinate(element(j,:),:)']\[0,0;1,0;0,1]);
  ph(3*(j-1)+[1,2,3],:)=ones(3,1)*gradUNC-(det([1 1 1;...
  coordinate(element(j,:),:)'])*f(sum(coordinate(element(j,:),:))/2))*...
  (coordinate(element(j,:),:)-ones(3,1)*MidPoint(j,:))/2;
end
```

Given $p_h \in RT_0(\mathcal{T})$, some remarks on the computation of the piecewise constant displacements $u_h \in P_0(\mathcal{T})$ conclude this section. With given $p_h \in RT_0(\mathcal{T})$ from Theorem 7.1 and the unknown $u_h \in P_0(\mathcal{T})$ satisfying (2.3), i.e. with ψ_E from (4.3), there holds

$$\int_\Omega p_h \cdot \psi_E \, dx + \int_\Omega \operatorname{div} \psi_E u_h \, dx = \int_{\Gamma_D} u_D \cdot \psi_E \, ds \qquad \text{for all } E \in \mathcal{E}_\Omega \cup \mathcal{E}_N. \tag{7.7}$$

For $E \in \mathcal{E}_D$ one obtains immediately

$$u_h = -\frac{2}{E} \int_{T_+} p_h \cdot \psi_E \, dx \quad \text{for } E \in \mathcal{E}_D \text{ and } E \subset T_+ \in \mathcal{T}, \tag{7.8}$$

while, for $E = \partial\overline{\Gamma}_+ \cap \partial\overline{\Gamma}_- \in \mathcal{E}_\Omega$ with $T_+, T_- \in \mathcal{T}$, there holds

$$u_h|_{T_+} \int_{T_+} \operatorname{div} \psi_E \, dx + u_h|_{T_-} \int_{T_-} \operatorname{div} \psi_E \, dx = \int_\Omega p_h \cdot \psi_E \, dx. \tag{7.9}$$

Given one the two values $u_h|_{T_+}$ and $u_h|_{T_-}$, the other value follows from (7.9). The Matlab function `Main_uhFromNC.m` displayed in the collected algorithm computes the values $u_h|_T$ from (7.8)-(7.9) in a loop over all elements T. The computation of u_h starts from the Dirichlet boundary from (7.8) and proceeds the values u_h on the neighbouring elements via (7.9).

8. A Posteriori Error Control

This section is devoted to a posteriori error control based on an averaging technique for the Poisson problem. Recall equation (2.1) with a given right-hand side $f \in L^2(\Omega)$ and a known approximation $p_h \in L^2(\Omega)^d$ to the unknown exact flux $p \in H(\mathrm{div};\Omega)$ in the bounded Lipschitz domain $\Omega \subset \mathbb{R}^d$.

Suppose $p_h \in L^2(\Omega)^2$ satisfies a Galerkin property with respect to a test function finite element space that includes continuous piecewise linear $\mathcal{S}^1_D(\mathcal{T})$ (with homogeneous Dirichlet boundary conditions) based on a regular triangulation $\mathcal{T}$ of Ω, i.e.

$$\int_\Omega p_h \cdot \nabla v_h dx = \int_\Omega f v_h \, dx \quad \text{for all } v_h \in S^1_D(\mathcal{T}). \tag{8.1}$$

In averaging techniques, the error $||p - p_h||_{L_2(\Omega)}$ is estimated by the approximation error of a smoother approximation $q_h \in \mathcal{S}^1(\mathcal{T})^d$ to p_h. In fact, the minimal value

$$\eta_M := \min_{p_h \in \mathcal{S}^1_D(\mathcal{T})} ||p_h - q_h||_{L_2(\Omega)} \tag{8.2}$$

is certainly efficient up to higher order terms of the exact solution p; a triangle inequality gives

$$\eta_M \le ||p - p_h||_{L_2(\Omega)} + \min_{q_h \in \mathcal{S}^1(\mathcal{T})^d} ||p - q_h||_{L_2(\Omega)}. \tag{8.3}$$

It is striking that η_M is also reliable [CBa, CBK] in the sense of

$$||p - p_h||_{L_2(\Omega)} \le C_{\mathrm{rel}} \eta_M + \text{h.o.t.} \tag{8.4}$$

This section describes a Matlab realization of one averaging operator A with $Ap_h = q_h$ in (8.2) to define an upper bound of η_M,

$$\eta_M \le \eta_A := ||p_h - Ap_h||_{L^2(\Omega)},$$

and emphasizes the proper treatment of boundary conditions. Consider the following discrete spaces

$$P_k(\mathcal{T}) := \{v_h \in L^\infty(\Omega) : \forall T \in \mathcal{T}, v_h|_T \in P_k(T)\} \quad \text{for } k = 0, 1, \tag{8.5}$$

$$\mathcal{S}^1(\mathcal{T}) := P_1(\mathcal{T}) \cup C(\Omega) = \mathrm{span}\{\varphi_z : z \in \mathcal{N}\}, \tag{8.6}$$

$$P_h := P(\mathcal{T}) := \{p_h \in L^\infty(\Omega)^d : \forall T \in \mathcal{T}, p_h|_T \in P(T)\} \subseteq P_1(\mathcal{T})^d, \tag{8.7}$$

$$\mathcal{Q}_h := \{q_h \in \mathcal{S}^1(\mathcal{T})^d : \forall z \in \mathcal{N} \cup \Gamma, q_h(z) \in \mathcal{A}_z\} \tag{8.8}$$

with the affine subspace

$$\mathcal{A}_z := \{a \in \mathbb{R}^d : \forall E \in \mathcal{E}_z \cap \mathcal{E}_N, g(z) = a \cdot \nu_E \text{ and } \forall E \in \mathcal{E}_z \cap \mathcal{E}_D, \nabla_E u_D(z) = (a)_E\} \tag{8.9}$$

of $\mathbb{R}^d$ and $\nabla_E u_D$ denotes the tangential derivative along E. Here, the Dirichlet and Neumann boundary conditions on the gradient $p = \nabla u$ are asserted at each boundary node $z \in \mathcal{N}$ by $p(z) \in \mathcal{A}_z$. Define Ap_h by an operator $A : P_h \to \mathcal{Q}_h$ to average p_h on its patch ω_z [C2] with

$$Ap_h := \sum_{z \in \mathcal{N}} A_z(p_h|_{\omega_z})\varphi_z \quad \text{and} \quad A_z := \pi_z \circ M_z : P_1(\mathcal{T}_z)^d \to \mathbb{R}^d. \tag{8.10}$$

The operator $M_z : P_1(\mathcal{T})^d \to \mathbb{R}^d$ defines an averaging process and is chosen as the integral mean of p_h

$$p_z := M_z(p_h) := \fint_{\omega_z} p_h \, dx = \int_{\omega_z} p_h \, dx / |\omega_z|$$

for any node z with patch ω_z of area $|\omega_z|$ (for $d = 2$). Let $\pi_z : \mathbb{R}^d \to \mathbb{R}^d$ denotes the orthogonal projection onto the affine subspace $\mathcal{A}_z \subset \mathbb{R}^d$ from (8.9) written in the form

$$\mathcal{A}_z = \pi_z(0) + V_z \tag{8.11}$$

is a linear subspace V_z of $\mathbb{R}^d$. The (non-linear) orthogonal projection π_z is Lipschitz continuous with $\mathrm{Lip}(\pi_z) \leq 1$ and, for each $a \in \mathbb{R}^d$, there holds $a - \pi_z(a) \perp V_z$.

For a Dirichlet boundary condition $u = u_D$ on E in term of $a = p(z) = \nabla u(z)$ at z, the term $\nabla_E u_D(z) = (a)_E$ in (8.9) is equivalent to $\partial u_D(z)/\partial \tau_E = a \cdot \tau_E$ for all (tangential unit vectors) $\tau_E \in \mathbb{R}^d$ with $\tau_E \perp \nu_E$.

Finally, we state the boundary conditions data via (8.9) as follows: For $z \in \Gamma_D$ we distinguish between the following cases (i) and (ii) to fulfill the discrete Dirichlet condition at z.
(i) In case $z \in E_1 \cap E_2$ for two distinct edges $E_1, E_2 \subset \Gamma_D$ with linearly independent tangents τ_{E_1} and τ_{E_2} on E_1, E_2, respectively, we consider the 2×2 systems

$$\tau_{E_1} \cdot p_z = (\partial u_D|_{E_1}/\partial s)(z) \quad \text{and} \quad \tau_{E_2} \cdot p_z = (\partial u_D|_{E_2}/\partial s)(z). \tag{8.12}$$

(ii) In the remaining cases $z \in E_1 \cap \Gamma_D$ for $E_1 \in \mathcal{E}_N$ or $z = E_1 \cap E_2$ with two parallel edges $E_1, E_2 \in \mathcal{E}_N$ with the unit tangent vector τ_{E_1} let $p_z \in \mathbb{R}^2$ solve

$$\tau_{E_1} \cdot p_z = (\partial u_D|_{E_1}/\partial s)(z) \quad \text{and} \quad \nu_{E_1} \cdot p_z = \fint_{\omega_z} \nu_{E_1} \cdot p_z \, dx. \tag{8.13}$$

Under these conditions, Theorem 8.1 of [C1, CBa] guarantees reliability for the averaging error estimators η_M and η_A up to higher-order terms h.o.t. which depend on the smoothness of the right-hand sides u_D, f, and g.

Theorem 8.1 ([C1, CBa]). *Suppose that Γ_N is connected and that Γ_D belongs to only one connectivity component of $\partial\Omega$ and let $f|_T \in H^1(T)$ for all $T \in \mathcal{T}$. Then, there exists $(h_T, h_\mathcal{E})$-independent constants $C_{\mathrm{eff}}, C_{\mathrm{rel}}$ (that exclusively depend on the shape of the elements and patches) such that*

$$C_{\mathrm{eff}}\eta_A - \textit{h.o.t.} \leq ||p - p_h||_{L^2(\Omega)} \leq C_{\mathrm{rel}}\eta_A + \textit{h.o.t.} \quad \square \tag{8.14}$$

The a posteriori error control is performed in the function `Aposteriori.m`. The calculation of the error estimator involve the calculation of $(\partial u_D|_E/\partial s)(z) := (\nabla u_D \cdot \tau_E)(z)$. The direction of nodes along the Dirichlet edges are computed and stored in a sparse matrix `DirectionEdge` of dimension $2\ \mathrm{card}(\mathcal{T}) \times 2$. The computation of the fluxes is provided in the function `pEval`.

The following Matlab routines calculate the estimated error.

```
function eta=Aposteriori(element,coordinate,Dirichlet,Neumann,u,pEval)
u_h=zeros(size(coordinate,1),2);
supp_area=zeros(size(coordinate,1),1);
for j=1:size(element,1)
  supp_area(element(j,:))=supp_area(element(j,:))+...
  ones(3,1)*det([1,1,1;coordinate(element(j ,:),:)']);
  u_h(element(j,:),:)=u_h(element(j,:),:)+...
  det([1,1,1;coordinate(element(j,:),:)'])*((pEval(3*(j-1)+[1,2,3],:))'*...
  [4 1 1;1 4 1;1 1 4]/36)';
end
u_h=Tangent(coordinate,Dirichlet,u_h./(supp_area*ones(1,2)));
eta=zeros(size(element,1),1);
```

```
for j=1:size(element,1)
  eta_T(j)=sqrt(det([1,1,1;coordinate(element(j,:),:)'])*...
  (sum(([4 1 1;1 4 1;1 1 4]/6*u_h(element(j,:),:)-...
  pEval(3*(j-1)+[1,2,3],:)).^2')*ones(3,1)/6));
end
eta=sqrt(sum(eta.^2));
```

The function `Aposteriori.m` utilizes the function `Tangent.m` to compute τ_E and the direction of $z \in \Gamma_D$ computed in `Tangent.m` and displayed in the collected algorithm.

The element contribution $\eta_T(j) = ||p_h - Ap_h||_{L^2(T)}$ can be employed in an adaptive algorithm for automatic mesh-refining. The provided Matlab software is fully computable with the adaptive mesh-generation algorithms [CBo] utilized in a numerical experiment of Subsection 9.3 below.

9. NUMERICAL EXAMPLES

The following examples provide the numerical solutions for the displacement u and the flux p for uniform mesh-refinement, and display the errors $||p - p_h||_{L_2(\Omega)}$ and $||u - u_h||_{L_2(\Omega)}$, and the experimental convergence rate

$$\alpha_N := \log(e_{N'}/e_N)/\log(N/N').$$

This is given from the corresponding error e, N' and $e_{N'}$ are the corresponding values of the previous step based on $\mathcal{T}_{k-1}$. Let denote the error of $||u-u_h||_{L_2(\Omega)}$ and $||p-p_h||_{L_2(\Omega)}$ by e_u and e_p, respectively. Since the results are the same for the edge basis function and the Lagrange multiplier technique, the corresponding errors and errors estimator will be only presented once. Below we denote by $N1$ and $N2$ the number of unknown EBmfem and LMmfem.

9.1. **Example on the L-shaped domain.** Let $f := 0$ on the L-shaped domain $\Omega := (-1,1)^2 \setminus [0,1] \times [-1,0]$, $u_D := 0$ on the Dirichlet boundary $\Gamma_D := \{0\} \times [-1,0] \cap [0,1] \times \{0\}$, and on the Neumann boundary $\Gamma_N := \partial\Omega \setminus \Gamma_D$,

$$g(r,\varphi) := 2/3r^{-1/3}(-\sin(\varphi/3), \cos(\varphi/3)) \cdot n$$

in polar coordinates (r,φ); the exact solution of (2.1) is $u(r,\varphi) := r^{2/3}\sin(2\varphi/3)$. The coarsest triangulation $\mathcal{T}_0$ consists of four squares halved by diagonals parallel to the vector $(1,1)$. The user-defined homogeneous functions for the right-hand sides f and `u_D.m` read

```
%f.m
function volumeforce=f(x);
volumeforce=zeros(size(x,1),1);
%u_D.m
function dir=u_D(x);
dir=zeros(size(x,1),1);
```

while `g.m` is given at the end of Subsection 6.3.

9.2. **Example on the disc domain.** Let $f := 1$ on the domain $\Omega = \{(x,y) \in \mathbb{R} : |x| + |y| < 1\} \setminus [0,1] \times \{0\}$. The exact solution of (2.1) is given by $u(r,\varphi) = r^{1/2}\sin(\varphi/2) - 1/2r^2\sin^2(\varphi)$, and the boundary $\Gamma_D := \partial\Omega$. The coarsest triangulation $\mathcal{T}_0$ consists of 4 triangles. The user-defined functions for the right-hand sides f and `u_D.m` read

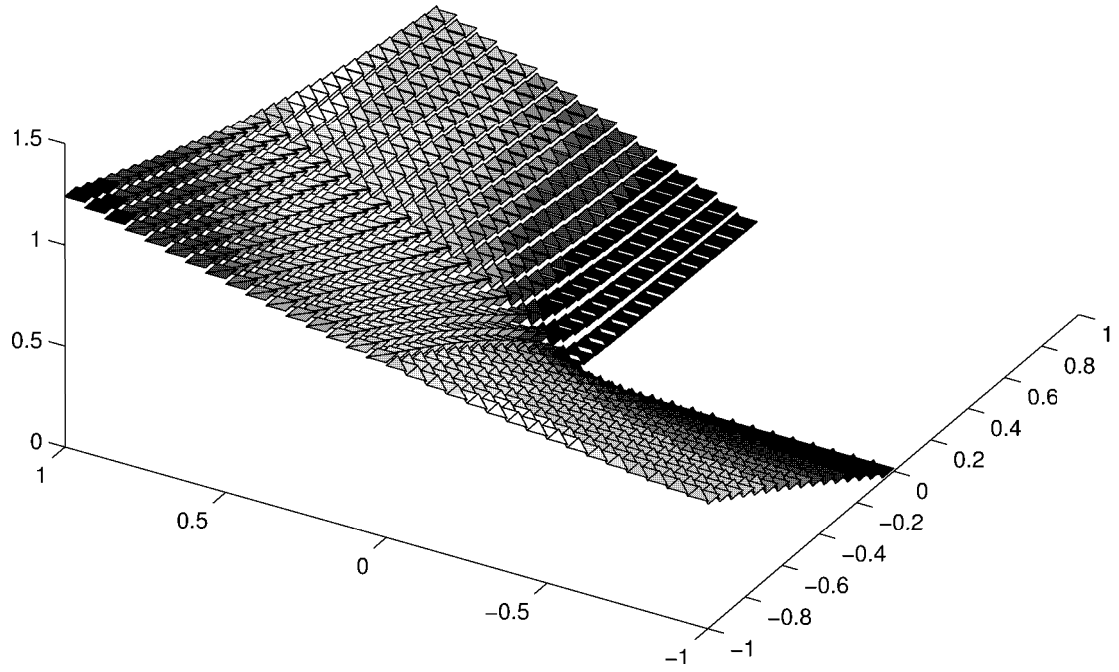

FIGURE 7. Discrete solution u_h for the example in Subsection 9.1.

TABLE 1. Error, error estimator, and the corresponding experimental convergence rates for uniform mesh-refinement in Subsection 9.1.

$N1$	$N2$	e_p	α_N	e_u	α_N	η	α_N
13	29	.44372547		.40360686		.63793932	
56	124	.28475108	.3037	.18344937	.5399	.47052834	.2084
232	512	.18454770	.3051	.08730675	.5223	.29475176	.3290
944	2080	.11881756	.3137	.04232753	.5158	.18617839	.3273
3808	8384	.07594682	.3208	.02073549	.5116	.11771707	.3286
15296	33664	.04829497	.3255	.01022855	.5082	.07438236	.3301
61312	134912	.03060675	.3285	.00506933	.5056	.04696028	.3312

```
%f.m
function volumeforce=f(x);
volumeforce=ones(size(x,1),1);
%u_D.m
function dir=u_D(x);
[a,r]=cart2pol(x(:,1),x(:,2));
ind=find(a<0);
a(ind)=a(ind)+2*pi*ones(size(ind));
dir=r.^(1/2).*sin(a/2)-1/2*(r.^2).*sin(a).*sin(a);
```

9.3. **Adaptive mesh refining.** Automatic mesh refining generates a sequence $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \ldots$ by marking and refining elements and based on red-green-blue refinements and refinement indication by $\eta_T := ||p_h - Ap_h||_{L^2(T)}$. The applied refinement criteria reads: Mark the element $T \in \mathcal{T}_k$ for

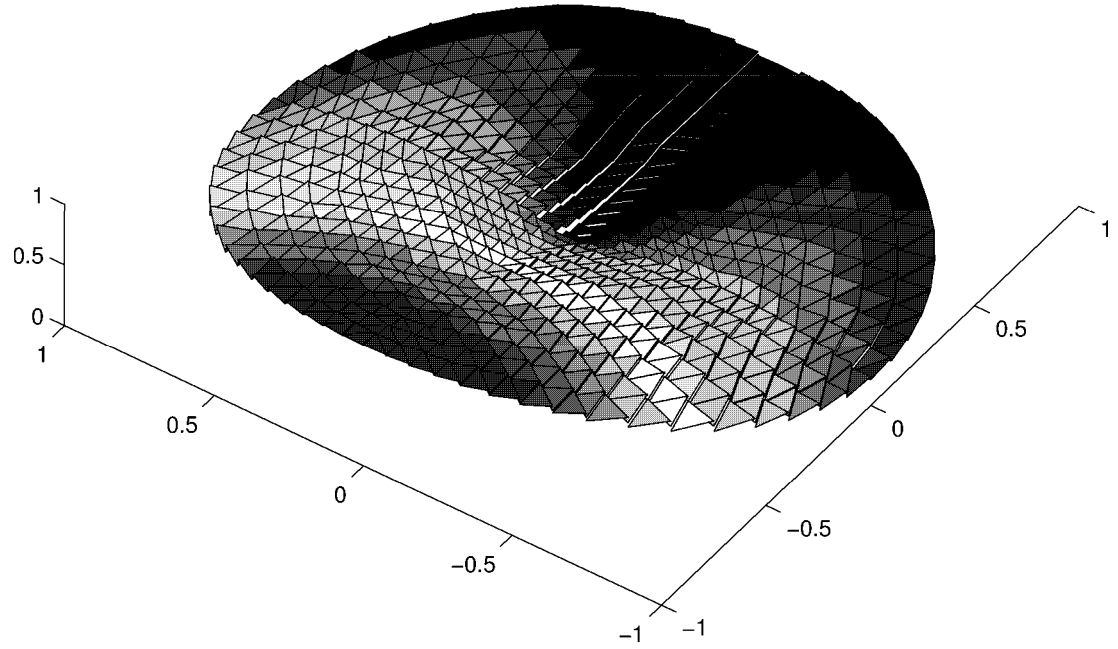

FIGURE 8. Discrete solution u_h for the example in Subsection 9.2.

TABLE 2. Error, error estimator, and the corresponding experimental convergence rates for uniform mesh-refinement in Subsection 9.2.

$N1$	$N2$	e_p	α_N	e_u	α_N	η_N	α_N
13	19	.73153614		.23874909		1.03438275	
46	82	.54718869	.2297	.15339379	.3500	.88569805	.1228
172	340	.38483410	.2668	.08674962	.4321	.69662448	.1820
664	1384	.26438491	.2779	.04519640	.4826	.52063501	.2155
2608	5584	.18295321	.2691	.02289997	.4969	.37806400	.2338
10336	22432	.12777411	.2606	.01150482	.4998	.27097004	.2418
41152	89920	.08976047	.2555	.00576308	.5003	.19297044	.2457

red-refinement if the error indicator η_T satisfies

$$\frac{1}{2}\max_{T'\in\mathcal{T}_k}\eta_A(T') \leq \eta_A(T).$$

The Matlab software in [CBo] is fully compatible with the data structures of this paper and was employed to generate uniform and adaptive mesh-refinements for example in Subsection 9.1. The results are summarized in Figure 9 which displays the error e_p and the error estimator η_A as a function of $N := N1$. We observe an experimental convergence rate $2/3$ and 1 (in terms of a fictive $h := N^{-1/2}$) for uniform and adapted mesh-refining, respectively. This is clear numerical evidence for the superiority of adaptive algorithms.

Acknowledgement. The development of the Matlab routines for mixed FEM started in 1995 at the Technische Hochschule Darmstadt and was continued at Christian-Albrecht University of Kiel. The authors thank Claudia Fischer and Dörte S. Helm for their valuable contributions. Financial

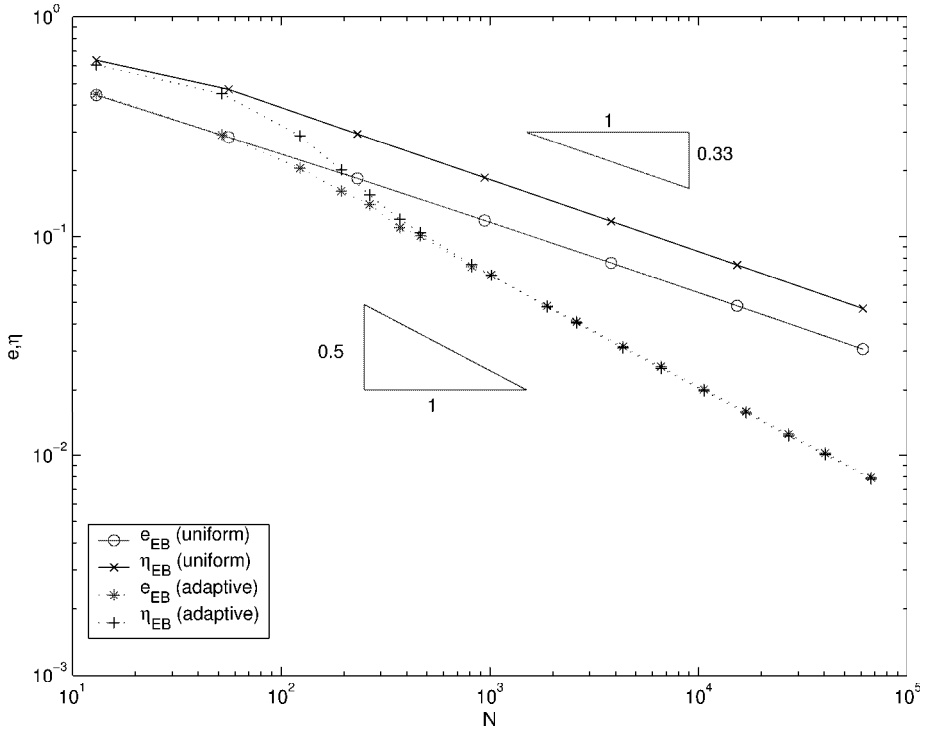

FIGURE 9. Error and error estimator for adaptive mesh-refinement vs. the number N of edges in Section 9.1.

support by the Austrian Sciences Fund (FWF) is thankfully acknowledged. The work has been finalized while the authors were visiting the Isaac-Newton Institute of Mathematical Sciences, Cambridge, England; the support of the EPSRC under grant N09176/01 is thankfully acknowledged.

REFERENCES

[AO] M. AINSWORTH, J.T. ODEN: *A Posteriori Error Estimation in Finite Element Analysis*. John Wiley & Sons, New York, 2001.

[ACF] J. ALBERTY, C. CARSTENSEN, S.A. FUNKEN: *Remarks around 50 lines of Matlab : Finite Element Implementation*. Numerical Algorithms **20** 117-137 (1999).

[ACFK] J. ALBERTY, C. CARSTENSEN, S.A. FUNKEN, R. KLOSE: *Matlab-implementation of the finite element method in elasticity* . Berichtsreihe des Mathematischen Seminars Kiel 00-21 (2000).

[AC] T. ARBOGAST, Z. CHEN: *On the implementation of mixed methods as nonconforming methods for second order elliptic problems*. Math. Comp., **64** (1995).

[AB] D.N. ARNOLD, F. BREZZI: *Mixed and Nonconforming finite element methods: implementations, postprocessing and error estimates*. Mathematical Modeling and Numerical Analysis, **19 (1)**, 7-32 (1985).

[AW] D.N. ARNOLD, R. WINTER: *Nonconforming mixed elements for elasticity*. Mathematical Methods and Models in the Applied Sciences, **12** (2002).

[BaR] I. BABUŠKA, T. STROUBOULIS: *The finite element method and its reliability*. Oxford University Press, 2001.

[B] D. BRAESS: *Finite elements: Theory, fast solver, and application in solid mechanics*. Second Edition, Cambridge University Press, 2001.

[BS] S.C. BRENNER, L.R. SCOTT: *The mathematical theory of finite element methods*. Texts in Applied Mathematics **15** Springer, New York, 1994.

[BF] F. BREZZI, M. FORTIN: *Mixed and hybrid finite element methods*. Springer-Verlag, 1991.

[C1] C. CARSTENSEN: *Some remarks on the history and future of averaging techniques in a posteriori finite element error analysis*. Gamm Workshop 2002. ZAMM to appear.

[C2] C. CARSTENSEN: *All first-order averaging techniques for a posteriori finite element error control on unstructured grids are efficient and reliable* . Math. Comp. to appear.

[CBa] C. CARSTENSEN, S. BARTELS: *Each averaging technique yields reliable a posteriori error control in FEM on unstructured grids part1: Low order conforming, nonconforming, and mixed FEM*. Math. Comp., **71**, 945-969 (2002).

[CBK] C. CARSTENSEN, S. BARTELS, R. KLOSE: *An experimental survey of a posteriori Courant finite element error control for the Poisson equation*. Advances in Computational Mathematics 15: 79-106, 2001.

[CBo] C. CARSTENSEN, J. BOLTE: *Adaptive mesh-refining algorithm for triangular finite elements in Matlab*. In preperation (2003).

[CK] C. CARSTENSEN, R. KLOSE: *Elastoviscoplastic finite element analysis in 100 lines of Matlab*. Journal of Numerical Mathematics, **10 3**, 157-192, 2002.

[Ci] P.G. CIARLET: *The finite element method for elliptic problems error analysis*. North-Holland, Amsterdam, 1978.

[EEHJ] K. ERIKSON, D. ESTEP, P. HANSBO, C. JOHNSON: *Introduction to adaptive methods for differential equations*. Acta Numerica 105-158 (1995).

[M] L.D. MARINI: *An Inexpensive method for the evaluation of the solution of the lowest order Raviart-Thomas mixed method* . SIAM J. Numer. Anal., **22**, 493-496 (1985).

[V] R. VERFÜRTH: *A review of a posteriori estimation and adaptive mesh-refinement techniques*. Wiley-Teubner, 1996.

E-mail address: corinna@aurora.anum.tuwien.ac.at

E-mail address: Carsten.Carstensen@tuwien.ac.at

INSTITUTE FOR APPLIED MATHEMATICS AND NUMERICAL ANALYSIS, VIENNA UNIVERSITY OF TECHNOLOGY, WIEDNER HAUPTSTRASSE 8-10, A-1040 VIENNA, AUSTRIA.