# Advanced Discretization Methods (WS 19/20)

# Homework 4

**(P. Knabner, L. Wester)**

| | |
|---|---|
| Deadline for submission (theory): | November 19th, 2019, 12:15 |
| Deadline for submission (programming): | November 19th, 2019, 12:15 |

**Remark:** When you apply theorems, whether they were found in Knabner/Angermann or another source, make sure to **cite them**!

## Exercise 9: Saddle point problems (revisited) (5+5)

a) Consider again the saddle point problem introduced at the end of Exercise 8: Let $a : H^{1,2}(\Omega) \times H^{1,2}(\Omega) \to \mathbb{R}$ and $b : H^{1,2}(\Omega) \times \mathbb{R} \to \mathbb{R}$ be defined by

$$a(u,v) := \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}x \quad \text{and} \quad b(v,\lambda) := \lambda \int_\Omega v \,\mathrm{d}x$$

for $u, v \in H^{1,2}(\Omega)$ and $\lambda \in \mathbb{R}$. We look for $(u,\lambda) \in H^{1,2}(\Omega) \times \mathbb{R}$, s.t.

$$\begin{aligned} a(u,v) + b(v,\lambda) &= \int_\Omega f v \,\mathrm{d}x + \int_{\partial\Omega} g v \; d\sigma \qquad && \forall v \in H^{1,2}(\Omega), \\ b(u,\mu) &= 0 && \forall \mu \in \mathbb{R} \end{aligned}$$

Use Thm. 6.12. in Knabner/Angermann to prove the unique existence of a solution to the above saddle point problem.
**Hint:** Use the coercivity of $\widetilde{a}$ (as proven in Exercise 8) to prove the coercivity of $a$ in $N = \ker B$.

b) Consider again the biharmonic equation (this time with a different boundary condition):

$$\Delta^2 u = f \text{ in } \Omega, \qquad u = \Delta u = 0 \text{ on } \partial\Omega$$

By setting $w := \Delta u$ (i.e. $\Delta w = f$), this problem can be reformulated as a saddle point problem: Find $w, u \in H_0^{1,2}(\Omega)$, s.t.

$$\begin{aligned} a(w,v) + b(u,v) &= 0 && \forall v \in H_0^{1,2}(\Omega), \\ b(w,\psi) &= -\int_\Omega f\psi \,\mathrm{d}x \qquad && \forall \psi \in H_0^{1,2}(\Omega). \end{aligned}$$

where

$$a(w,v) := \int_\Omega wv \,\mathrm{d}x \qquad \text{and } b(w,u) := \int_\Omega \nabla w \nabla u \,\mathrm{d}x$$

for $u, v, w \in H_0^{1,2}(\Omega)$ (you do not need to prove that these systems are equivalent).
Apply again Thm. 6.12. to prove the unique existence of a solution to the above saddle point problem.

## Exercise 10: The nature of saddle point problems (10)

When faced with a variational saddle point problem of the form: Find $(u,p) \in V \times W$, s.t.

$$\begin{aligned} a(u,v) + b(v,p) &= f(v) \qquad && \forall v \in V, \\ b(u,q) &= g(q) \qquad && \forall q \in W, \end{aligned}$$

discretization leads to a matrix system of the following form:

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}$$

where $A_{ij} = a(\varphi_j, \varphi_i), B_{ij} = b(\varphi_i, \psi_j), f_i = f(\varphi_i), g_i = g(\psi_i)$ for given shape functions $\varphi_\cdot, \psi_\cdot$. Assume that this system consists of a symmetric positive definite matrix $A \in \mathbb{R}^{n\times n}$ and a non-square matrix $B \in \mathbb{R}^{m\times n}$ with full rank $m (< n)$. Show that this linear system of equations is equivalent to a saddle point problem

$$\max_{q\in\mathbb{R}^m} \min_{v\in\mathbb{R}^n} J(v,q)$$

for a certain bilinear form $J : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$.

## Programming exercise 3: Quadratic Elements (Part 1) (5+2 Bonus Points)

Revisit `gen_mesh_rectangle.m` and upgrade it with the capability to generate meshes for quadratic Lagrange elements. It should now take the additional argument `el_type`:

```
[coord,elemNodeTable,boundary] = gen_mesh_rectangle(...,el_type)
```

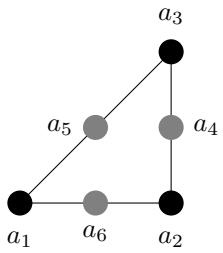
where

$$\texttt{el\_type} = \begin{cases} 1 & \text{Linear Lagrange Elements} \\ 2 & \text{Quadratic Lagrange Elements} \end{cases}$$

In the case of quadratic Lagrange elements, our data structures change as follows:

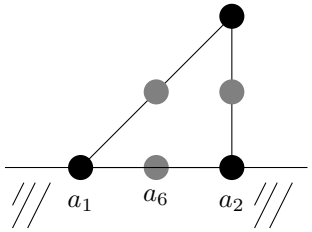
- `coord` now also saves all the edge midpoints on every triangle. For future applications it is highly recommended to use a global ordering that lists all the corner nodes first and **then** all the edge midpoints. Doing so will award the extra 2 bonus points mentioned in the title.

- `elemNodeTable` is now a $n_T \times 6$ matrix, locally ordering the node indices of each triangle like this:




  i.e. $a_4$ should be opposite of $a_1$, $a_5$ opposite of $a_2$, etc. In particular one should be able to extract the corresponding linear Lagrange mesh from the first 3 columns of `elemNodeTable`.

- `boundary` is now a $n_B \times 4$ matrix, where the last 3 entries define the boundary edge, i.e.:




  If the above element lies on the boundary as sketched, `boundary` would have an entry like so:

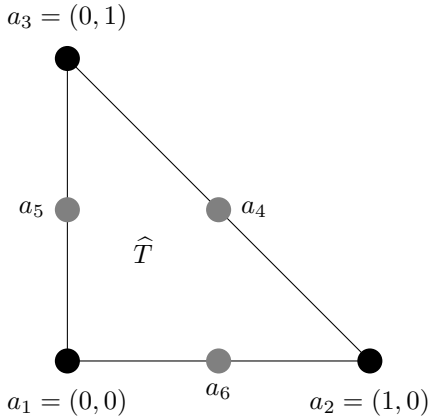
$$\texttt{boundary} = \begin{bmatrix} & \vdots & & \\ \text{bd\_type} & a_1 & a_6 & a_2 \\ & \vdots & & \end{bmatrix}$$

  One should be able to extract the linear Lagrange `boundary` information by dropping the "middle" column.

## Programming exercise 4: Quadratic Elements (Part 2) (15)

We now want to finish our preparations ahead of solving a Stokes system. For this purpose, revisit `AssembleMatrices.m` and `AssembleRHS.m` and upgrade them to be able to handle quadratic (as well as linear!) Lagrange elements. Both methods should be able to determine the element type from the size of `elemNodeTable` (compare Programming exercise 3 for data structure explanations), so no additional input is required.

The shape functions on the reference element $\widehat{T}$ are given as follows:




- $N_1(x,y) = (1-x-y)(1-2x-2x)$
- $N_2(x,y) = x(2x-1)$
- $N_3(x,y) = y(2y-1)$
- $N_4(x,y) = 4xy$
- $N_5(x,y) = 4y(1-x-y)$
- $N_6(x,y) = 4x(1-x-y)$

Like before, use the trapezoidal rule (on the edge midpoints) to evaluate the integrals on each triangle.
You can generate meshes using your code from Programming exercise 3, or help yourself to the 3 sample meshes deposited on StudOn (files in MatLab can be loaded using `load([filename])`).
Test your program with the example from Programming exercise 1 (i.e. $u(x,y) = \sin(2\pi x)\exp(y)$ on $\Omega = [0,1] \times [0,2]$ with $a(x,y) = x + \sin^2(y),\ r(x,y) = 2x + 2y$.