TU Berlin — Department of Mathematics
Dr. Kersten Schmidt, Dirk Klindworth

winter term 2015/2016

NUMERICS OF PARTIAL DIFFERENTIAL EQUATIONS

Series 5

1. Let $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, be a Lipschitz domain, $c \in L^\infty(\Omega)$, $f \in L^2(\Omega)$, and $g \in H^{1/2}(\partial\Omega)$. We consider the boundary value problem

$$-\Delta u + cu = f \qquad \text{in } \Omega, \tag{1a}$$
$$u = g \qquad \text{on } \partial\Omega. \tag{1b}$$

a) Let $c, g \equiv 0$. Derive the variational formulation of (1), choose appropriate test and trial spaces and prove that it is well-posed.

b) Let $g \equiv 0$. Find $c_0 \leq 0$ such that (1) is well-posed for any c with $c \geq c_0$ almost everywhere.

c) Now let $g \not\equiv 0$. Derive the variational formulation of (1) with identical test and trial spaces.
Hint: Use the Dirichlet lift ansatz, i. e. set $u = u_0 + u_g$ with the new unknown u_0, that satisfies $u_0|_{\partial\Omega} = 0$, and a known function u_g such that $u_g|_{\partial\Omega} = g$.

2. Let $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, be a Lipschitz domain, $0 < \sigma_0 \leq \sigma \leq \sigma_1 < \infty$ almost everywhere, $f \in L^2(\Omega)$, and $g \in H^{-1/2}(\partial\Omega)$. We consider the boundary value problem

$$-\operatorname{div} \sigma \,\mathbf{grad}\, u = f \qquad \text{in } \Omega, \tag{2a}$$
$$\sigma \,\mathbf{grad}\, u \cdot \mathbf{n} = g \qquad \text{on } \partial\Omega, \tag{2b}$$

where f and g satisfy the compatibility condition

$$\int_\Omega f \,\mathrm{d}\mathbf{x} + \int_{\partial\Omega} g \,\mathrm{d}S(\mathbf{x}) = 0. \tag{3}$$

a) Derive the variational formulation

$$\mathsf{a}(u, v) = \ell(v) \tag{4}$$

of (2), choose appropriate test and trial spaces and prove that it is well-posed.

b) We consider the mixed variational formulation: find $u \in H^1(\Omega)$ and $\lambda \in \mathbb{R}$ such that

$$\mathsf{a}(u, v) + \lambda \int_\Omega v \,\mathrm{d}\mathbf{x} = \ell(v) \qquad \forall v \in H^1(\Omega), \tag{5a}$$
$$\int_\Omega u \,\mathrm{d}\mathbf{x} = 0. \tag{5b}$$

Choose a test function v in (5a) to determine the (exact) value of the Lagrange multiplier λ.
Hint: Use the compatibility condition (3).

c) Assume for now that (5) is well-posed. Explain why u is the unique solution of (5) if and only if it is the unique solution of the variational formulation (4) of the boundary value problem (2).

See next page!

3. Programming exercise: mesh generation (due to November 24th, 2015, 10.15. a.m.)*

a) Install the mesh generator Gmsh (`http://geuz.org/gmsh/`).

b) Study the documentation of Gmsh available online and familiarize yourself either with the graphical user interface or with the scripting of `.geo`-files to produce 2d triangular meshes. In particular, study how to define the maximal mesh width `h0`.

c) Write a Python function `read_gmsh`, that accepts the name of a `.msh`-file as input parameter and that returns the following variables of a 2d mesh:

- `p`: a Numpy-array of size $N \times 2$ with the x_1- and x_2-coordinates of the N vertices of the mesh,
- `t`: a Numpy-array of size $M \times 3$, where each row contains the indices of the three vertices of one of the M triangles in counterclockwise order.
- `be`: a Numpy-array of size $B \times 2$, where each row contains the indices of the two vertices of one of the B boundary edges.

Save this function in a Python module `meshes.py`.

d) Add a member `grid_square` to the module `meshes.py` that produces a uniform grid of a square as presented in Figure 1. This new function `grid_square` should accept the side length `a` of the square and a maximal mesh width `h0`. The lower left corner of the square should be positioned at the origin. The return values of `grid_square` are the same like those of `read_gmsh`.

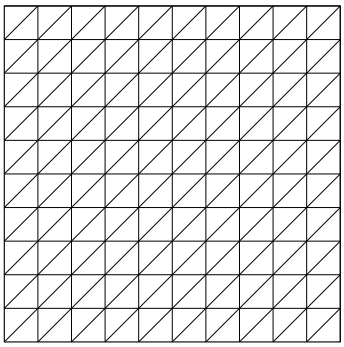

Figure 1: Uniform grid of triangles as a mesh for a square.

e) Add a member `show` to the module `meshes.py` that draws a mesh using Matplotlib. This member should accept a Numpy array `p` of the x_1- and x_2-coordinates of the vertices of the mesh and a Numpy array `t` representing the triangles of the mesh as indices into `p`.

f) Create an unstructured mesh of a square with side length $a = 1$ and maximal mesh width $h_0 = 0.1$ using Gmsh and a uniform grid using `grid_square` with the same parameters and plot these two meshes with the help of your function `show`.

g) Add a member `max_mesh_width` to the module `meshes.py` that returns the actual maximal mesh width of an unstructured mesh with vertices `p` and triangles `t`.

h) Plot the number of vertices in dependence of the actual maximal mesh width for both, the unstructured mesh produced by Gmsh and the uniform grid produced by `grid_square`. To this end, choose $a = 1$ and at least six different maximal mesh widths as input parameter of the two functions. Be prepared to give an interpretation of your results in the tutorial class.

See next page!

To be handed in by: November 17th, 2015, 10.15 a.m. (before lecture starts)
Please hand in your solutions of at least two parts of exercise 1 and two parts of exercise 2.

* Send your Python module `meshes.py`, the `.msh`-files of all generated Gmsh meshes, and a Python script, that produces the plots in 3.f) and 3.h) when executed from command line without any need of interaction, to `klindworth@math.tu-berlin.de`! **The code must be well documented using prologue and inline comments! The plots must have titles, labels and legends!**

This exercise series will be discussed in the tutorial class on November 19th, 2015, 2.15 p.m. in A 052.

Coordinator: Dirk Klindworth, MA 365, 030/314-25192, `klindworth@math.tu-berlin.de`
Website: `http://www.tu-berlin.de/?NumPDE`