Numerical Algorithms

Winter semester 2018/2019
Prof. Dr. Marc Alexander Schweitzer
Denis Duesseldorf

universitätbonn

Exercise sheet 0. revision and warm-up sheet, no submission, no grading

- The website of this lecture can be found at
`https://ins.uni-bonn.de/teachings/ws-2018-230-v4e1-numerical-algori/`
- For admission to the exams at the end of the lecture *50%* of achievable points from normal exercises and *50%* of achievable points from programming exercises are required.
- Please, feel free to submit your homework assignments in groups of up to 3 students.
- Tutorials for this lecture take place in the seminar room 1.008 of Endenicher Allee 60, Wednesdays, 08-10 c.t.
- This first worksheet is intended for revision and preparation, hence, there will be neither submission nor grading. In particular, it should be used to refresh and (re)introduce `Python/NumPy/matplotlib`.

Exercise 1. (First Strang lemma and and quadrature)

For the solution of the variational problem $u \in \mathcal{V} \subset H^m(\Omega)$

$$a(u, v) = (l, v) \qquad , \forall v \in \mathcal{V}$$

consider a numerical solution $u_h \in \mathcal{S}_h \subset \mathcal{V}$ from a finite dimensional space $\dim \mathcal{S}_h < \infty$

$$a_h(u_h, v) = (l_h, v) \ , \qquad \forall v \in \mathcal{S}_h.$$

Assuming that a is elliptic and a_h are uniformly elliptic we get the error estimate

$$\|u - u_h\| \leq c \left(\inf_{v_h \in \mathcal{S}_h} \left(\|u - v_h\| + \sup_{w_h \in \mathcal{S}_h} \frac{|a(v_h, w_h) - a_h(v_h, w_h)|}{\|w_h\|} \right) + \sup_{w \in \mathcal{S}_h} \frac{|(l, w_h) - (l_w, w_h)|}{\|w_h\|} \right)$$

with some constant $c > 0$, i.e. the global error is bounded by the best approximation error $\inf_{v \in \mathcal{S}_h} \|u - v_h\|$, the consistency error of the bilinear form a and the consistency error of the functional l. Assume an underlying mesh of triangles T approximating Ω.

a) Assume that approximation of a were exact (or optimal), i.e. for all intents and purposes $a = a_h$. What follows for the relation between best approximation error $\|u - u_h\|$ and the consistency error for l and l_h when l_h is given by a discrete quadrature approximation of l

$$(l, v) := \int_\Omega f(x)v(x)\mathrm{d}x \ , \qquad (l_h, v) := Q_h(fv) = \sum_{i \in N_h} w_i f(x_i) v(x_i) \ ?$$

b) Assume that

$$\sup_{v_h \in \mathcal{S}_h} \|u - v_h\| \in O(h^k) \; .$$

for some $k \in \mathbb{N}$. Assume that $\forall u \in \mathcal{V}$ it holds that $u \in C^\infty(T)$ for all triangles T of the underlying mesh. Which quadrature rule Q_h on single triangles T is sufficient for a globally optimal approximation u_h? Which rule is sufficient for $k = 1$?

c) What are smoothness requirements on f for Q_h to have the required consistency error?

Exercise 2. (A pure Neumann problem without compatibility condition)

Consider the variational form of the pure Neumann problem

$$\begin{aligned} -\nabla \cdot (\kappa \nabla u) &= f \; , & x &\in \Omega \\ \kappa \frac{\partial u}{\partial \nu} &= h \; , & x &\in \partial\Omega \end{aligned}$$

and suppose that

$$\int_\Omega f + \int_{\partial\Omega} h \neq 0 \; .$$

a) Show that

$$u \in H^1(\Omega) \;: \qquad \int_\Omega \kappa \nabla u \cdot \nabla v = \int_\Omega f v + \int_{\partial\Omega} h v \qquad \forall v \in H^1$$

has no solution.

b) Show that with

$$V := \{ v \in H^1(\Omega) : \int_\Omega v = 0 \}$$

the solution of the variational problem

$$u \in V \;: \qquad \int_\Omega \kappa \nabla u \cdot \nabla v = \int_\Omega f v + \int_{\partial\Omega} h v \qquad \forall v \in V$$

solves the boundary value problem

$$\begin{aligned} -\nabla \cdot (\kappa \nabla u) &= \tilde{f} & x &\in \Omega \\ \kappa \frac{\partial u}{\partial \nu} &= h & x &\in \partial\Omega \end{aligned}$$

where

$$\tilde{f} = f - \frac{\gamma}{|\Omega|} \; , \qquad \gamma = \int_\Omega f + \int_{\partial\Omega} h \; .$$

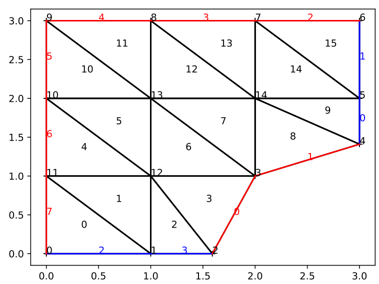
Programming exercise 1. (Simple mesh data structures)

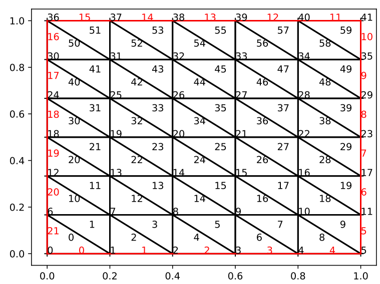
We will accompany the lecture with programming exercises to put the learned methods into practice. The first major program will be the step-wise construction of a simple finite element code for the Poisson problem

$$
\begin{aligned}
-\nabla \cdot (A\nabla u) &= f & x &\in \Omega \\
u &= g & x &\in \Gamma_D \subset \partial\Omega \\
\frac{\partial}{\partial \nu} u &= h & x &\in \Gamma_N = \partial\Omega \setminus \Gamma_D
\end{aligned}
$$

on a bounded Lipschitz domain Ω with a polygonal boundary $\partial\Omega$, Dirichlet boundary conditions on Γ_D and (natural) Neumann boundary conditions on Γ_N with an underlying mesh for the resolution of Ω.

The most important aspect in the beginning is the choice of data structures for the mesh/triangulation. In this we will just store everything in lists and arrays in a straight forward fashion, i.e. for the left mesh depicted here





we have

- The vertices are just stored as arrays of the coordinates.

```
# x-coordinate, y-coordinate
coordinates = array([[0,0],[0,1],[1.59,0],[2,1],[3,1.41],[3,2],[3,3],
                     [2,3],[1,3],[0,3],[0,2],[0,1],[1,1],[1,2],[2,2]])
```

- Triangles as just stored as lists of the indices of the vertices in counterclockwise ordering.

```
# index of node 0, node 1, node 2
triangles = array([[0,1,11],[1,12,11],[1,2,12],[2,3,12],
                   [11,12,10],[12,13,10],[12,3,13],[3,14,13],
                   [3,4,14],[4,5,14],[10,13,9],[13,8,9],
                   [13,14,8],[14,7,8],[14,5,7],[5,6,7]])
```

- Dirichlet and Neumann boundary conditions are given as list of edges to which they are applied to.

```
# edge node left, edge node right
neumann = array([[4,5],[5,6],[0,1],[1,2]])
dirichlet = array([[2,3],[3,4],[6,7],[7,8],[8,9],[9,10],[10,11],[11,0]])
```

a) Write a function to draw the vertices of the mesh.

b) Write a function to draw the edges of mesh.

c) Write functions to draw the boundary edges of the mesh, Dirichlet boundary Γ_D in red, Neumann boundary in blue.

d) Write a function `grid_mesh(nn,mm)` that creates the `coordinates`, `triangle`, `neumann` and `dirichlet` arrays for the domain $\Omega = [0,1]^2, \Omega_D = \partial\Omega$ with regular Courant triangles aligned in a grid with nodes (x_i, y_i), $x_i = i\frac{1}{\texttt{nn+1}}, y_j = j\frac{1}{\texttt{mm+1}}$, $i = 0, \ldots, \texttt{nn} + 1$, $j = 0, \ldots, \texttt{mm} + 1$. Plot the resulting triangulation for $\texttt{nn}= 4$,$\texttt{mm}= 5$ producing a 5×6 grid of squares of two triangles each.

Send to `duesseldorf@ins.uni-bonn.de`