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 \( a, b \in \mathbb{R}^2 \) and \( b \in \mathbb{R} \) on each element and then enforces \( p_h \in H(\text{div}, \Omega) \) 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 \( RT_0(T) \) from Section 4 in the Matlab program EBmfem leads to a linear system of the form

\[
\begin{pmatrix}
B & C \\
C^T & 0
\end{pmatrix}
\begin{pmatrix}
x_\psi \\
x_o
\end{pmatrix}
=
\begin{pmatrix}
b_D \\
b_f
\end{pmatrix}
\]

for given \( b_D \) and \( b_f \) which reflect inhomogeneous Dirichlet boundary conditions and volume forces, and for unknowns \( x_\psi \), the normal components of the flux \( p_h \cdot \nu_E \), which correspond to the basis of \( \mathcal{M}_{h,0} \), and the elementwise constant displacements with components \( x_o \).

The continuity condition \( p_h \in H(\text{div}, \Omega) \) 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

\[
\begin{pmatrix}
B & C & D & F \\
C^T & 0 & 0 & 0 \\
0 & D^T & 0 & 0 \\
0 & 0 & F^T & 0
\end{pmatrix}
\begin{pmatrix}
x_\psi \\
x_\alpha \\
x_{\lambda_M} \\
x_{\lambda_N}
\end{pmatrix} =
\begin{pmatrix}
b_D \\
b_f \\
0 \\
b_g
\end{pmatrix}
\]

with given \(b_D, b_f, b_g\) and unknowns \(x_\psi, x_\alpha, x_{\lambda_M}, x_{\lambda_N}\). 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 \(x_\psi\) while \(x_\alpha\) denote the elementwise constant displacement approximations. The unknown \(x_{\lambda_M}\) are the Lagrange multipliers for the flux boundary conditions on \(\Gamma_N\) while the unknown \(x_{\lambda_N}\) 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

\[
Ax = f
\]

for given \(b\) and unknown \(x\) associated with the volume force and the edge-oriented basis of the nonconforming finite element space \(S_h^{1,\text{NC}}(\mathcal{T})\). 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 \(S^1(\mathcal{T})^d\), the continuous \(\mathcal{T}\)-piecewise affine functions [C1, C2, CBA], to the discrete flux \(p_h\) obtained by an averaging operator \(A : P_h \to S^1(\mathcal{T})^d\) to \(p_h\). That is, for each node \(z \in N\) and its patch \(\omega_z\), \(A(p_h)(z) := A_z p_h\) 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 \(A_z \subset \mathbb{R}^d\) that carries proper boundary conditions (cf. (8.9) below for details).

Then, \(A p_h\) is defined by interpolation with first-order nodal basis functions \((\varphi_z : z \in N)\):

\[
A p_h = \sum_{z \in N} A_z(p_h|_{\omega_z}) \varphi_z
\]

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

\[
C_{\text{eff}} \eta_A - \text{h.o.t.} \leq \|p - p_h\|_{L^2(\Omega)} + \text{h.o.t.} \leq C_{\text{red}} \eta_A + \text{h.o.t.}
\]

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 \((\varphi_E : E \in \mathcal{E})\) 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 $E \in \mathcal{E}_D$. 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 $\eta_A$. 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 $\Omega$ be a bounded Lipschitz domain in the plane with outer unit normal $\nu$ on the polygonal boundary $\Gamma = \Gamma_D \cup \Gamma_N$ split into a relatively open Neumann boundary $\Gamma_N$ 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 $u_D \in H^1(\Omega) \cap C(\Omega)$, seek $u \in H^1(\Omega)$ such that

\begin{equation}
\Delta u + f = 0 \quad \text{in} \ \Omega, \quad u = u_D \quad \text{on} \ \Gamma_D, \quad \nabla u \cdot \nu = g \quad \text{on} \ \Gamma_N.
\end{equation}

(2.1)

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

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

\begin{equation}
\text{div } p + f = 0 \quad \text{and} \quad p = \nabla u \quad \text{in} \ \Omega
\end{equation}

(2.2)

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

$$H(\text{div}, \Omega) := \{q \in L^2(\Omega)^2 : \text{div } q \in L^2(\Omega)\},$$

$$H_{0,N}(\text{div}, \Omega) := \{q \in H(\text{div}, \Omega) : q \cdot \nu = 0 \text{ on } \Gamma_N\},$$

$$H_{0,N}(\text{div}, \Omega) := \{q \in H(\text{div}, \Omega) : q \cdot \nu = g \text{ on } \Gamma_N\}.$$
Then, the weak formulation of (2.2) reads: Given \( f \in L^2(\Omega) \), \( g \in L^2(\Gamma_N) \), and \( u_D \in H^1(\Omega) \cap C(\Omega) \), seek \( p \in H_{0,N}(\text{div}, \Omega) \) and \( u \in L^2(\Omega) \) such that

\[
\begin{align*}
&\int_{\Omega} p \cdot q \, dx + \int_{\Omega} u \, \text{div} \, q \, dx = \int_{\Gamma_D} u_D \, q \cdot \nu \, ds \quad \text{for all } q \in H_{0,N}(\text{div}, \Omega), \\
&\int_{\Omega} v \, \text{div} \, p \, dx = -\int_{\Omega} vf \, dx \quad \text{for all } v \in L^2(\Omega)
\end{align*}
\]  

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, \S II, Thm. 1.2]).

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

\[
RT_0(T) := \{ q \in L^2(T) : \forall T \in T \, \exists a \in \mathbb{R}^2 \, \exists b \in \mathbb{R} \, \forall x \in T, q(x) = a + bx \quad \text{and} \quad \forall E \in \mathcal{E}_T, |q|_E \cdot \nu_E = 0 \},
\]

where \( T \) is a regular triangulation (cf. Section 3), \( \mathcal{E}_T \) 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 \( T \).

The continuity of the normal components on the boundaries reflects the conformity \( RT_0(T) \subset H(\text{div}, \Omega) \) (as defined below). For the second approach with EBfem, this continuity is built in the shape function \( \psi_E \) along \( E \in \mathcal{E}_T \). In the hybrid formulation, the continuity of \( E \in \mathcal{E}_T \) 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

\[
\begin{align*}
M_{h,a} &:= \{ q_h \in RT_0(T) : q_h \cdot \nu = g_h \text{ on } \Gamma_N \}, \\
M_h &:= M_{h,0} = RT_0(T) \cap H_{0,N}(\text{div}, \Omega), \\
L_h = P_0(T) &:= \{ v_h \in L^2(\Omega) : T \in T, v_h|_T \in P_0(T) \}.
\end{align*}
\]

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

\[
\begin{align*}
&\int_{\Omega} p_h \cdot q_h \, dx + \int_{\Omega} u_h \, \text{div} \, q_h \, dx = \int_{\Gamma_D} u_D \, q_h \cdot \nu \, ds \quad \text{for all } q_h \in M_{h,0}, \\
&\int_{\Omega} v_h \, \text{div} \, p_h \, dx = -\int_{\Omega} v_h f \, dx \quad \text{for all } v_h \in L_h.
\end{align*}
\]

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

In Section 4 we will define an edge-oriented basis \((\psi_j \mid j = 1, \ldots, N)\) of \( RT_0(T) \) with \( M_{h,0} = \text{span}\{ \psi_1, \ldots, \psi_M \} \subset RT_0(T) \). With respect to this basis (possibly in a different order of the indices), the components \( x_\psi = (x_1, \ldots, x_N) \) of \( \psi_h = \sum_{k=1}^N x_k \psi_k \in M_{h,a} \) and \( x_u = (x_{N+1}, \ldots, x_{N+L}) \) of \( u_h|_{\Gamma_\ell} = x_{N+\ell} \) for \( \ell = 1, \ldots, L \) and for an enumeration \( T = \{ T_1, \ldots, T_L \} \) of the \( L = \text{card}(T) \) elements. Then, (2.5)-(2.6) are recast into the linear system of equations for the
unknown \((x_1, \ldots, x_M)\) and \((x_{N+1}, \ldots, x_{N+L})\)

\[(2.7)\]
\[
\sum_{k=1}^{M} x_k \int_{\Omega} \psi_j \cdot \psi_k \, dx + \sum_{\ell=1}^{L} x_{N+\ell} \int_{T_\ell} \text{div} \, \psi_j \, dx = \int_{\Gamma_D} u_D \psi_j \cdot \nu \, ds - \sum_{m=M+1}^{N} g_h|\Omega_m \int_{\Omega} \psi_j \cdot \psi_m \, dx,
\]

\[(2.8)\]
\[
\sum_{k=1}^{M} x_k \int_{T_\ell} \text{div} \, \psi_k \, dx = - \int_{T_\ell} f \, dx - \sum_{m=M+1}^{N} g_h|\Omega_m \int_{T_\ell} \text{div} \, \psi_k \, dx
\]

for \(j = 1, \ldots, M, \ell = 1, \ldots, L\) and the known \((x_{M+1}, \ldots, x_N) := (g_h|\Omega_m, m = M + 1, \ldots, N)\). 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 \(T\) and the edges \(E\) and their data representation.

#### 3.1. Geometric Description

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

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

of all edges in \(T\) is partitioned into edges on the Dirichlet boundary \(\mathcal{E}_D := \{E \in E : E \subset \Gamma_D\}\), on the Neumann boundary \(\mathcal{E}_N := \{E \in E : E \subset \Gamma_N\}\), 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 \(\bigcup E = \bigcup_{E \in E} E = \{x \in \Omega : \exists T \in T, x \in \partial T\}\).

All the Matlab programs employ the following data representation of \(T, N\) and \(E\). The \(n := \text{card}(N)\) nodes \(N = \{z_1, \ldots, z_n\}\) 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 \(T_j = \text{conv}\{z_k, z_\ell, z_m\}\) is the convex hull of its three vertices \(z_k, z_\ell, z_m\) in \(N\) described by the global numbers \(k, \ell, m\) 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 \(k, \ell, m\) of the three nodes is counterclockwise. The information of the edge \(E = \text{conv}\{z_k, z_\ell\}\) of \(\mathcal{E}_D\) and \(\mathcal{E}_N\) is stored in the data files Dirichlet.dat and Neumann.dat, respectively, represent \(E = \text{conv}\{z_k, z_\ell\}\) in row \(j\) by the two (global) numbers \(k, \ell\). It is a convention that the tangential unit vector \(\tau_E\) along \(E\) points from \(z_k\) to \(z_\ell\) and that the outer normal \(\nu_E\) points to the right. Figure 1 and 2 display a triangulation \(T\) and its data representation in coordinate.dat, element.dat, and Dirichlet.dat.

The initialization of coordinate.dat, element.dat, Dirichlet.dat, and Neumann.dat 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).

<table>
<thead>
<tr>
<th>coordinates.dat</th>
<th>element.dat</th>
<th>Dirichlet.dat</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>1 2 5</td>
<td>1 2</td>
</tr>
<tr>
<td>0.5 0</td>
<td>1 5 4</td>
<td>2 3</td>
</tr>
<tr>
<td>1 0</td>
<td>2 3 6</td>
<td>3 6</td>
</tr>
<tr>
<td>0 0.5</td>
<td>2 6 5</td>
<td>6 9</td>
</tr>
<tr>
<td>0.5 0.5</td>
<td>4 5 8</td>
<td>9 8</td>
</tr>
<tr>
<td>1 0.5</td>
<td>4 8 7</td>
<td>8 7</td>
</tr>
<tr>
<td>0.5 1</td>
<td>5 6 9</td>
<td>7 4</td>
</tr>
<tr>
<td>1 1</td>
<td>5 9 8</td>
<td>4 1</td>
</tr>
</tbody>
</table>

Figure 2. Data files coordinates.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 \( \nu_E \) be a unit normal which coincides with the exterior unit normal \( \nu = \nu_E \) along \( \Gamma \) if \( E \in \mathcal{E}_D \cup \mathcal{E}_N. \) Given any \( E \in \mathcal{E}_\Omega \) and any \( T \)-piecewise continuous function \( \rho \in L^2(\Omega; \mathbb{R}^3), \) let \( J_E := [\rho \cdot \nu_E] \) denote the jump of \( \rho \) across \( E \) in the direction \( \nu_E \) 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 \( T_+, T_- \in T \) such that \( \nu_E \) 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 \( u_\text{D}=0 \). 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 \( E = \partial T_+ \cap \partial T_- \) with initial node \( A \) and end node \( B \) and unit normal \( \nu_E \). The orientation of \( \nu_E \) 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 \( \text{nodes2edge}, \text{nodes2element} \), and \( \text{edges2element} \) are computed in the function \texttt{edge.m}

\[
\text{function } \{ \text{nodes2element}, \text{nodes2edge}, \text{noedges}, \text{edge2element}, \ldots \} \\
\quad \text{interioredge}=\text{edge(element,coordinate)};
\]

The corresponding data structures and computation of the three matrices \( \text{nodes2edge}, \text{nodes2element} \), and \( \text{edges2element} \) are given in Subsection 3.2.1-3.2.3.

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

\[
\text{nodes2element}(k,\ell) = \begin{cases} 
  j & \text{if } (k,\ell) \text{ are numbers of nodes of element number } j; \\
  0 & \text{otherwise}.
\end{cases}
\]
Notice carefully that the two neighbouring elements $T_+$ and $T_-$ as depicted in Figure 4 that share one common edge $\text{conv}(A, B)$ with endpoints $A = \text{coordinate}(k)$ and $B = \text{coordinate}(\ell)$ have the number $j_+ = \text{nodes2element}(k, \ell)$ and $j_- = \text{nodes2element}(\ell, k)$; i.e. $T_+$ has number $j_+$. The computation of the matrix $\text{nodes2element}$ in Matlab reads

\begin{verbatim}
% Matrix nodes2element
nodes2element=sparse(size(coordinate,1),size(coordinate,1));
for \( i=1:\text{size}($element,1)$
  nodes2element(element(\( i,: \)),element(\( i,[2 \ 3 \ 1]) \))= ...
  nodes2element(element(\( i,: \)),element(\( i,[2 \ 3 \ 1]) \))=i*eye(3,3);
end
\end{verbatim}

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

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

The computation of the matrix $\text{nodes2edge}$ in Matlab reads

\begin{verbatim}
% 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';
\end{verbatim}

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

3.2.3. Matrix $\text{edges2element}$. The $(\text{noedges} \times 4)$ matrix $\text{edges2element}$ represents the initial node $k$ and the end node $\ell$ 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 $\text{edges2element}$ 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 $\text{edges2element}(j,3)$ defines $T_+$ and hence the orientation of the normal $\nu_E$ of the edge $E = T_+ \cap T_-$ number $j$ throughout all the Matlab programs. In case of an exterior edge $E \in E_D \cup E_N$ the fourth entry is zero [thereby, $\nu_E$ is exterior to $\Omega$].

$$\text{edges2element}(j,[3,4]) = \begin{cases} [m, n] & \text{if a common edge } j \text{ belongs to elements } m, n; \\
[m, 0] & \text{if an edge } j \text{ belongs to an element } m. \end{cases}$$

The computation of the matrix $\text{edges2element}$ in Matlab reads

\begin{verbatim}
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) ...
    \end{verbatim}
nodes2element(end node, initial node); end end

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

\begin{align*}
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
\end{align*}

**Figure 5.** Matrix \( \text{edge2element} \) generated by the function \( \text{edge.m} \) 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 \( \psi_E \).** 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 \( \psi_E \)). 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 \( \nu_E \), denote the unit normal vector of \( E_j \) chosen with a global fixed orientation while \( \nu_j \) denotes the outer unit normal of \( T \) along \( E_j \). Define

\[
\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T|} (x - P_j) \quad \text{for } j = 1, 2, 3 \text{ and } x \in T,
\]

where \( \sigma_j = \nu_j \cdot \nu_{E_j} \) is +1 if \( \nu_{E_j} \) points outward and otherwise −1; \( |E_j| \) is the length of \( E_j \), and \( |T| \) is the area of \( T \).

\[
2|T| = \det (P_2 - P_1, P_3 - P_1) = \det \begin{pmatrix} P_1 & P_2 & P_3 \\ 1 & 1 & 1 \end{pmatrix}
\]
Figure 6. Triangle $T = \text{conv}\{P_1, P_2, P_3\}$ with vertices $P_1, P_2, P_3$ (ordered counterclockwise) and opposite edges $E_1 = \text{conv}\{P_2, P_3\}$, $E_2 = \text{conv}\{P_1, P_3\}$, $E_3 = \text{conv}\{P_1, P_2\}$ 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 $3 \times 3$-matrix that consists of the $2 \times 3$ 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 $\nu_E$ points outward from $T_+$ to $T_-$ (ct. 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 $\psi_E$). Given an edge $E \in \mathcal{E}$ there are either two elements $T_+$ and $T_-$ in $T$ with the joint edge $E = \partial T_+ \cap \partial T_-$ or there is exactly one element $T_+$ in $T$ 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

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

**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 $RT_0(T)$;

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

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

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

For any $F \in E_\Omega$ there follows $[\psi_E]_F \cdot \nu_F = 0$ from (a). Hence $\psi_E \in RT_0(T) \subseteq H(\text{div}, \Omega)$.\n
(c) The functions $(\psi_E : E \in \mathcal{E})$ are (uniquely) determined by $\psi_E \in RT_0(T)$ 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 $q_h \in RT_0(T)$ notice that $q_h \cdot \nu_E$ is constant on $E \in \mathcal{E}$ and define

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

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

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

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

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

we deduce with (a) that $0 = p_h|_F \cdot \nu_F = x_F$.\n
(d) This is immediate from (4.3). $\square$

4.2. Local stiffness matrices. In this subsection, we recall notation from Figure 6 for a triangle $T$ with edges $E_1, E_2, E_3$ 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 $B_T, C_T \in \mathbb{R}^{3 \times 3}$ be defined by

\begin{equation}
(B_T)_{jk} := \int_T \psi_j \cdot \psi_k \, dx \text{ for } j, k = 1, 2, 3,
\end{equation}

\begin{equation}
C_T := \text{diag}\left(\int_T \text{div} \psi_1 \, dx, \int_T \text{div} \psi_2 \, dx, \int_T \text{div} \psi_3 \, dx\right)
\end{equation}

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

$$M := \begin{pmatrix}
2 & 0 & 1 & 0 & 0 & 0 \\
0 & 2 & 0 & 1 & 0 & 1 \\
1 & 0 & 2 & 0 & 1 & 0 \\
0 & 1 & 0 & 2 & 0 & 1 \\
1 & 0 & 1 & 0 & 2 & 0 \\
0 & 1 & 0 & 1 & 0 & 2
\end{pmatrix} \in \mathbb{R}^{6 \times 6} \text{ and } N := \begin{pmatrix}
0 & P_1 - P_2 & P_1 - P_3 \\
P_2 - P_1 & 0 & P_2 - P_3 \\
P_3 - P_1 & P_3 - P_2 & 0
\end{pmatrix} \in \mathbb{R}^{3 \times 3}.$$
there holds

\[(4.6) \quad B_T = \frac{1}{48|T|} C_T^T N^T M N C_T.\]

**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

\[
B_{jk} = \int_T \psi_j \cdot \psi_k \, dx = \sigma_{E_j}\sigma_{E_k} \frac{|E_j||E_k|}{4|T|^2} \sum_{\ell = 1}^3 \int_T \lambda_\ell (P_\ell - P_j) \cdot \lambda_m (P_m - P_k) \, dx
\]

\[
= \frac{\sigma_{E_j}|E_j|\sigma_{E_k}|E_k|}{4|T|^2} \sum_{\ell = 1}^3 (P_\ell - P_j) \cdot (P_m - P_k) \int_T \lambda_\ell \lambda_m \, dx.
\]

Since \(\int_T \lambda_\ell \lambda_m \, dx = \frac{|T|}{15}(1 + \delta_{\ell m}),\) this yields

\[
(B_T)_{jk} = \frac{\sigma_{E_j}|E_j|\sigma_{E_k}|E_k|}{4|T|^2} \sum_{\ell = 1}^3 (P_\ell - P_j) \cdot (P_m - P_k) \left(\frac{|T|}{12}(1 + \delta_{\ell m})\right)
\]

\[
= \frac{\sigma_{E_j}|E_j|}{48|T|} \left(\sum_{\ell = 1}^3 (P_\ell - P_j)\right) \cdot \left(\sum_{m = 1}^3 (P_m - P_k)\right)
\]

\[
+ \sum_{\ell = 1}^3 (P_\ell - P_j) \cdot (P_\ell - P_k) \sigma_{E_k}|E_k|.
\]

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

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

```matlab
function B=stimaB(coord);
N=coord(:,1:3)-repmat(coord(:,1:3),1,3);
C=diag([norm(N([1,1,2])),norm(N([1,2,3])),norm(N([1,2,3]))]);
M=spdiaqs([ones(3,1),ones(3,1),zeros(3,1),ones(3,1),ones(3,1)],... [-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 \(\sigma_{E_3}\) 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 \(I=\text{nodes2edge}(\text{element}(j,[2\ 3\ 1]),\text{element}(j,[3\ 1\ 2]))\) gives the vector \((m_1, m_2, m_3)\) of global edge numbers. The sign \(\sigma_{E_{m_{\nu}}}\) of the global edge number \(m_{\nu}\) with respect to the current element \(T\) with outer unit normal \(\nu_T\) is \(\sigma_{E_{m_{\nu}}} = \nu_{E_{m_{\nu}}} \cdot \nu_T\) for \(k = 1,2,3.\) The sign
\[ \sigma_{E_{mk}} = \pm 1 \text{ is negative if and only if } j = \text{edge2element}(m_k, 4). \] 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

\[ B(T) \begin{pmatrix} m_1, m_2, m_3 \end{pmatrix} = \text{diag} \left( \sigma_{E_{m_1}}, \sigma_{E_{m_2}}, \sigma_{E_{m_3}} \right) \text{stimaB(coord) diag} \left( \sigma_{E_{m_1}}, \sigma_{E_{m_2}}, \sigma_{E_{m_3}} \right) \]

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

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

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

```matlab
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 \( \Gamma_N \) to relax the continuity required on the normal component. Set

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

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

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 \((v_h, q_h, \mu_h, m_h) \in L_h \times M_h \times \Lambda_h \times N_h\)

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

(5.4) \[ \int_\Omega v_h \cdot \text{div} p_h \, dx = - \int_\Omega v_h f \, dx, \]

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

(5.6) \[ - \int_{\Gamma_N} [p_h \cdot \nu] \cdot m_h \, ds = - \int_{\Gamma_N} g \cdot m_h \, ds. \]
Throughout this section, given any triangle $T$ with the center of gravity $(x_T, y_T)$ we consider the shape functions

\begin{equation}
\psi_1 = (1, 0), \quad \psi_2 = (0, 1), \quad \psi_3 = (x - x_T, y - y_T), \quad \text{for} \ x \in T.
\end{equation}

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

\begin{equation}
(B_T)_{jk} := \int_T \psi_j \cdot \psi_k \, dx \quad \text{for} \ j, k = 1, 2, 3.
\end{equation}

\begin{equation}
(C_T)_j := \int_T \text{div} \ \psi_j \, dx \quad \text{for} \ j = 1, 2, 3.
\end{equation}

**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

\[ B = |T| \ \text{diag}(1, 1, s/36) \ \text{and} \ C_T = (0, 0, 2|T|). \]

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

\[(B_T)_{11} = (B_T)_{22} = |T|, \quad (B_T)_{33} = \int_T \left( (x - x_T)^2 + (y - y_T)^2 \right) \, dy = 1/36 \ |T| \ s. \]

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

\[(C_T)_j = \int_T \text{div} \ \psi_j \, dx = 0 \quad \text{for} \ j = 1, 2, \quad \text{and} \ (C_T)_3 = 2|T|. \quad \Box \]

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

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

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

\begin{equation}
\begin{pmatrix}
B_1 & 0 & \cdots & 0 \\
0 & B_2 & \cdots & : \\
\vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & B_{\text{card}(T)}
\end{pmatrix}
\quad \text{and} \quad
\begin{pmatrix}
C_1 & 0 & \cdots & 0 \\
0 & C_2 & \cdots & : \\
\vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & C_{\text{card}(T)}
\end{pmatrix}.
\end{equation}
\textbf{Definition 5.3.} For each \( E_k \in \mathcal{E}_t \), the local vector \( D_k \in \mathbb{R}^{b \times 1} \) for the Lagrange multiplier technique is defined by

\begin{equation}
D_{kj} = -\int_{E_k} \psi_j \cdot \nu_{E_k} \; ds \quad \text{for } j = 1, \ldots, 6.
\end{equation}

where \( \psi_1, \psi_2, \psi_3 \) and \( \psi_4, \psi_5, \psi_6 \) denote the three shape functions in \( RT_0(T_+) \) and \( RT_0(T_-) \), respectively.

Recall the notation and orientation of Figure 4 and define

\begin{equation}
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\}
\end{equation}

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).

\textbf{Lemma 5.4.} For each \( E_k \in \mathcal{E}_o \), \( k = 1, \ldots, M \), 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

\begin{equation}
D_k = \begin{pmatrix} -EV(k)_y & EV(k)_x & -h & EV(k)_y & -EV(k)_x & h \end{pmatrix}^T.
\end{equation}

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

\[
-D_{k1} = D_{k4} = \int_{E_k} \psi_1 \cdot \nu_k \; ds = \int_{E_k} \nu_x(k) \; ds = -\nu_x(k)|E_k| = EV(k)_y,
\]

\[
-D_{k2} = D_{k5} = \int_{E_k} \psi_2 \cdot \nu_k \; ds = \int_{E_k} \nu_y(k) \; ds = -\nu_y(k)|E_k| = EV(k)_x,
\]

\[
-D_{k3} = D_{k6} = \int_{E_k} \psi_3 \cdot \nu_k \; ds = \int_{E_k} (y_T - y, x_0 - x_T) \cdot EV(k)/|E_k| \; ds =: h. \quad \square
\]

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

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

With the sublist \( I \) of \texttt{edge2element} of all interior edges, the Matlab realization of the global matrix \( D \) reads

\begin{verbatim}
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 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])=EV(k,2);EV(k,1);-h(1);EV(k,2);-EV(k,1);h(2));
end
\end{verbatim}

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 ingredients 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 \( I_\ell \) with centre of gravity \( z_{I_\ell} \), namely

\[
- \int_{I_\ell} f(x) \, dx \quad \text{for } \ell = 1, \ldots, L.
\]

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

\[
b_{f_\ell} := -|T_\ell| f(z_{I_\ell})
\]

which in Matlab reads

\[
\text{det}([[1 1 1; coordinate(element(1,:),:)',]) * \ldots 
\text{f(sum(coordinate(element(1,:),:))/3)}/\ell}
\]

The global vector \( b \) (of length \( N + L \) or \( 4 \times L + M \) 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 \( b(\text{nedges} + \ell) \); and for (1.2) the entry (6.2) equals \( b(4 \times \text{size(element)} + \ell) \) 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 BMfem.** Since the normal components of the test functions are zero or equal one along the edge \( E \in \Eu_D \) of number \( j \) with mid-point \((x_M, y_M)\), a simple one-point integration reads

\[
b_D := u_D(x_M, y_M) |E| \approx \int_E u_D \, ds.
\]

Given the values of \( u.D \) in a user-specified function \( u.D.m \), the Matlab realization of (6.3) reads for \( k=1:\text{size(Dirichlet,1)} \)

\[
\text{b(nodes2edge(Dirichlet(k,1),Dirichlet(k,2)))=...} 
\text{norm(coordinate(Dirichlet(k,1,:))-coordinate(Dirichlet(k,2,:)))}\ldots 
\text{u.D(sum(coordinate(Dirichlet(k,:),:))/2));}
\end
\]

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

\[
b_D := \sum_{k=1}^{\text{card}(\Eu_D)} u_D(x_M, y_M) \left( EV(k)_y, -EV(k)_x, (y_T - y_0, x_0 - x_T) \cdot EV(k) \right)
\]

\[
\approx \sum_{k=1}^{\text{card}(\Eu_D)} \int_{E_k} u_D \psi_j \cdot \nu_k \, ds
\]

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,:,:),:)-
        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 EBmef. Let $B_1 := B_{1,\ldots,M}^{(1,\ldots,M)}$, $B_2 := B_{M+1,\ldots,\cdot,N}^{(M+1,\ldots,N)}$, and $B_3 := B_{M+1,\ldots,N}^{(M+1,\ldots,N)}$ 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,\ldots,M) \\
x_\psi(M+1,\ldots,N) \\
x_u
\end{pmatrix}
= \begin{pmatrix}
b_D(1,\ldots,N) \\
b_f
\end{pmatrix}.
\]$$

(6.5)

Therein, $x_\psi(1,\ldots,M)$ is the vector of the unknowns at the free edges $E_\Omega$ to be determined and $x_\psi(M+1,\ldots,N)$ 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

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

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

\begin{verbatim}
if isempty(Neumann; 
    tmp=zeros(noedges+size(element,1),1); 
    tmp(diag(noedges2edge(Neumann(:,1),Neumann(:,2))))=... 
        ones(size(diag(noedges2edge(Neumann(:,1),Neumann(:,2)))),1),1); 
    FreeEdge=find(tmp); 
    x=zeros(noedges+size(element,1),1); 
    CN=coordinate(Neumann(:,2,:))-coordinate(Neumann(:,1,:)); 
    for i=1:size(Neumann,1)
        x(noedges2edge(Neumann(i,1),Neumann(i,2)))=... 
            sum(coordinate(Neumann(i,:,:),:))/2,CN(i,:)*[0,-1;1,0]/norm(CN(i,:))); 
    end 
    b=b-A*x; 
end
\end{verbatim}

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

$$
x(FreeEdge)=A(FreeEdge,FreeEdge)\backslash b(FreeEdge)
$$

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

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

(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

\[
F_1 = \int_{E_k} \psi_1 \cdot \nu_k \, ds = \int_{E_k} \nu_x(k) \, ds = \nu_x(k) |E_k| = EV(k)_y,
\]

\[
F_2 = \int_{E_k} \psi_2 \cdot \nu_k \, ds = \int_{E_k} \nu_y(k) |E_k| \, ds = \nu_y(k) |E_k| = -EV(k)_x,
\]

\[
F_3 = \int_{E_k} \psi_3 \cdot \nu_k \, ds = \frac{1}{|E_k|} \int_{E_k} (y_T - y, x - x_T) \cdot EV(k) \, ds
= (y_T - y, x - x_T) \cdot EV(k) =: h, \quad \text{where } (x, y) \in E_k \in \mathcal{E}_N.
\]

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

```matlab
if isempty(Neumann);
    F=sparse(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))];
end

% 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
```

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

```matlab
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 $\nabla u_{NC}$ 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

\[
P_1(T) = \{ f \in L^2(\Omega) : \forall T \in \mathcal{T}, f|_T \in P_1(T) \}.
\]

(7.1)

\[
S_{1,NC}(T) = \{ v \in P_1(T) : v \text{ are continuous in all midpoints } z_E \text{ of edges } E \in \mathcal{E}_\Omega \}.
\]

\[
S_{1,NC}^D(T) = \{ v \in S_{1,NC}(T) : v(z_E) = 0 \text{ for all } E \in \mathcal{E}_D \}.
\]

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

(7.2)

\[
\int_{\Omega} \nabla_T u_{NC} \cdot \nabla_T v_h = \int_{\Omega} f_h v_h \, dx + \int_{\Gamma_N} g_h v_h \, ds \quad \text{for all } v_h \in S_{1,NC}^D(T).
\]

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

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

(7.3)

\[
p_h = \nabla_T u_{NC} + b_h.
\]

**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 \( p_h \in H(\text{div}, \Omega) \) satisfies

\[
\int_{\Omega} \text{div} \, p_h \, dx = -\int_{\Omega} f \, dx
\]

and so [with \( \text{div} \, p_h = b_T n \),

\[
f_T := \int_{T} f(x) \, dx = -b_T n \quad \text{whence} \quad b_T = -f_T/n.
\]

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

\[
\int_{\psi_E} p_h \cdot \nu[\psi_E] \, ds = 0.
\]

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

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

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

\[
\int_{\Gamma_N} g_h \psi_E \, ds = \int_{\psi_E} p_h \cdot \nu[\psi_E] \, ds
\]

\[
= \int_{\Omega} \psi_E \text{div} \, p_h \, dx + \int_{\Omega} p_h \cdot \nabla_T \psi_E \, dx.
\]

Since \( \text{div} \, p_h = -f_h \) in \( \Omega \) 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 $\nabla_T \psi_E$ is constant on each $T$ and so
\[
\int_{\Omega} p_h \cdot \nabla_T \psi_E \, dx = \sum_{T \in T} |T| a_T \cdot \nabla_T \psi_E|_T.
\]
Let $a_{h}$ denote the $T$-piecewise constant values of $(a_T : T \in T)$, i.e. $a_{h}|_T := a_T$ for all $T \in T$. Since $u_{NC} \in S_{D}^{1,NC}(T)$ solves (7.2) there holds
\[
\int_{\Omega} (a_{h} - \nabla_T u_{NC}) \cdot \nabla_T v_{h} \, dx = 0 \quad \text{for all } v_{h} \in S_{D}^{1,NC}(T).
\]
Define $a_{h} - \nabla_T u_{NC} =: c_{h} \in P_{0}(T)^{n}$. Then
\[
0 = \int_{\Omega} c_{h} \cdot \nabla_T \psi_E \, dx = \int_{\Omega} [c_{h} \cdot \nu \psi_E] \, ds
\]
\[
= \int_{E} \nu_{E} \cdot [c_{h}] \, ds + \int_{(\Omega \setminus E) \cap E} [c_{h} \cdot \nu \psi_E] \, ds.
\]
Since the second integral vanishes $[c_{h}] \cdot \nu = 0$ on $E$. This holds for all $E \in \mathcal{E}_{\Omega}$. Thus $c_{h} \in H(\text{div}, \Omega)$ and $c_{h} \cdot \nu = 0$ on $\Gamma_{N}$. Hence $c_{h}$ is a proper test function in the first equation for $p_{h}$, namely
\[
\int_{\Omega} p_{h} \cdot c_{h} \, dx + \int_{\Omega} \text{div} \, c_{h} \, p_{h} \, dx = 0.
\]
Observe that $\text{div} \, c_{h} = 0$ as $c_{h} \in H(\text{div}, \Omega) \cap P_{0}(T)^{n}$. 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
\[
\int_{\Omega} c_{h} \cdot \nabla_T u_{NC} \, dx = 0.
\]
Since $c_{h} = a_{h} - \nabla_T u_{NC}$ this shows $0 = \int_{\Omega} c_{h} \cdot c_{h} \, dx = 0$, i.e. $a_{h} \equiv \nabla_T u_{NC}$. \hfill \square

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 $T \in T$ denote by $M^{NC}$ and $M$ the local stiffness matrix of $P1$ nonconforming finite element and $P1$ conforming finite element, respectively. Then there holds
\[
(7.4) \quad M^{NC} = 4 \, M.
\]

**Proof.** For a triangle $T \in T$ set $P_{j} := (x_{j}, y_{j})$ for $1 \leq j \leq 3$ and let $z := (\bar{x}_{j}, \bar{y}_{j}) = (x_{j+1} + x_{j+2}, y_{j+1} + y_{j+2})/2$ be the mid points on edges. Here the indices are modulo 3. Then
\[
(7.5) \quad \frac{y_{j} - y_{j+1}}{\bar{x}_{j+1} - \bar{x}_{j}} = \frac{1}{2} \left( y_{j+1} + y_{j+2} - y_{j} - y_{j+3} \right) = \frac{1}{2} \left( y_{j+1} + y_{j+2} - y_{j} - y_{j+3} \right).
\]
Since $T = \text{conv} \,(P_{j}, z_{j})$ for $1 \leq j \leq 3, \, |T| = |T| \text{conv} \,(P_{j}, z_{j}) = 4 |T| \text{conv} \,(z_{1}, z_{2}, z_{3})$. With this, (7.5) and by [ACF] it follows that $M^{NC} = 4 \, M$. \hfill \square

The Matlab realization of the stiffness matrix $M^{NC}$ in (7.4) via a result from [ACF] reads
function M = 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 \( u_{NC} \in S^{1,NC}(T) \) from CR_FEM.m, Theorem 7.1 allows the computation of \( p_h(x) := \nabla u_{NC} - f_h(x - x_T)/2 \) for \( x \in T \in T \). Let \((\varphi_E : E \in \mathcal{E})\) be the edge-oriented first-order nonconforming basis functions of \( S^{1,NC}(T) \) and let \((\bar{x}_j, \bar{y}_j)\) be the midpoint of edge \( E_j \) for a local enumeration \( E_1, E_2, E_3 \) of the edges of \( T \). Based on

\[
\begin{pmatrix}
\nabla \varphi_{E_1} \\
\nabla \varphi_{E_2} \\
\nabla \varphi_{E_3}
\end{pmatrix}
= \begin{pmatrix}
1 & 1 & 1 \\
\bar{x}_1 & \bar{x}_2 & \bar{x}_3 \\
\bar{y}_1 & \bar{y}_2 & \bar{y}_3
\end{pmatrix}^{-1}
\begin{pmatrix}
0 & 0 \\
1 & 0 \\
0 & 1
\end{pmatrix},
\]

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

```matlab
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(j*size(element, 1), 2);
for i = 1:size(element, 1)
    I = diag(nodes2edge(element(j, [1 2 3], element(j, [3 1 2]))));
    gradUNC = [(-1, 1, 1; -1, 1, 1; 1, -1) * uNC(I)];
    (...)
end
```

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

\[
\int_{\Omega} p_h \cdot \psi_E dx + \int_{\Gamma_0} \frac{d}{d s} \psi_E u_h ds = \int_{\Gamma_0} u_D \cdot \psi_E ds \quad \text{for all } E \in \mathcal{E}_0 \cup \mathcal{E}_N
\]

For \( E \in \mathcal{E}_D \) one obtains immediately

\[
\int_{\Omega} p_h \cdot \psi_E dx = \frac{2}{E} \int_{T_+} u_h \cdot \psi_E dx \quad \text{for } E \in \mathcal{E}_D \text{ and } E \subset T_+ \in T,
\]

while, for \( E = \partial T_+ \cap \partial T_- \in \mathcal{E}_0 \) with \( T_+, T_- \in T \), there holds

\[
\int_{T_+} \frac{d}{d s} \psi_E u_h ds + \int_{T_-} \frac{d}{d s} \psi_E u_h ds = \int_{\Omega} p_h \cdot \psi_E dx
\]

Given one of 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 $p_h \in L^2(\Omega)$ to the unknown exact flux $p \in H(div; \Omega)$ 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 $S^1_f(T)$ (with homogeneous Dirichlet boundary conditions) based on a regular triangulation $T$ of $\Omega$, i.e.

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

In averaging techniques, the error $||p - p_h||_{L^2(\Omega)}$ is estimated by the approximation error of a smoother approximation $q_h \in S^1(T)^d$ to $p_h$. In fact, the minimal value

$$
\eta_M := \min_{p_h \in S^1(T)} ||p_h - q_h||_{L^2(\Omega)}
$$

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

$$
\eta_M \leq ||p - p_h||_{L^2(\Omega)} + \min_{q_h \in S^1(T)^d} ||p - q_h||_{L^2(\Omega)}.
$$

It is striking that $\eta_M$ is also reliable [CBA, CBK] in the sense of

$$
||p - p_h||_{L^2(\Omega)} \leq C_{rel} \eta_M + \text{h.o.t.}
$$

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 $\eta_M$, 

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

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

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

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

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

$$
Q_h := \{q_h \in S^1(T)^d : \forall z \in N \cup \Gamma, q_h(z) \in A_z\}
$$

with the affine subspace

$$
A_z := \{a \in \mathbb{R}^d : \forall E \in E_z \cap E_N, g(z) = a \cdot n_E \text{ and } \forall E \in E_z \cap E_D, \nabla_E u_D(z) = (a)_E\}
$$

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 $z \in N$ by $p(z) \in A_z$. Define $Ap_h$ by an operator $A : P_h \rightarrow Q_h$ to average $p_h$ on its patch $\omega_z$ [C2] with

$$
Ap_h := \sum_{z \in N} A_z(p_h|_{\omega_z}) \varphi_z \quad \text{and} \quad A_z := \pi_z \circ M_z : P_1(T_z)^d \rightarrow \mathbb{R}^d.
$$

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

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

where $|\omega_z|$ is the measure of $\omega_z$. This is the averaging operator $A$ for the Galerkin approximation of the Poisson problem.
for any node \( z \) with patch \( \omega_z \) 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

\[
\mathcal{A}_z = \pi_z(0) + V_z
\]

is a linear subspace \( V_z \) of \( \mathbb{R}^d \). The (non-linear) orthogonal projection \( \pi_z \) is Lipschitz continuous with \( \text{Lip}(\pi_z) \leq 1 \) 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 \( \tau_{E_1} \) and \( \tau_{E_2} \) on \( E_1, E_2 \), respectively, we consider the \( 2 \times 2 \) systems

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

(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 \( \tau_{E_1} \), let \( p_z \in \mathbb{R}^2 \) solve

\[
\tau_{E_1} \cdot p_z = (\partial u_D|_{E_1}/\partial s)(z) \quad \text{and} \quad \nu_{E_1} \cdot p_z = \int_{\omega_z} \nu_{E_1} \cdot p_z \, dx.
\]

Under these conditions, Theorem 8.1 of [C1, Cba] guarantees reliability for the averaging error estimators \( \eta_M \) and \( \eta_A \) 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 \( \Gamma_N \) is connected and that \( \Gamma_D \) 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 \( (h_T, h_E) \)-independent constants \( C_{\text{eff}}, C_{\text{rel}} \) (that exclusively depend on the shape of the elements and patches) such that

\[
C_{\text{eff}} \eta_A \leq \text{h.o.t.} \leq ||p - p_h||_{L^2(\Omega)} \leq C_{\text{rel}} \eta_A + \text{h.o.t.} \quad \square
\]

The a posteriori error control is performed in the function \texttt{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 \texttt{DirectionEdge} of dimension \( 2 \cdot \text{card}(\mathcal{T}) \times 2 \). The computation of the fluxes is provided in the function \texttt{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 i=1:size(element,1) supp_area(element(:,:))=supp_area(element(:,:))+... ones(3,1)*det([1,1,1;coordinate(element(:,:))])/6; u_h(element(:,:))=u_h(element(:,:))+... det([1,1,1;coordinate(element(:,:))])*det(pEval(3*(i-1)+[1,2,3,:]))'*... 4 1 1; 4 1 1 1 41/36); end u_h=Tangent(coordinate,Dirichlet,u_h./(supp_area*ones(1,2))); eta=zeros(size(element,1),1);```
for $j=1:\text{size(elements,1)}$
  eta_2(j) = \sqrt{\det([1,1,1; \text{coordinate(elements}(j,:),:)]) * ... (sum([1 4 1 1 4 1 1 4]) / 6) * \text{ones}(3,1) / 6);$
end
eta_2 = \sqrt{\text{sum}(eta_2)}/2$)

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

The element contribution $\eta(j) = \|p_h - A_p_h\|_{L^2(T)}$ 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 9.3.3 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

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

This is given from the corresponding error $e$, $N'$ and $e_{N^e}$ 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 $(r, \varphi)$; the exact solution of (2.1) is $u(r, \varphi) := r^{2/3}\sin(2\varphi/3)$. The coarsest triangulation $T_0$ 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 $u_D$ read

```matlab
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 $T_0$ consists of 4 triangles. The user-defined functions for the right-hand sides $f$ and $u_D$ read
TABLE 1. Error, error estimator, and the corresponding experimental convergence rates for uniform mesh-refinement in Subsection 9.1.

<table>
<thead>
<tr>
<th>$N_1$</th>
<th>$N_2$</th>
<th>$e_p$</th>
<th>$\alpha_N$</th>
<th>$e_r$</th>
<th>$\alpha_N$</th>
<th>$\eta$</th>
<th>$\alpha_N$</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>29</td>
<td>0.44372547</td>
<td>0.40360686</td>
<td>0.63793932</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>124</td>
<td>0.28475108</td>
<td>0.3037</td>
<td>0.5399</td>
<td>0.47052834</td>
<td>0.2084</td>
<td></td>
</tr>
<tr>
<td>232</td>
<td>512</td>
<td>0.18454770</td>
<td>0.3051</td>
<td>0.5223</td>
<td>0.29475176</td>
<td>0.3290</td>
<td></td>
</tr>
<tr>
<td>944</td>
<td>2080</td>
<td>0.11881756</td>
<td>0.3137</td>
<td>0.5158</td>
<td>0.18617839</td>
<td>0.3273</td>
<td></td>
</tr>
<tr>
<td>3808</td>
<td>8384</td>
<td>0.07594682</td>
<td>0.3208</td>
<td>0.5116</td>
<td>0.11771707</td>
<td>0.3286</td>
<td></td>
</tr>
<tr>
<td>15296</td>
<td>33664</td>
<td>0.04829497</td>
<td>0.3255</td>
<td>0.5082</td>
<td>0.07438236</td>
<td>0.3301</td>
<td></td>
</tr>
<tr>
<td>61312</td>
<td>134912</td>
<td>0.03060675</td>
<td>0.3285</td>
<td>0.5056</td>
<td>0.04696028</td>
<td>0.3312</td>
<td></td>
</tr>
</tbody>
</table>

9.3. Adaptive mesh refining. Automatic mesh refining generates a sequence $T_0, T_1, T_2, \ldots$ by marking and refining elements and based on red-green-blue refinements and refinement indication by $\eta_T := \|p_h - A p_h\|_{L^2(T)}$. The applied refinement criteria reads: Mark the element $T \in T_k$ for
red-refinement if the error indicator $\eta_T$ satisfies
\[
\frac{1}{2} \max_{T' \in T_h} \eta_A(T') \leq \eta_A(T).
\]

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 $\eta_A$ as a function of $N := N_1$. 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
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


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.