**Proseminar Physically-Based Simulation**
**Winter Semester 2017**

***Programming Assignment Sheet 2***
**Hand-out: November 9th, 2017**
**Hand-in: November 30th, 2017**

# Finite Element Method

**Outline**

In this exercise, a Finite Element Method (FEM) solver will be completed to solve a Poisson´s problem in 2D. The target is to find a solution to the following 2nd order elliptic partial differential equation:

$$-\Delta u(x,y) = f(x,y)$$

with the problem domain given as the unit square

$$\Omega = [0,1]\times[0,1]$$

with source term

$$f(x,y) = -6 - 12xy$$

and Dirichlet boundary condition on the domain boundary

$$u(x,y) = 3x^2 + 2xy^3 \qquad (x,y) \in \delta\Omega .$$

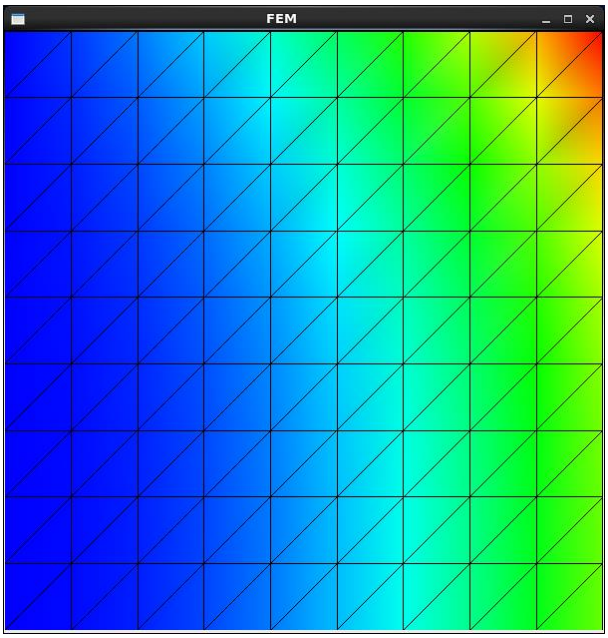
Note that here the selected boundary condition is also the exact, analytical solution.

The problem domain is regularly subdivided into linear triangular elements. The resolution of the mesh (i.e. number of pairs of triangles per each axis) can be specified as a command line parameter (default 20). The key tasks are to set up the stiffness matrix, the right-hand side vector, and (after solving) to compare the numerical to the analytical solution.

The main goal of the assignment is to get acquainted with a simple FEM implementation, to apply the theoretical concepts taught in the lecture, and to examine the behavior of this numerical method.

**Simulation Framework**

A C++ project framework is provided that sets up the FE model, calls the solver, and then displays the solution (or the error). Legacy OpenGL is employed for the visualization (values mapped to color range BLUE-RED). Utility classes for handling 2D vectors, 3D vectors, and 3×3 matrices are also provided, implementing basic operations. Moreover, utility functions for handling sparse symmetric matrices (i.e. the stiffness matrix) and HSV-to-RGB conversion (for the visualization) are also available. Finally, a preconditioned conjugate gradient solver is provided (thus an iterative solver does not have to be implemented). After adding the required changes and afterwards running the code, the program should provide a visualization of the result as follows (here resolution set to 10):



**Tasks**

The exercise consists of setting up the linear system of equations of the FE solver and to evaluate the accuracy of the numerical solution. The code has to be added to the files `LinTriElement.cpp` and `FEModel.cpp`. Further changes to the code should not be necessary.

1. Implement code for computing the partial derivatives of the linear basis functions on the triangular elements. Linear basis functions are given as:

$$N_i(x,y) = c_{i,0} + c_{i,1}x + c_{i,2}y \qquad i = 1,2,3$$

2. Assemble the global stiffness matrix. For this the contributions of the element stiffness matrices have to be determined and added to the global matrix.

$$K_{ij} = \sum_{S_{ij}} \left( \frac{\partial N_i}{\partial x} \cdot \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \cdot \frac{\partial N_j}{\partial y} \right) A_e$$

3. Determine the right-hand side vector of the FE model. This amounts to evaluating an element´s contribution, employing one-point quadrature at the barycenter for approximating the integral.

$$\int_{\Omega_e} f \cdot N_j d\Omega_e \approx A_e \cdot f(x_c, y_c) \cdot N_j(x_c, y_c)$$

4. After setting up the system of linear equations the program calls a preconditioned conjugate gradient solver for finding an approximate solution for the vector of unknown nodal values $\mathbf{u}$.

$$\mathbf{K}\mathbf{u} = \mathbf{f}$$

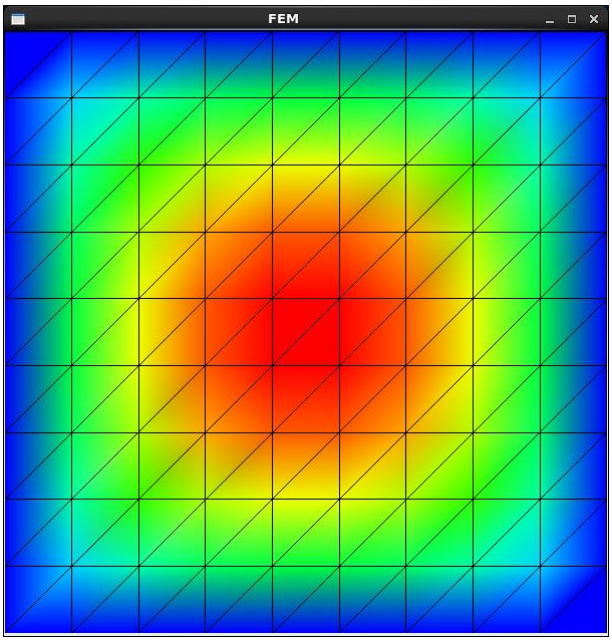
Evaluate the accuracy of the solution by determining the absolute error between the numerical and the analytical solution.

$$\mathbf{u}_{error} = \left| \mathbf{u}_{numerical} - \mathbf{u}_{analytical} \right|$$

Based on this, obtain a final scalar error by evaluating an inner product error norm.

$$error = \sqrt{\mathbf{u}_{error}^T \mathbf{K}\mathbf{u}_{error}}$$

Examine the behavior of error norm and convergence for different mesh resolutions. An example visualization of the error is as follows:




Submission of your solution is due on November 30th, 2017 at 11:59pm. Please submit all source code (i.e. no executables) and report documents in a ZIP file via OLAT (submission access will be made available soon).

The solution should be carried out in teams of 2-3 students. Please get in touch if you are unable to find a team. Make sure to acknowledge all team members in the code and submission (please name the ZIP file with all surnames). Only one submission per team is required.

Finally, please respect the academic honor code. Each team should work on its own on the solution of the assignment. Any attempts at plagiarism or collusion will lead to 0 marks and further scrutiny. Please get in touch in case of questions or problems with regard to the assignment.

In total there are 20 marks achievable in this assignment.

**Implementation Remarks**

The solution should be developed using the provided framework. While you are free to develop in any environment, the final submission should compile and run in Linux. As reference, the machines in RR21 will be considered. Make sure that after unpacking your solution can be compiled directly in the top-level directory via `make`. Follow a proper coding style and provide appropriate comments in the code.