INTRODUCTION TO THE FINITE ELEMENT METHOD

Series 5

1. Boundary value problem with pure homogeneous Neumann boundary conditions

We are interested in the solution of

\[-\Delta u + u = f \quad \text{in } \Omega,\]
\[\nabla u \cdot \mathbf{n} = 0 \quad \text{on } \partial \Omega.\]  

a) What is the variational formulation of (1)?

b) Give appropriate function spaces for the test and trial functions!

c) Give an appropriate linear FE function space for the discrete test and trial functions!

d) How do we account for the boundary conditions (1b) in our FEM code?

2. Element stiffness matrix, element matrix and element load vector for linear finite elements

a) Write a function `elemStiffnessP1.m` that returns the local stiffness matrix for a single triangle. The input parameters are the coordinates of the three nodes. Use the analytic formulas derived in the lecture.

b) Write a function `elemMassP1.m` that returns the local mass matrix for a single triangle. The input parameters are the coordinates of the three nodes. Use the analytic formulas derived in the lecture.

c) Write a function `elemLoadP1.m` that returns an approximation to the element load vector, that is the vector related to the linear form (2) for the three linear shape functions, using Gauss quadrature. The number of quadrature points should be given as input parameter.

```matlab
function phi = elemLoadP1(f, p, n)
if nargin < 3
    n = 1;  # in case of only two input arguments
    # take one quadrature point
end
```

See next page!
Check your code (at least) with \( f = 1 \). Note that \( \sum_{j=1}^{3} \phi_j \) for \( f = 1 \) is equal to the area of the triangle, where \( \phi_j, j = 1, 2, 3 \), are the three components of the element load vector. The function \( f \) can be defined as handle, e.g., \( f = @(x,y) (x^2 - y^2) \) or as a m-file

\[
\text{function } z = f(x,y) \\
z = x^2 - y^2;
\]

Formulas for numerical quadratures on triangles are defined on the triangle \( \tilde{K} \) with the nodes \((-1,-1), (1,-1), (-1,1)\), see for instance Gauß quadrature rules on page 141 in chapter 4 of the book of P. Šolín. They read

\[
\int_{\tilde{K}} f(\xi) d\xi \approx \sum_{j=1}^{n} w_j f(\xi_j).
\]

On the other hand, the element shape functions are defined on the reference triangle \( \hat{K} \) with nodes \((0,0), (1,0), (1,1)\).

Write the linear form

\[
\ell_K(v) = \int_K f(x)v(x)dx. \quad (2)
\]

as integral over \( \hat{K} \) (in dependence of the coordinates \( p_0, p_1, \) and \( p_2 \) of the nodes of \( K \)) and using element shape functions \( \hat{N} \), which simply transform to \( K \).

For the Gauß quadrature rules on triangles use the function \texttt{gaussTriangle.m} that provides the quadrature points and the corresponding weights for all orders up to five. You find the function \texttt{gaussTriangle.m} on the course webpage.

3. Assembling of stiffness matrix, mass matrix and load vector

a) The number of non-zero entries in the \( N \times N \) stiffness matrix is much smaller than \( N^2 \). Matlab provides a sparse matrix format, see \texttt{help sparse}.

Write a function returning the stiffness matrix in sparse format.

\[
\texttt{# Returns the stiffness matrix for} \\
\texttt{# int}_\texttt{Omega} \texttt{ grad u . grad v dx} \\
\texttt{# for linear FEM on triangles} \\
\texttt{#} \\
\texttt{function } A = \texttt{stiffnessP1(p, t)} \\
\texttt{# input:} \\
\texttt{# p } \quad \texttt{N} x \texttt{2 matrix with coordinates of the nodes} \\
\texttt{# t } \quad \texttt{M} x \texttt{3 matrix with indices of nodes of the} \\
\texttt{# triangles} \\
\texttt{#} \\
\texttt{# output:} \\
\texttt{# A } \quad \texttt{N} x \texttt{N} \texttt{ sparse matrix}
\]

Use the function \texttt{elemStiffnessP1.m} and the relation from local numbering of the shape functions to the global numbering of the basis function given the matrix \( t \).

b) Write a function returning the mass matrix in sparse format.

See next page!
# Returns the mass matrix for
# \int_\Omega u \, v \, dx
# for linear FEM on triangles
#
function M = massP1(p, t)
# input:
# p \quad Nx2 matrix with coordinates of the nodes
# t \quad Mx3 matrix with indices of nodes of the
# triangles
#
# output:
# M \quad NxN sparse matrix

Use the function `elemMassP1.m` and the relation from local numbering of the shape functions to the global numbering of the basis function given the matrix `t`.

c) Write a function returning the load vector.

# Returns the load vector for
# \int_\Omega f \, v \, dx
# for linear FEM on triangles
#
function b = loadP1(p, t, f)
# input:
# p \quad Nx2 matrix with coordinates of the nodes
# t \quad Mx3 matrix with indices of nodes of the
# f
#
# output:
# b \quad Nx1 matrix

Use the function `elemLoadP1.m` and the relation from local numbering of the shape functions to the global numbering of the basis function given the matrix `t`.

d) Write the main code for solving (1) including

- selection of a geometry (e.g. square),
- mesh generation for the chosen geometry,
- computation of the stiffness matrix, mass matrix and load vector,
- solving the system using Matlab’s backslash operator, and
- graphical representation with `trimesh` or `trisurf`.

e) Choose $f$ and $\Omega$ such that an analytical solution is known and check your code.

Send your code to klindworth@math.tu-berlin.de by Tuesday, June 3rd, 2014, 10.00am!
There are consultations in the tutorial classes on May 22nd, 2014, 10.15am, MA 542, and May 27th, 2014, 10.15am, MA 545. Do not forget to bring your laptop!

Coordinator: Dirk Klindworth, MA 665, 030/314-25192, klindworth@math.tu-berlin.de
Website: http://www.math.tu-berlin.de/?fem-lecture