Prof. Dr. Angela Kunoth
Samuel Leweke

Ausgabe: 10.06.2020

Übung 7

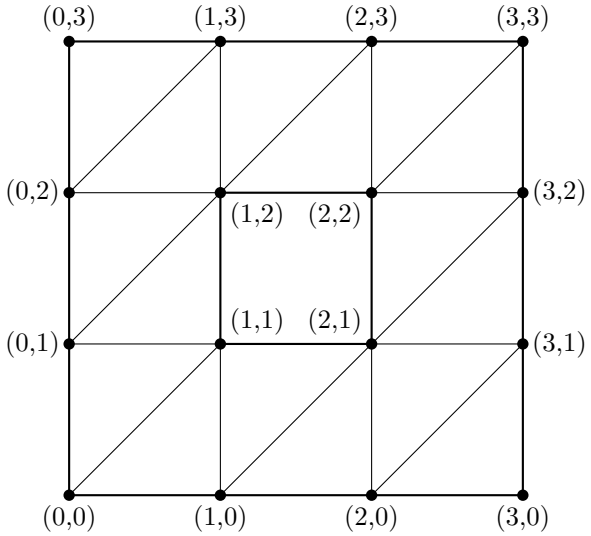
Abgabe: 17.06.2020, 12:00 Uhr

In dieser Übung wird die Wärmeleitungsgleichung mit linearen P1-Elementen auf einer Dreieckszerlegung adaptiv gelöst. Betrachten Sie das Problem

$$\frac{\partial u}{\partial t} - \Delta u = f \qquad \text{in } (0,T)\times\Omega, \tag{1a}$$
$$u|_{x\in\partial\Omega} = 0 \qquad \text{in } (0,T), \tag{1b}$$
$$u|_{t=0} = 0 \qquad \text{in } \Omega, \tag{1c}$$

wobei $T = 1$ ist und das Gebiet $\Omega := (0,3)\times(0,3)\setminus(1,2)\times(1,2)$ ein Loch enthält. Die rechte Seite f ist gegeben durch

$$f(t,\boldsymbol{x}) = \min(1,25t)\cdot\exp\left(-\frac{3}{4}\|\boldsymbol{x}-\boldsymbol{p}(t)\|_2^2\right), \qquad \boldsymbol{p}(t) = \frac{9}{8}\begin{pmatrix}\cos(2\pi t)\\ \sin(2\pi t)\end{pmatrix} + \frac{3}{2}\begin{pmatrix}1\\1\end{pmatrix}. \tag{2}$$



Für diese Übung wird der Matlab Code `p1afem` verwendet [FPW11]. Unter `https://www.asc.tuwien.ac.at/~praetorius/matlab/` steht sowohl der Code als auch eine Beschreibung im PDF-Format zur Verfügung.

Zunächst wird (1) in der Zeit mit Hilfe des θ-Verfahrens diskretisiert. Dazu wird $(0,T)$ in Intervalle (t_n, t_{n+1}) der Größe $h_t = T/N_t$ mit $N_t \in \mathbb{N}$ zerlegt, wobei $t_n = nh_t$, $n = 0,\dots,N_t$. Das θ-Verfahren für die gewöhnliche Differentialgleichung

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = \boldsymbol{g}(t,\boldsymbol{v})$$

ist gegeben durch

$$\frac{\boldsymbol{v}^n - \boldsymbol{v}^{n-1}}{h_t} = \theta\boldsymbol{g}(t_n,\boldsymbol{v}^n) + (1-\theta)\boldsymbol{g}\big(t_{n-1},\boldsymbol{v}^{n-1}\big), \tag{3}$$

wobei $\theta\in[0,1]$ und $\boldsymbol{v}^n$ eine Approximation von $\boldsymbol{v}(t_n)$ ist. Für $\theta = 1$ ergibt sich also das implizite Euler-Verfahren, für $\theta = 0$ das explizite Euler-Verfahren und für $\theta = 1/2$ das Crank-Nicolson-Verfahren.

Wird das θ-Verfahren (3) auf (1) angewandt und anschließend mit finiten Elementen im Raum diskretisiert, ergibt sich die diskrete Form

$$(\mathbf{M} + h_t\theta\mathbf{A})\,\boldsymbol{u}^n = \mathbf{M}\boldsymbol{u}^{n-1} - h_t(1-\theta)\mathbf{A}\boldsymbol{u}^{n-1} + h_t\left((1-\theta)\boldsymbol{F}^n + \theta\boldsymbol{F}^{n-1}\right), \tag{4}$$

wobei $\mathbf{A}$ die Steifigkeitsmatrix bezeichnet, $\mathbf{M}$ die Massenmatrix und $\boldsymbol{F}^n$ die diskrete Form von f zum Zeitpunkt t_n.

In jedem Zeitschritt ist also ein elliptisches Problem zu lösen. Bei der räumlich adaptiven Lösung dieses Problems ergibt sich die Schwierigkeit, dass $\boldsymbol{u}^{n-1}$ auf einer anderen Zerlegung als $\boldsymbol{u}^n$ definiert ist (bzw. in einem anderen diskreten Teilraum V_n von $H_0^1(\Omega)$). Das heißt, in (4) treten rechteckige Matrizen auf, die durch die mangelnde Übereinstimmung der Dreiecke in beiden Zerlegungen sehr schwer aufzustellen sind.

Als Alternative wird in dieser Übung die adaptive Anpassung der Zerlegung nur in jedem 5. Zeitschritt durchgeführt und die Lösung mittels Interpolation auf die neue Zerlegung übertragen. Dies zieht zwar einen zusätzlichen Interpolationsfehler nach sich und sorgt dafür, dass die Zerlegung der Lösungsstruktur zeitlich hinterher hängt, stellt aber einen pragmatischen und durchführbaren Ansatz dar. Beim Übergang zwischen zwei Zerlegungen (gekennzeichnet durch die Superskripte n und $n-1$) mit N_{n-1} und N_n Knoten wird also statt (4) das folgende System gelöst:

$$(\mathbf{M}^n + h_t\theta\mathbf{A}^n)\,\boldsymbol{u}^n = \mathbf{M}^n\mathbf{I}_{n-1}^n\boldsymbol{u}^{n-1} - h_t(1-\theta)\mathbf{A}^n\mathbf{I}_{n-1}^n\boldsymbol{u}^{n-1} + h_t\left((1-\theta)\boldsymbol{F}^n + \theta\tilde{\boldsymbol{F}}^{n-1}\right), \tag{5}$$

wobei der Interpolationsoperator $\mathbf{I}_{n-1}^n$ aufgrund der nodalen Basen $\{\varphi_j^n : j = 1,\dots,N_n\}$ und $\{\varphi_j^{n-1} : j = 1,\dots,N_{n-1}\}$ durch das Auswerten an den Knoten $\{\boldsymbol{x}_j^n : j = 1,\dots,N_n\}$, der neuen Zerlegung definiert ist:

$$\mathbf{I}_{n-1}^n\colon \mathbb{R}^{N_{n-1}} \to \mathbb{R}^{N_n}, \qquad \left(\boldsymbol{I}_{n-1}^n\boldsymbol{u}^{n-1}\right)_j = \sum_{i=1}^{N_{n-1}} u_i^{n-1}\varphi_i^{n-1}\left(\boldsymbol{x}_j^n\right). \tag{6}$$

Die diskrete Form $\tilde{\boldsymbol{F}}^{n-1}$ der rechten Seite f des vorherigen Zeitschritts wird durch erneute Integration bezüglich der neuen Basis bestimmt:

$$\left(\tilde{\boldsymbol{F}}^{n-1}\right)_j = \int_\Omega \varphi_j^n(\boldsymbol{x}) f(t_{n-1},\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \qquad \forall j = 1,\dots,N_n.$$

Damit die Anzahl der Freiheitsgrade nicht beliebig anwächst, muss die Zerlegung regelmäßig vergröbert werden. So kann die Lösung in Regionen, in denen keine feine Auflösung mehr benötigt wird, wieder mit weniger Freiheitsgraden approximiert werden. Insgesamt ergibt sich daher der folgende Grob-Algorithmus:

1. Baue Grundzerlegung auf: Erstelle Zerlegung gemäß Abbildung und verfeinere zweimal uniform
2. Setze $n \coloneqq 1$ und $\boldsymbol{u}_n \coloneqq \mathbf{0}$
3. Wenn $(n-1) \equiv 0 \bmod 5$:
   a) Setze $k \coloneqq 0$
   b) Löse (5)
   c) Berechne Fehlerindikatoren
   d) Wenn k gerade: Verfeinere die Elemente mit den größten Fehlern, die zusammen 30% des Gesamtfehlers ausmachen (Dörfler-Marking)
   Wenn k ungerade: Vergröbere die Elemente mit den kleinsten Fehlern, die zusammen 85% des Gesamtfehlers ausmachen (Dörfler-Marking), aber vergröbere nicht über die Grundzerlegung hinaus
   e) Setze $k \coloneqq k + 1$
   f) Wenn $k < 6$: Gehe zu b)
4. Wenn $(n-1) \not\equiv 0 \bmod 5$: Löse (4)
5. Setze $n \coloneqq n + 1$
6. Wenn $n \leq N_t$: Gehe zu 3.

Aufgaben (20 Punkte)

a) Leiten Sie (4) her.

b) Setzen Sie $u^n = \sum_{i=1}^{N_n} \varphi_i^n u_i^n \in V_n$ und $u^{n-1} = \sum_{i=1}^{N_{n-1}} \varphi_i^{n-1} u_i^{n-1} \in V_{n-1}$, wobei $V_n, V_{n-1} \subset H_0^1(\Omega)$ zwei diskrete Teilräume zu verschiedenen Zerlegungen und $\varphi_i^n, \varphi_i^{n-1}$ die entsprechenden Basisfunktionen sind. Erklären Sie, wie (potentiell) rechteckige Matrizen bei einem Wechsel der Zerlegung zwischen zwei Zeitschritten in (4) entstehen.

c) Geben Sie einen geeigneten Fehlerindikator an.

d) Schreiben Sie eine Matlab-Funktion `uNew = InterpolateSolution(coordsOld, elementsOld, coordsNew, uOld)`, die die Koeffizienten `uOld` der Lösung $\boldsymbol{u}^n$ auf der Zerlegung mit den Knoten `coordsOld` und Elementen `elementsOld` an den Knoten `coordsNew` der neuen Zerlegung auswertet und in `uNew` zurückgibt (siehe (6)). Den Aufbau der Knoten- und Elemente-Matrizen entnehmen Sie [FPW11].

e) Schreiben Sie eine Funktion `u = SolveDirichlet(B, g, dirichlet)`, welche das System $\mathtt{B}\mathtt{u} = \mathtt{g}$ mit homogenen Dirichlet-Randbedingungen löst. Das heißt, die Freiheitsgrade werden aufgeteilt in innere Knoten (Indexmenge $\mathcal{I}$) und Knoten auf dem Rand (Indexmenge $\mathcal{R}$, gegeben durch die Liste der Dirichlet-Kanten `dirichlet`). Damit ergibt sich $\mathtt{u}_\mathcal{R} = \mathbf{0}$ und das zu lösende System $\mathtt{B}_{\mathcal{I},\mathcal{I}}\mathtt{u}_\mathcal{I} = \mathtt{g}_\mathcal{I}$. Verwenden Sie für Letzteres den Backslash-Operator.

f) Implementieren Sie den Grob-Algorithmus mit $T = 1$, $N_t = 100$ und $\theta = 1/2$. Verwenden Sie zum Verfeinern und Vergröbern die Newest-Vertex-Bisection Methode (siehe Hinweise). Zeichnen Sie nach jedem Zeitschritt die Zerlegung und die Lösung auf und erstellen Sie am Ende ein Video der Lösung. Fixieren Sie dazu die Grenzen der z-Achse.

Hinweise

- Ein Punkt liegt in einem Dreieck, wenn er auf der gleichen Seite jeder Kante liegt.
- Nutzen Sie die Matlab-Funktionen `VideoWriter`, `writeVideo` und `getframe`.
- Nutzen Sie die Matlab-Funktionen `refineRGB`, `refineNVB` und `coarsenNVB` aus der `p1afem` Toolbox.
- Berechnen Sie die Steifigkeits- und Massenmatrizen sowie die rechten Seiten $\boldsymbol{F}^n$ und $\boldsymbol{F}^{n-1}$ so selten wie möglich.
- Verwenden Sie die folgenden bereitgestellten Matlab-Funktionen:
  - `etaR = computeEtaR_heat(u,coordinates,elements,dirichlet,f,un,unm1,hTime)` berechnet die Fehlerindikatoren für jedes Element. Das Residuum ist dabei $\mathtt{f} - (\mathtt{un} - \mathtt{unm1})/\mathtt{hTime}$. Die Koeffizienten $\boldsymbol{u}^n$ der Lösung werden als Vektor `u` übergeben, `coordinates`, `elements`, `dirichlet` bezeichnen die entsprechenden Listen der Zerlegung im Format der `p1afem` Toolbox, `f` ist ein Function-Handle mit Signatur `v = f(x,y)`, `unm1` enthält die Koeffizienten von $\boldsymbol{u}^{n-1}$ (auf der gleichen Zerlegung) und `hTime` ist h_t.
  - `[M, A] = MatAssemblyTri(coordinates, elements)` berechnet die Massenmatrix `M` und die Steifigkeitsmatrix `A` zur durch `coordinates` und `elements` gegebenen Zerlegung.
  - `b = VecAssemblyTri(coordinates, elements, f)` berechnet den Vektor `b` mit

$$\mathtt{b}_j = \int_\Omega \varphi_j(\boldsymbol{x}) f(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \tag{7}$$

    für die durch `coordinates` und `elements` gegebene Zerlegung. Die Funktion f wird als Function-Handle mit der Signatur `v = f(x,y)` übergeben.

Literaturverzeichnis

[FPW11] S. Funken, D. Praetorius und P. Wissgott. Efficient implementation of adaptive P1-FEM in Matlab. *Computational Methods in Applied Mathematics*, 11(4), 2011. DOI: 10.2478/cmam-2011-0026.