Technische Universität Chemnitz
Dr. M. Winkler

Chemnitz, 28th June 2019
hand in on 10th July 2019

Numerical Methods for Partial Differential Equations

Sheet 10

Exercise 25: Implementation of adaptive finite elements

This exercise is based on the implementation from Sheet 9, Exercise 23. It is the aim to implement an adaptive method using a residual-based error estimator. The steps

$$\texttt{SOLVE} \quad \rightarrow \quad \texttt{ESTIMATE} \quad \rightarrow \quad \texttt{MARK} \quad \rightarrow \quad \texttt{REFINE}$$

should be realized one after the other.

`ESTIMATE` : Here, you should implement the error estimator from Theorem 16.4. Suitable quadrature formulas can be taken from `quadrature_unit_triangle_area.m`. Note that these formulas are valid on the unit triangle only. To this end, one has to work with affine transformations again. Some steps are similar to the assembly routines from `area_integrator.m`.

`MARK` : Implement the Dörfler-Strategy from Example 16.7. The function header should have the form

```
function ref_elem = doerfler_marking(eta2,sigma)
```

The vector `eta2` ccontains the squares of the locally estimated errors η_K. The return value `ref_elem` is a vector containing the indices of that cells that should be refined.

`REFINE` : Read the help text of `myrefinemesh.m`. The function implemented therein has to be called with a third parameter to realize a local refinement. This is exactly the return value `ref_elem` from `doerfler_marking`.

Test your implementation for different domains with different input data and observe where the mesh will be refined. This should happen e. g. in the following cases:

- Near reentrant corners of an L-shaped domain (use `geometry = 'lshapeg';` to create such a geometry),
- In case of $\mathbb{P}_1$-elements near points where the right-hand side f has large function values.
- Near jumping coefficients.

Hint: Use the template files `estimate_error.m`, `doerfler_marking.m` and `SolveAdaptive.m`.