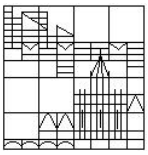

Universität Konstanz
FB Mathematik & Statistik
Prof. Dr. M. Junk
Sebastian Sahli

Ausgabe: 15.06.2016
Abgabe: 22.06.2016

Übungen zu Numerik PDGL II

Blatt 10

Aufgabe 1:

Betrachten Sie die elliptische Randwertaufgabe mit nichtlinearer Funktion f

$$\Delta u(x,y) = f(u), \quad (x,y) \in \Omega \subset \mathbb{R}^2,$$

$$u(x,y) = 0, \quad (x,y) \in \partial\Omega,$$

wobei $f : \mathbb{R} \mapsto \mathbb{R}\backslash\{0\}$ stetig, beschränkt und streng monoton wachsend ist.

1) Geben Sie eine sinnvolle schwache Formulierung an und schreiben Sie diese als $N(u) = 0$ mit
$$N \; : \; H_0^1(\Omega) \to (H_0^1(\Omega))'.$$

2) Das Newton-Verfahren wird eingesetzt um $N(u) = 0$ zu lösen. Formulieren Sie die Berechnungsvorschrift so, dass die beteiligte FEM-Grundaufgabe erkennbar ist.

Aufgabe 2 (Programmieraufgabe):

Diese Aufgabe schließt direkt an Blatt 8 an. Den aktuellen Stand des Projekts finden Sie auf der Vorlesungshomepage.

- Auf der Vorlesungshomepage finden Sie ebenfalls ein m-File, das eine Funktion `gauss(n)` beeinhaltet, die die Stützstellen und Gewichte der Gauß-Legendre-Quadratur vom Grad n berechnet. Verwenden Sie den dortigen Code, um – analog zur Klasse `TrapezoidalRule` vom letzten Blatt – eine spezielle `MultiEvaluationFunctional`-Klasse `GaussLegendre` zu erstellen, die eine entsprechende Quadraturformel darstellt.

- Neben den *Funktionen*systemen werden wir auch Systeme von *Funktionalen* benötigen. Erstellen Sie daher analog zur `FunctionSystem`-Klasse eine Klasse `FunctionalSystem` mit abstrakter `evaluate`-Methode. Erstellen Sie eine spezielle `FunctionalSystem`-Klasse, die einen Vektor $(x_1, \ldots, x_n)$ übergeben bekommt und das Funktionalsystem $(\delta_{x_i})_{i=1,\ldots,n}$ beschreibt.

- In Aufgabe 6 von Blatt 1 haben wir gesehen (und werden es auch an weiteren Stellen feststellen), dass es manchmal nötig ist, alle Funktionale eines Funktionalsystems $\{\Lambda_1, \ldots, \Lambda_n\}$ an allen Funktionen eines Funktionensystems $\{\varphi_1, \ldots, \varphi_m\}$ auf einmal auszuwerten.

  Erweitern Sie die `FunctionalSystem`-Klasse um eine zunächst abstrakte Methode `evaluateFunctionSystem`, die ein Objekt vom Typ `FunctionSystem` übergeben bekommen und die Matrix $(\Lambda_i(\varphi_j))_{\substack{i=1,\ldots,n \\ j=1,\ldots,m}} \in \mathbb{R}^{n\times m}$ zurückgibt. Implementieren Sie diese Methode für alle bekannten Spezialisierungen von `FunctionalSystem`. Erweiten Sie hierbei gegbenenfalls die `FunctionSystem`- oder andere Klassen um benötigte Hilfsfunktionen.