7. Assignment „Numerische Mathematik für Ingenieure II“

http://www.moses.tu-berlin.de/Mathematik/

Construction of finite elements – PART I

---

1. **Exercise:** Finite elements in 1D **10 points**

Consider the classical 1D elliptic problem with Dirichlet boundary conditions

$$\begin{aligned} -u''(x) + cu(x) &= f(x), && \text{in } \Omega = (0,1) \\ u(0) = u(1) &= 0 \end{aligned}$$

where $0 < c \in \mathbb{R}$.

(a) Let V a vectorspace of functions over $\mathbb{R}$ which vanish at the boundary.[1] Derive the variational form, i.e. determine the bilinear form $a : V \times V \to \mathbb{R}$ and the linear form $f : V \to \mathbb{R}$, so that for a weak solution u we have $a(u,v) = f(v)$ for all $v \in V$. Show that a is symmetric.

(b) For given collection of points $0 = x_0 < x_1 < \ldots < x_N = 1$ and $N \in \mathbb{N}$ consider the elements $\Omega_i = (x_{i-1}, x_i)$ for $i = 1, .., N$ and the discrete space

$$V_h = \left\{ v \in V : v\big|_{\Omega_i} \text{is affine linear} \right\},$$

where the basis for $j = 1, .., N-1 = \dim V_h$ is

$$w_j(x) = \begin{cases} \frac{x - x_{j-1}}{x_j - x_{j-1}} & x \in \Omega_j \\ \frac{x_{j+1} - x}{x_{j+1} - x_j} & x \in \Omega_{j+1} \\ 0 & \text{else} \end{cases}$$

The Galerkin approximation is $u_h(x) = \sum_{j=1}^{\dim V_h} \alpha^j w_j(x) \in V_h$. Show that

$$\int_{\bar\Omega} u_h'(x) w_i'(x) \mathrm{d}x = \sum_{k=1}^{N} \int_{\bar\Omega_k} {u_h}'(x) w_i'(x) \mathrm{d}x = \sum_j \left[\sum_{k=1}^{N} \int_{\bar\Omega_k} w_i'(x) w_j'(x) \mathrm{d}x \right] \alpha^j .$$

$$\int_{\bar\Omega} u_h(x) w_i(x) \mathrm{d}x = \sum_{k=1}^{N} \int_{\bar\Omega_k} u_h(x) w_i(x) \mathrm{d}x = \sum_j \left[\sum_{k=1}^{N} \int_{\bar\Omega_k} w_i(x) w_j(x) \mathrm{d}x \right] \alpha^j .$$

(c) The elementary task is to compute

$$\bar S = \int_{\bar\Omega_k} w_i'(x) w_j'(x)\, \mathrm{d}x \qquad \text{and} \qquad \bar M = \int_{\bar\Omega_k} w_i(x) w_j(x)\, \mathrm{d}x.$$

The claim is that this was done in assignment 6, exercise 4. Please explain! What is the map $F_k : (0,1) \to \Omega_k$ in terms of points x_i and/or $h_k = x_k - x_{k-1}$ and specify the nonzero terms explicitly.

Hint: You need to interprete $\bar M$ and $\bar S$ as 2×2 matrices of nonzero parts.

(d) How can you use $\bar S$ and $\bar M$ to compute the Galerkin matrix A_h?

Lesson: Teaching the basic idea of finite elements with a simple function space.

2. **Programming exercise:** Discretization of variational form in 1D **13 points**

(a) In this exercise you write a MATLAB program for the variational form of exercise 1. The main task is to build the matrix $(A_h)_{ij} = a(w_j, w_i)$ using the symmetric, bilinear form a. We divide the construction in different steps, which are generalizable to higher dimension. From exercise 1 you know a and f, the basis w_j, and the elements Ω_i.

---

[1] **Remark:** This space is constructed in such a way, that differentiation is well-defined and functions in V satisfy the boundary conditions. Note that differentiability is imposed in a weaker sense than in $C_0^1(\bar\Omega)$. However, you might think of V being C_0^1 most of the time. We have $w_i \in C(\bar\Omega) \subset V$.

step 1) **element generation:**

Write a `function [nelement,npoint,e2p]=generateelements(x)` which for given set of points `x` where `x(1)<x(2)<...<x(end)` returns the number of elements `nelelement` and the number of points `npoint`. The variable `e2p` (element-to-point map) is a `nelement`×2 matrix for which `e2p(k,1)` is the index of the left node and `e2p(k,2)` is the index of the right node of the element Ω_k.

Hint: With index we mean $1 \le i \le$ `npoint`, such that the actual point is $x(i)$.

step 2) **computation of transformation:** Consider the affine linear function F_k, which maps the reference element $\Omega_{\text{ref}} = (0,1)$ to an element Ω_k. Write a `function [edet,dFinv]=generatetransformation(k,e2p,x)` which for given element number `k` returns `edet`$=\det(\nabla F_k)$ and `dFinv`$=(\nabla F_k)^{-1}$.

Hint: Since F_k is affine linear, these two are just constant expressions.

step 3) **computation of local matrices:** In order to compute the Galerkin matrix A_h we need the local matrices $\bar{S}, \bar{M}$ from (1c). Write a `function mloc=localmass(edet)` and a function a `function sloc=localstiff(edet,dFinv)` which for given value of `edet` and `dFinv`$= G$ computes the element mass and stiffness matrices

$$\bar{M} = \int_{\Omega_k} w_i(x) w_j(x)\,\mathrm{d}x = \int_{\Omega_{\text{ref}}} \phi_{\bar{i}}(x)\phi_{\bar{j}}(x)|\texttt{edet}|\,\mathrm{d}x$$

$$\bar{S} = \int_{\Omega_k} w_i'(x) w_j'(x)\,\mathrm{d}x = \int_{\Omega_{\text{ref}}} \phi_{\bar{i}}'(x) G G^T \phi_{\bar{j}}'(x)|\texttt{edet}|\,\mathrm{d}x$$

with ϕ_i from the previous assignment and $\bar{i}, \bar{j} = 1, 2$. How can one relate $\bar{i}, \bar{j}$ for given k with i, j using `e2p`? **Hint:** In 1D G is a number, so $G = G^T = 1/$`edet`.

step 4) **construction of global matrix:** The construction of the Galerkin matrix for $c = 0$ is done in the MATLAB-lines below. Please study the code `elliptic1d.m` from the ISIS 2 page and explain in detail how this works (in particular the boundary conditions).

```
%% build matrices
ii = zeros(nelement,nphi^2); % sparse i-index
jj = zeros(nelement,nphi^2); % sparse j-index
aa = zeros(nelement,nphi^2); % entry of Galerkin matrix
bb = zeros(nelement,nphi^2); % entry in mass-matrix (to build rhs)

%% build global from local
for k=1:nelement                  % loop over elements
    [edet,dFinv]  = generatetransformation(k,e2p,x); % compute map

    % build local matrices (mass, stiffness, ...)
    sloc = localstiff(edet,dFinv);      % element stiffness matrix
    mloc = localmass(edet);             % element mass matrix

    % compute i,j indices of the global matrix
    ii( k,: ) = [e2p(k,1) e2p(k,2) e2p(k,1) e2p(k,2)]; % local-to-global
    jj( k,: ) = [e2p(k,1) e2p(k,1) e2p(k,2) e2p(k,2)]; % local-to-global

    % compute a(i,j) values of the global matrix
    aa( k,: ) = sloc(:);
    bb( k,: ) = mloc(:);
end
% create sparse matrices
A=sparse(ii(:),jj(:),aa(:));
M=sparse(ii(:),jj(:),bb(:));
```

(b) Modify `elliptic1d.m` to `elliptic1dwithc.m` to solve the problem for $c = 1$ and $f(x) \equiv 1$. Compare with the exact solution $u = e^{-x}(1 - e^x)(e^x - e)/(1 + e)$.

(c) Modify `elliptic1d.m` into `elliptic1dinhom.m`, to compute the solution with $f(x) \equiv 2$, $c = 0$ and inhomogeneous Dirichlet boundary conditions $u(0) = 1$ and $u(1) = 0$ by modification of f as explained in the lecture. Compare with the exact solution.

(d) Modify `elliptic1d.m` into `elliptic1djump.m`, so that you compute a FEM solution of assignment 6, exercise 3c). Compare with the exact solution.

Lesson: How to write a general FEM program in the simplest case.

total sum: 23 points