Technische Universität Graz
Institut für Angewandte Mathematik
Prof. Dr. Olaf Steinbach
Julia Hauser

Numerische Mathematik 3

Finite Elemente - Erste Schritte, 3. April 2019

Teil 1 - 1D Finite Element Methode

Wir beschäftigen uns nun zunächst mit dem 1D Problem

$$-u''(x) + u(x) = 0 \quad \text{für } x \in (a,b), \qquad u'(a) = g_N(a), \quad u'(b) = g_N(b)$$

Betrachte dafür die Zerlegung $a = x_1 < x_2 < \cdots < x_{M-1} < x_M = b$ des Intervalls $(a,b) \subset \mathbb{R}$ in N Elemente $\tau_l = (x_{l_1}, x_{l_2})$ mit Maschenweite h.
Gegeben sei

- das Referenzelement $\tau = (0,1)$,
- die auf dem Referenzelement gegebenen Formfunktionen $\psi_1^1(\xi) = 1 - \xi$ und $\psi_2^1(\xi) = \xi$,
- die affin lineare Transformation $F_l(\xi) := x_{l_1} + (x_{l_2} - x_{l_1})\xi$ von τ auf τ_l,
- der lokale Index $\alpha(k,l) \in \{1, 2\}$, welcher durch $\alpha(l_1, l) = 1$ und $\alpha(l_2, l) = 2$ definiert ist.

Als endlichen Teilraum des Ansatz und Testraumes wählen wir den Raum der stückweise linearen und global stetigen Funktionen $S_h^1(a,b)$. Die Basis des Raumes ist durch die Basisfunktionen

$$\varphi_k(x) = \begin{cases} \psi^1_{\alpha(k,l)}(F_l^{-1}(x)) & \text{für } x \in \overline{\tau_l}, \\ 0 & \text{sonst,} \end{cases} \qquad k = 1, \ldots, M$$

gegeben. Beachte, dass wir im Vergleich zum Tutorium nun auch ein Referenzelement in einer Dimension behandeln. Bevor das Gleichungssystem ausstellen, benötigen wir noch die Implementation der Massematrix. Diese möchten wir durch Assemblierung erstellen. Berechne und implementiere dafür die lokale Massematrix in Aufgabe 6.

Aufgabe 6: (MATLAB)

a) Zeige für das Element $\tau_l = (x_{l_1}, x_{l_2})$, dass die lokale Massematrix $M_h^l \in \mathbb{R}^{2\times 2}$ durch

$$M_h^l := \begin{pmatrix} \int_{x_{l_1}}^{x_{l_2}} \varphi_{l_1}(x)\varphi_{l_1}(x)\mathrm{d}x & \int_{x_{l_1}}^{x_{l_2}} \varphi_{l_2}(x)\varphi_{l_1}(x)\mathrm{d}x \\ \int_{x_{l_1}}^{x_{l_2}} \varphi_{l_2}(x)\varphi_{l_1}(x)\mathrm{d}x & \int_{x_{l_1}}^{x_{l_2}} \varphi_{l_2}(x)\varphi_{l_2}(x)\mathrm{d}x \end{pmatrix} = \frac{|x_{l_2} - x_{l_1}|}{6} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

gegeben ist. Verwende dabei die Transformation auf das Referenzelement.

b) Die Zerlegung wird durch die gegebenen Dateien `coordinates1.dat` und `elements1.dat` beschrieben. In der Datei `coordinates1.dat` ist die k-te Zeile die Koordinate des Knotens x_k. In der Datei `elements1.dat` beschreibt die l-te Zeile die Indizes l_1, l_2 der Knoten des Elements τ_l. Skizziere die gegebene Zerlegung. Implementiere in MATLAB die Routinen

```
function [M_el] = MassElemMat1D(x_l1, x_l2)
% Berechnung der lokalen Massematrix für das Intervall (x_l1, x_l2)
%
% x_l1 ... linker Eckpunkt des Intervalls (x_l1, x_l2)
% x_l2 ... rechter Eckpunkt des Intervalls (x_l1, x_l2)
% M_el ... lokale Massematrix für das Intervall (x_l1, x_l2)
```

um die lokalen Matrix M_h^l für ein Element $\tau_l = (x_{l_1}, x_{l_2})$ zu berechnen. Teste das Programm mit der gegebenen Zerlegung in `Mesh/level1/`.

Nun haben wir alle nötigen Routinen um das ganze Programm zu schreiben. Betrachte dazu das Problem

$$\begin{aligned} -u''(x) + u(x) &= 0 \quad \text{für } x \in (a,b), \\ -u'(a) &= g_N(a) =: \text{na}, \\ u'(b) &= g_N(b) =: \text{nb}, \end{aligned}$$

wobei $g_N \colon \{a,b\} \to \mathbb{R}$ ein gegebenes Neumann-Datum ist. Die dazugehörige Variationsformulierung ist:
Gesucht ist $u \in H^1(a,b)$ sodass

$$\int_a^b u'(x)v'(x)\ dx + \int_a^b u(x)v(x)\ dx = g_N(b)v(b) + g_N(a)v(a)$$

für alle $v \in H^1(a,b)$.
Implementiere nun das zugehörige Gleichungssystem in Aufgabe 7.

Aufgabe 7: (MATLAB) Für eine Zerlegung wie in Aufgabe 6 leite die diskrete Variationsformulierung für den Raum $S_h^1(a,b) \subset H^1(a,b)$ und das resultierende Gleichungssystem $(A_h + M_h)\vec{u} = \vec{F}$ mit der Steifigkeitsmatrix $A_h \in \mathbb{R}^{M\times M}$, der Massematrix $M_h \in \mathbb{R}^{M\times M}$ und der rechten Seite $\vec{F} \in \mathbb{R}^M$ her. Gebe die Matrixeinträge in Abhängigkeit von den Basisfunktionen an.
Weiters soll dieses diskrete Variationsproblem gelöst werden. Dafür gehe wie folgt vor:

a) Imlementiere die Routinen

```
function [mass] = Mass1D(coordinates, elements)
% Massematrix
%
% coordinates ... Koordinatenvektor
% elements ..... Matrix über die Indizes der zu den Elementen zugehöringen Knoten
% mass ......... Massematrix
```

und

```
function [stiff] = Steif1D(coordinates, elements)
% Steifigkeitsmatrix
%
% coordinates ... Koordinatenvektor
% elements ..... Matrix über die Indizes der zu den Elementen zugehöringen Knoten
% stiff ........ Steifigkeitsmatrix
```

um die Massematrix M_h und die Steifigkeitsmatrix A_h zu assemblieren, dabei laufe über die Elemente in `elements.dat` und verwende die Routinen `MassElemMat1D` und `SteifElemMat1D` aus Aufgabe 6 und dem Tutorium. In `neumann.dat` ist die k-te Zeile der Index des Neumann-Knotens.

b) Implementiere die Routine

```
function [rhs] = rhs1D(coordinates, elements, neumann, gN)
% Rechte Seite mit Neumannteil
%
% coordinates ... Koordinatenvektor
% elements ..... Matrix über die Indizes der zu den Elementen zugehöringen Knoten
% neumann ....... Vektor mit den Indizes der Neumannknoten
% gN ............ Neumannranddaten
% rhs ........... Rechte Seite mit Neumannteil
```

um die rechte Seite $\vec{F}$ zu berechnen, verwende dafür aus dem Tutorium die Routine `LastElemVec1D`. Laufe danach über die Knoten in der Datei `neumann.dat` um die Neumanndaten an den richten Zeilen zu addieren. Achte darauf, dass dabei die Einträge richtig zugewiesen werden.

c) Weiters implementiere

```
function [uh] = LoeseNeumann1D(coordinates, elements, neumann, gN)
% Löst die homogene Yukawa Gleichung mit Neumannranddaten
%
% coordinates ... Koordinatenvektor
% elements ..... Matrix über die Indizes der zu den Elementen zugehöringen Knoten
% neumann ....... Vektor mit den Indizes der Neumannknoten
% gN ............ Neumannranddaten
% uh ............ Lösungsvektor
```

um das gegebene Neumann-Randwertproblem zu lösen.

Teste das Programm für Level 1, 2 und mit den Werten `na` $= 2$ und `nb` $= 1$. Plotte die Näherungslösungen $u_h \in S_h^1(a,b)$ mittels `plot(coordinates,uh)` zusammen mit der exakten Lösung

$$u(x) = \frac{1}{e^{b-a} - e^{a-b}} \left(\left(e^{-a} g_N(b) + e^{-b} g_N(a) \right) e^x + \left(e^a g_N(b) + e^b g_N(a) \right) e^{-x} \right).$$

Teil 2 - 2D Finite Element Methode

Um ein 2D Problem lösen zu können, gehen wir wieder zurück auf die Teilelemente der Zerlegung und berechnen alle lokalen Einträge. Sei $\tau_l \subset \mathbb{R}^2$ ein beliebiges Dreieck mit Knoten $x_{l_1}, x_{l_2}, x_{l_3}$ und $e_j \subset \mathbb{R}^2$ eine beliebige Kante mit Knoten x_{j_1}, x_{j_2}. Gegeben sei

- das Referenzelement $\tau = \{ \underline{\xi} = (\xi_1, \xi_2)^T \mid 0 \leq \xi_1 + \xi_2 \leq 1 \}$,
- die auf dem Referenzelement definierten Formfunktionen $\psi_1^1(\xi_1, \xi_2) = 1 - \xi_1 - \xi_2$, $\psi_2^1(\xi_1, \xi_2) = \xi_1$ und $\psi_3^1(\xi_1, \xi_2) = \xi_2$.
- die affin lineare Transformation $F_l(\underline{\xi}) := \underline{x}_{l_1} + J_l \underline{\xi}$ von τ auf τ_l mit

$$J_l = \begin{pmatrix} x_{l_2} - x_{l_1} & x_{l_3} - x_{l_1} \\ y_{l_2} - y_{l_1} & y_{l_3} - y_{l_1} \end{pmatrix},$$

- der lokale Index $\alpha(k,l) \in \{1,\ 2,\ 3\}$, welcher durch $\alpha(l_1, l) = 1$, $\alpha(l_2, l) = 2$ und $\alpha(l_2, l) = 3$ definiert ist,

- die Basisfunktionen

$$\varphi_k(x) = \begin{cases} \psi^1_{\alpha(k,l)}(F_l^{-1}(x)) & \text{für } x \in \overline{\tau_l}, \\ 0 & \text{sonst,} \end{cases} \qquad k = 1, \ldots, M$$

Aufgabe 8: (MATLAB)

a) Analog zu Aufgabe 6 implementiere die lokale Massematrix $M_h^l \in \mathbb{R}^{3\times 3}$ für das Dreieck τ_l und lineare Ansatzfunktionen. Die Einträge der lokalen Matrix sind durch

$$M_h^l[i,j] = \int_{\tau_l} \varphi_{l_j}(x)\varphi_{l_i}(x)\mathrm{d}x$$

für $i, j = 1, \ldots, 3$ gegeben. Zeige, dass gilt

$$M_h^l = \frac{|detJ_l|}{24}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}.$$

Führe dafür wieder die Transformation auf das Referenzelement τ durch. Weiters implementiere die Routine

```
function [M_el] = mass2dEl(x_l1, x_l2, x_l3)
% Berechnung der lokalen Massematrix für das Dreieck mit den
% Eckpunkten x_l1, x_l2, x_l3
%
% x_l1 ... Erster Eckpunkt des Dreiecks
% x_l2 ... Zweiter Eckpunkt des Dreiecks
% x_l3 ... Dritter Eckpunkt des Dreiecks
% M_el ... Lokale Massematrix des Dreiecks
```

und teste diese für das gegebene Dreieck $x_{l_1} = (0,0)^\top$, $x_{l_2} = (1,0)^\top$, $x_{l_3} = (0,1)^\top$.

b) Analog zu a) implementiere man für gegebene Funktionen $f \in C(\overline{\tau_l})$, $g_N \in C(e_j)$ in MATLAB die Berechnung der lokalen Vektoren

$$\vec{F}_f^l := \begin{pmatrix} \int_{\tau_l} f(x)\varphi_{l_1}(x)\mathrm{d}x \\ \int_{\tau_l} f(x)\varphi_{l_2}(x)\mathrm{d}x \\ \int_{\tau_l} f(x)\varphi_{l_3}(x)\mathrm{d}x \end{pmatrix}$$

unter Verwendung der Mittelpunktsformel auf dem Dreieck τ_l

$$\int_{\tau_l} f(x)\varphi_k(x)\mathrm{d}x \approx |\tau_l|\, f(s_{\tau_l})\varphi_k(s_{\tau_l})$$

und

$$\vec{F}_N^l := \begin{pmatrix} \int_{e_j} g_N(x)\varphi_{j_1}(x)\mathrm{d}s_x \\ \int_{e_j} g_N(x)\varphi_{j_2}(x)\mathrm{d}s_x \end{pmatrix}$$

unter Verwendung der Mittelpunktsformel auf der Kante e_j

$$\int_{e_j} g_N(x)\varphi_k(x)\mathrm{d}s_x \approx |e_j|\, g_N(m_{e_j})\varphi_k(m_{e_j}).$$

Teste für $f(x_1, x_2) = g_N(x_1, x_2) = 1$ die Implementierungen für das Dreieck $x_{l_1} = (0,0)^\top$, $x_{l_2} = (1,0)^\top$, $x_{l_3} = (0,1)^\top$ und der Kante zwischen den Eckpunkten x_{l_1} und x_{l_2}.

Um die Berechnungen evaluieren zu können, benötigen wir die Berechnung der L^2- und H^1-Norm. Implementiere dafür die lokale Fehlerberechnung in Aufgabe 9.

Aufgabe 9: (MATLAB) Implementiere die lokale Fehlerberechnung $\|u - u_h\|$ (unter der Annahme, dass die exakte Lösung u und ∇u bekannt und gegeben sind) in der

a) $L^2(\Omega)$-Norm durch numerische Integration mit einer 7-Punkt-Formel. Implementiere dafür die Routine

```
function [errl2] = errorL2El(u_exact, u_h, x_l1, x_l2, x_l3, points, weights)
% Berechnung des lokalen L2-Fehlers
%
% u_exact .... Funktion u
% u_h ........ Vektor der berechneten Approximation in den Knoten
% x_l1 ....... Koordinaten des ersten Knoten
% x_l2 ....... Koordinaten des zweiten Knoten
% x_l3 ....... Koordinaten des dritten Knoten
% points ..... Gaussquadraturpunkte
% weights .... Gaussquadraturgewichte
% errl2 ...... Berechneter Fehler
```

Verwende die gegebene Routine `gauss7Points()`, welche die Integrationspunkte und -gewichte im Referenzdreieck liefert.

b) $H^1(\Omega)$-Seminorm ($\|\nabla .\|_{L^2}$) durch numerische Integration mit einer 7-Punkt-Formel. Überlege dafür wie ∇u_h lokal aussieht und verwende die Formel $\nabla_x \varphi_k = J_l^{-1} \nabla_\xi \psi_{\alpha(k,l)}$. Schreibe die Routine

```
function [errh1] = errorH1SemiEl(D1_u_exact, D2_u_exact, u_h, x_l1, x_l2, x_l3, ...
                                 points, weights)
% Berechnung des lokalen H1-Seminorm-Fehlers
%
% D1_u_exact .. Funktion der Ableitung von u nach x_1
% D2_u_exact .. Funktion der Ableitung von u nach x_2
% u_h ......... Vektor der berechneten Approximation in den Knoten
% x_l1 ........ Koordinaten des ersten Knoten
% x_l2 ........ Koordinaten des zweiten Knoten
% x_l3 ........ Koordinaten des dritten Knoten
% points ...... Gaussquadraturpunkte
% weights ..... Gaussquadraturgewichte
% errh1 ....... Berechneter Fehler
```