# Scientific Computing 1

Winter term 2017/18
Priv.-Doz. Dr. Christian Rieger
Christopher Kacwin

universitätbonn

## Sheet 11

Submission on **Thursday, 18.1.18**.

**Exercise 1.** (Raviart-Thomas element)

In this exercise, we analyse the behaviour of the Raviart-Thomas element in the framework of primal mixed methods. Let $\Omega \subset \mathbb{R}^d$ be a polygonal domain with a quasi-uniform triangulation $\mathcal{T}_h$. We consider the saddle point problem:
Find $(\sigma, u) \in X \times M = L^2(\Omega)^d \times H_0^1(\Omega)$ such that

$$\begin{aligned} (\sigma,\tau)_0 - (\tau, \nabla u)_0 &= 0 \quad \text{for all } \tau \in L^2(\Omega)^d\,, \\ -(\sigma, \nabla v)_0 + (f,v)_0 &= 0 \quad \text{for all } v \in H_0^1(\Omega)\,. \end{aligned}$$

The Raviart-Thomas finite element spaces $X_h = RT_0$ and $M_h = \{v \in L^2(\Omega) \mid v|_T \text{ constant } \forall T \in \mathcal{T}_h\}$ now describe a nonconformal discretization, therefore we introduce mesh-dependent norms for $\tau \in X_h$, $v \in M_h$

$$\begin{aligned} \|\tau\|_{0,h} &= \left( \|\tau\|_0^2 + h \sum_{e\in\Gamma_h} \|\tau\cdot n_e\|_{0,e}^2 \right)^{1/2}, \\ |v|_{1,h} &= \left( \sum_{T\in\mathcal{T}_h} |v|_{1,T}^2 + h^{-1} \sum_{e\in\Gamma_h} \|J(v)\|_{0,e}^2 \right)^{1/2}, \end{aligned}$$

where $\Gamma_h$ is the set of interior edges, $n_e$ is a normal vector corresponding to an edge $e$, and $J(v)$ is the jump of $v$ at an interior edge $e$.

a) Show that with respect to these norms, the bilinear forms $a(\tau,\sigma) = (\tau,\sigma)_0$ and $b(\tau, v) = (\operatorname{div}\tau, v)_0$ are continuous. Furthermore, show that $a(\cdot,\cdot)$ is coercive.

b) Show that the *inf-sup* condition

$$\sup_{\tau\in X_h} \frac{b(\tau,v)}{\|\tau\|_{0,h}} \geq \beta |v|_{1,h} \quad \text{for all } v \in M_h$$

holds, with $\beta$ independent of $h$.

(8 points)

**Programmieraufgabe 1.** (global system)

The goal of this exercise is to assemble the global system of equations resulting from the finite element discretization on sheet 8. We start with the data obtained from exercise d) on sheet 8.
We recall the basic concepts of the finite element scheme we are implementing. Let $\{\phi_i\}_{i=1}^N$ be the $H^1$-conform, nodal basis (with respect to all nodes, not only the non-boundary nodes). We can state a discrete weak formulation for the Galerkin approximation $u_h$, without boundary conditions:

$$\sum_{T\in\mathcal{T}_h} \left( m|_T \int_T \nabla u_h \nabla\phi_j \,\mathrm{d}x + c|_T \int_T u_h\phi_j \,\mathrm{d}x \right) = \sum_{T\in\mathcal{T}_h} f|_T \int_T \phi_j \,\mathrm{d}x\,, \quad j = 1,\dots,N\,.$$

We expand $u_h = \sum_{i=1}^{N} u_i \phi_i$ and obtain

$$\sum_{i=1}^{N} \sum_{T \in \mathcal{T}_h} \left( m|_T \int_T \nabla\phi_i \nabla\phi_j \,\mathrm{d}x + c|_T \int_T \phi_i \phi_j \,\mathrm{d}x \right) u_i = \sum_{T \in \mathcal{T}_h} f|_T \int_T \phi_j \,\mathrm{d}x\,, \quad j = 1, \ldots, N\,.$$

Here, we see that $S(i,j,T) = \int_T \nabla\phi_i \nabla\phi_j \,\mathrm{d}x$ are the local stiffness matrix entries, $M(i,j,T) = \int_T \phi_i \phi_j \,\mathrm{d}x$ are the local mass matrix entries, and $L(j,T) = \int_T \phi_j \,\mathrm{d}x$ are the local load vector entries. We therefore can write

$$\sum_{i=1}^{N} \sum_{T \in \mathcal{T}_h} \left( m|_T S(i,j,T) + c|_T M(i,j,T) \right) u_i = \sum_{T \in \mathcal{T}_h} f|_T L(j,t)\,, \quad j = 1, \ldots, N\,.$$

To impose Dirichlet boundary conditions, we can set $u_i = 0$ for boundary nodes, as well as set $S(i,j,T) = 0, M(i,j,T) = 0, L(j,T) = 0$ if either $i$ or $j$ corresponds to a boundary node. This is equivalent to removing the rows and columns corresponding to boundary nodes from the equation system, however we keep them set to zero for convenience.

a) Write a routine that assembles the global load vector in the following format: The vector has length the number of nodes, each entry corresponding to its corresponding node in the 'Nodes' vector. If the corresponding Node lies on the boundary, its entry should be zero. Otherwise it is the sum of all local loadvector entries corresponding to this Node, multiplied with the corresponding `fValue`. Achieve this by iterating over the Elements' local loadvectors.

The global system matrix will have many zero entries, so we opt for a sparse matrix format which allows fast multiplication. We first assemble the entries of the global system matrix as a list, then convert this list to the compressed sparse row (CSR) matrix format.

b) Construct an object 'matrixEntry' that can hold a row index (nonnegative integer), column index (nonnegative integer) and a real number (double precision). Write a routine that creates a 'matrixEntry'-array of all nonzero entries for the global system matrix. Start with an array with local entries, i.e. one for each local stiffness matrix entry multiplied with its corresponding `mValue` and one for each local mass matrix entry multiplied with its corresponding `cValue`. Here, only consider entries where both node indices correspond to non-boundary nodes. Then, sort this array into lexicographical order with respect to the row and column indices. Finally, create a new array with global entries by adding up all local entries which have the same row and column index.

c) Inform yourself about the CSR matrix format, e.g. here: `https://en.wikipedia.org/wiki/Sparse_matrix#Compressed_sparse_row_(CSR,_CRS_or_Yale_format)`. Construct an object 'CSRmatrix' and write a routine which converts the list of global entries obtained from exercise b) into a CSRmatrix.

d) Test your implementation. Start with the file `uniformMesh3.txt` from the homepage, and go through all necessary steps to obtain the global load vector and the global system matrix in CSR format.

(16 points)

The programming exercise should be handed in during the exercise classes (bring your own laptop!) in two weeks, i.e. on 25.1/26.1.18. All group members need to attend the presentation of your solution to get points.