Universität Konstanz *WS 13/14*
Fachbereich Mathematik und Statistik
Prof. Dr. J.Schropp, J.Budday

ÜBUNGEN ZU **Theorie und Numerik partieller Differentialgleichungen**

http://www.math.uni-konstanz.de/∼schropp/thnumpdg.html

Blatt 10 Ausgabe: 13.01.2014 **Abgabe: 20.01.2014**

Aufgabe 1 (Theorie) (6 Punkte)

Es sei $A \in \mathbb{R}^{N,N}$ symmetrisch und positiv definit. Die Vektoren $d_0, d_1, \ldots, d_{N-1} \in \mathbb{R}^N$ heißen konjugiert oder A-orthogonal, falls $d_i^T A d_j = 0$ für $i \neq j$, $0 \leq i, j \leq N-1$ und $d_i^T A d_i \neq 0$ für $0 \leq i \leq N-1$.
a) Verifizieren Sie, dass $d_0, d_1, \ldots, d_{N-1}$ eine Basis des $\mathbb{R}^N$ bilden.
b) Vorgelegt sei das Gleichungssystem $Ax = b$. Zeigen Sie: Zerlegt man die Lösung $x = A^{-1}b$ nach der Basis $d_0, \ldots, d_{N-1}$, so gilt

$$x = \sum_{k=0}^{N-1} \alpha_k d_k \quad \text{mit} \quad \alpha_k = \frac{d_k^T A x}{d_k^T A d_k},\ k = 0, \ldots, N-1.$$

c) Zeigen Sie: Für jedes $x_0 \in \mathbb{R}^N$ liefert die durch

$$x_{k+1} = x_k + \alpha_k d_k \quad \text{mit} \quad \alpha_k = \frac{-d_k^T (A x_k - b)}{d_k^T A d_k}$$

für $k \geq 0$ erzeugte Folge nach höchstens N Schritten die Lösung $x_N = A^{-1}b$.

Aufgabe 2 (Programmieraufgabe) (8 Punkte)

Ziel dieser Aufgabe ist der Vergleich einer iterativen Methode mit den klassischen direkten Lösungsmethoden im Fall von Finiten Differenzen Verfahren für elliptische Randwertprobleme. Als Basis benutzen Sie das Programm von Aufgabe 3, Blatt 9 mit $f(x, y) = 20$ und $g(x, y) = y \cos(4\pi x)$ mit $h = 1/M$, $M = 10$, 20, 40, 80.

Implementieren Sie das CG (*conjugated-gradient*) Verfahren:

CG-Algorithmus

Zu lösen: $Ax = b$ (A symmetrisch und positiv definit)

Gegeben: tol, x^0

Initialisierung:

$r^0 = b - Ax^0$, $p^0 = r^0$

Iteration:

Für $k = 0, 1, 2, \ldots$

$$\alpha_k = \frac{(p^k)^T r^k}{(p^k)^T A p^k}$$
$$x^{k+1} = x^k + \alpha_k p^k$$
$$r^{k+1} = r^k - \alpha_k A p^k$$
$$\beta_k = \frac{(Ap^k)^T r^{k+1}}{(Ap^k)^T p^k}$$
$$p^{k+1} = r^{k+1} - \beta_k p^k$$

Visualisieren Sie das Verhalten der Norm des Residuums $\|r\|_2$ versus Anzahl der Iterationen k für das CG-Verfahren. Füllen Sie die folgende Tabelle aus. Die Zahlen in der zweiten Zeile der Tabelle sind die Werte von M und die in der ersten Spalte die von $\|r\|_2$.

CG-Verfahren				
$\|r\|_2 \backslash M$	10	20	40	80
10^{-1}				
10^{-2}				
10^{-4}				
10^{-6}				

Was beobachten Sie? Ist das CG-Verfahren für diese Situation im Vergleich zu den direkten klassischen Lösungsmethoden empfehlenswert? Verwenden Sie für den Vergleich zum Lösen von $Ax = b$ die LU-Zerlegung von A, d.h. $[L, U] = lu(A)$ und $x = U \setminus (L \setminus b)$.

Aufgabe 3 (Programmieraufgabe) (8 Punkte)

Wir betrachten die Randwertaufgabe

$$\begin{cases} -\Delta u(x,y) = f(x,y)\ ,\ (x,y) \in \Omega \\ u(x,y) = g(x,y)\ ,\ (x,y) \in \partial\Omega \end{cases}$$

für den Einheitskreis $\Omega := \{(x,y)^T \in \mathbb{R}^2 \mid x^2 + y^2 < 1\}$. Geben Sie eine interessante Lösung u der Randwertaufgabe vor und bestimmen Sie die Funktionen f und g so, dass u tatsächlich Lösung dieser Randwertaufgabe ist. Ziel ist es nun, diese Randwertaufgabe über der vorgegebenen krummlinig berandeten Menge Ω numerisch zu lösen. Für die Implementierung sollten Sie folgende Hinweise unbedingt beachten:

- Betten Sie den Einheitskreis Ω in ein Quadrat Q ein, das an allen 4 Seiten einen echt positiven Abstand zu $\partial\Omega$ aufweist. Erzeugen Sie mit

`[X,Y]=meshgrid(...)`

auf Q ein Gitter, das den Nullpunkt enthält, und finden Sie mit Hilfe des `find`-Befehls durch `I=find(X.^2+Y.^2<1)` die Nummern jener Gitterpunkte, die in Ω liegen.

- Zur Nummerierung der Gitterpunkte über Ω wird es hilfreich sein, die Nummer eines Gitterpunktes in die zugehörigen Indizes der Gittermatrix umrechnen zu können und umgekehrt. Programmieren Sie diese beiden Funktionen ν und ν^{-1} mit Hilfe der Nummern in `I` und den Befehlen `find`, `ind2sub` und `sub2ind` inline in Ihrem Code. (Bemerkung: $\nu(i,j)$ soll [ ] sein, wenn $x_{ij} \notin \Omega$. Beachten Sie, dass gilt: $x_{ij} = (ih, jh)$)

- Als nächstes müssen aus den gefundenen Gitterpunkten die numerischen Randpunkte ausfindig gemacht werden, also jene die zwar in Ω liegen, ihnen dort aber in mindestens einer Raumrichtung ein Gitternachbar aus Ω fehlt. Die einfachste Möglichkeit (jedoch nicht die zeiteffizienteste) ist zunächst eine Schleife über alle Ω-Gitterpunkte zu programmieren, in der die Existenz der 4 Nachbarn überprüft wird. Die Nummern der Nachbarn erhalten Sie mit den Funktionen ν und ν^{-1} (z.B. den östlichen Nachbarn mit `[i,j] =` ν^{-1}`(k)` und ν`(i+1,j)`).

- Stellen Sie das zu lösende Gleichungssystem auf und behandeln Sie die Randpunkte dabei wie es im Skript für solche Fälle beschrieben wurde.

- Verwenden Sie zur Visualisierung Ihrer numerischen Lösung das gesamte Gitter über Q. Damit die Lösung nachher jedoch nur direkt über Ω visualisiert wird und Matlab am Rand von Ω keine graphischen Übergänge interpoliert, gehen Sie wie folgt vor:

  Erzeugen Sie eine dem numerischen Q-Gitter entsprechend große Matrix mit `NaN`-Einträgen. Wenn also mit

  `[X,Y]=meshgrid(linspace(a,b,n),linspace(c,d,m))`

  das Gitter erzeugt wurde, so gelingt dies einfach durch `Z=NaN(m,n)`. Sind in `I` jene Nummern der Gitterpunkte abgespeichert worden, die über den `find`-Befehl als Punkte aus Ω identifiziert wurden, und in `u` die zugehörigen Werte Ihrer numerischen Lösung, so ersetzen Sie mit `Z(I)=u` die entsprechenden `NaN`-Einträge über Ω durch Ihre Lösungswerte. Plotten Sie nun mit Hilfe von `surf(X,Y,Z)`, so wird das graphische Interpolieren aufgrund der `NaN`-Einträge außerhalb von Ω verhindert.

allgemeine Hinweise zur Abgabe:

- Die Programmieraufgaben können in 2er-Gruppen bearbeitet werden.

- Abgabe der Programme per Email an den jeweiligen Tutor bis spätestens 12 Uhr am Abgabetag.

- In den Programmen muss jeder Schritt angemessen kommentiert sein und erklärt werden können.

- Abgabe der Theorieaufgaben in den entsprechenden Briefkasten vor F441 bis spätestens 12 Uhr am Abgabetag.