Institute for Analysis and Scientific Computing
Auzinger, Danczul, Faschingleitner, Heitzinger, Nannen,
Weinmüller, Wess

Abgabe bis: KW 2/3 18. Dezember 2017

Übung zur Numerischen Mathematik – Projekt 2

Aufgabe 4:
Ziel des Projektes ist, eine nichtlineare Differentialgleichung mit Hilfe des Newton-Verfahrens zu lösen. Dazu sei u die Lösung des Randwertproblems

$$-u''(x) + (u(x))^3 = f(x), \qquad x \in (0,1) \tag{1}$$

mit den zugehörigen homogenen Randbedingungen

$$u(0) = u(1) = 0. \tag{2}$$

a) Zur numerischen Lösung des Randwertproblems (1) mit den Randbedingungen (2) kann das Verfahren der Finite Differenzen Verfahren verwendet werden. Dabei werden die zweiten Ableitungen an Stützwerten $x_j := hj$ mit $h := 1/(N+1)$ durch Differenzenquotienten approximiert

$$u''(x_j) \approx D_h u(x_j) := \frac{1}{h^2}\left(u(x_{j-1}) - 2u(x_j) + u(x_{j+1})\right), \qquad j = 1, ..., N. \tag{3}$$

Zeigen Sie, dass auf diese Weise ein nichtlineares Gleichungssystem

$$F(\overline{u}_h) = 0 \tag{4}$$

mit den Unbekannten $\overline{u}_h \approx (u(x_1), u(x_2), ..., u(x_N))^\top \in \mathbb{R}^N$ mit $F : \mathbb{R}^N \to \mathbb{R}^N$ gewonnen werden kann.

b) Formulieren Sie in Pseudocode das Newton-Verfahren zur Lösung von (4). Geben Sie dabei die Funktion $F : \mathbb{R}^N \to \mathbb{R}^N$ sowie deren Jacobimatrix DF an. Beachten Sie die Randbedingungen (2).

c) Zeigen Sie, dass (4) eine Lösung besitzt. Dazu sollten Sie nachweisen, dass ein Minimierer des Funktionals

$$\mathcal{F} : \mathbb{R}^N \to \mathbb{R} \qquad \text{mit } \mathcal{F}\overline{u} := \frac{1}{2}\overline{u}^\top L_h \overline{u} - (f(x_1), ..., f(x_N))\overline{u} + \frac{1}{4}\sum_{j=1}^{N} \overline{u}_j^4 \tag{5}$$

mit geeignetem $L_h \in \mathbb{R}^{N\times N}$ eine Lösung $\overline{u}_h$ von (4) ist. Unter der Annahme, dass L_h symmetrisch positiv definit ist, können Sie dann die Existenz eines solchen Minimums nachweisen.

d) Schreiben Sie ein Programm, welches (4) mit Hilfe des Newton-Verfahrens löst. Testen Sie das Programm anhand der exakten Lösung $u(x) = \sin(\pi x)$ und der zugehörigen rechten Seite f. Beachten Sie, dass Sie zwei unterschiedliche Fehlerquellen haben: Die Schrittweite h des Finite Differenzen Verfahrens und den Fehler durch das Newton-Verfahren. Die folgenden Teilaufgaben sollten helfen einzuschätzen, in welchem Fall welcher Fehler dominant ist.

d1) Zur genaueren Untersuchung des Fehlers des Newton-Verfahrens wählen Sie zunächst eine sehr kleine, fixe Schrittweite h und beobachten Sie die Entwicklung des Fehlers in jeder einzelnen Newton-Iteration. Variieren Sie die Startwerte für $\overline{u}_h$ (z.B. Nullvektor, Zufallsvektor, ...). Welchen Einfluss hat der Startvektor auf die Anzahl der zu einer bestimmten Genauigkeit benötigten Anzahl von Newtoniterationen?

d2) Nutzen Sie ein geeignetes Abbruchkriterium und variieren Sie nun die Schrittweite $h = 2^{-2}, 2^{-3}$, usw. und beobachten Sie den Fehler zwischen $\overline{u}_h$ und der exakten Lösung $(u(x_1), ..., u(x_N))^\top$. Stellen Sie dabei sicher, dass der beobachtete Fehler nicht aus dem Newton-Verfahren sondern einzig aus der Diskretisierung der nichtlinearen Differentialgleichung (1) stammt. Geben Sie an, wie der Fehler von der Schrittweite h abhängt. Dabei kann der Matlab-Befehl *polyfit* hilfreich sein.

Aufgabe 5:
Eine Kurve in der Ebene sei in impliziter Form gegeben durch eine Gleichung $F(x, y) = 0$, wobei $F\colon \mathbb{R}^2 \to \mathbb{R}$ stetig differenzierbar, mit partiellen Ableitungen $\partial F/\partial x$ und $\partial F/\partial y$. Wir kennen einen Punkt (x_0, y_0) auf der Kurve. Gesucht ist der Verlauf der Kurve, bzw. zu erstellen ist die entsprechende Grafik.

a) Wir nehmen zunächst an, dass durchwegs gilt $\partial F/\partial y \neq 0$. Nach dem *Satz über implizite Funktionen* ist dann die Gleichung $F(x, y) = 0$ überall lokal eindeutig nach y auflösbar, und es existiert eine Auflösungsfunktion $y = f(x)$, die die Kurve in expliziter Weise beschreibt. Das bedeutet im Allgemeinen natürlich nicht, dass man f explizit angeben kann.

Implementieren und testen Sie folgenden numerischen Algorithmus: Wir wählen eine Schrittweite h und Gitterpunkte $x_j = x_0 + j\,h$, $j = 0, 1, 2, 3, \ldots$. Ausgehend von (x_0, y_0) gehen wir wie folgt vor $(j = 0, 1, 2, 3, \ldots)$:

(i) Prediktor-Schritt:

Lege die Tangente an die Kurve im aktuell ermittelten Kurvenpunkt (x_j, y_j) mittels impliziter Differentiation der Gleichung $F(x, y) = 0$. Für die Gleichung der Tangente schreiben wir $y = T_j(x)$. Dann ist $(x_{j+1}, T_j(x_{j+1}))$ eine Näherung für den nächsten gesuchten Kurvenpunkt (x_{j+1}, y_{j+1}).

(ii) Korrektor-Schritt:

Löse die Gleichung $F(x_{j+1}, z) = 0$ mit Hilfe des Newton-Verfahrens, ausgehend vom Startwert $z_0 := T(x_{j+1})$. Mit der Lösung z erhält man die Näherung für den neuen Kurvenpunkt: (x_{j+1}, y_{j+1}) mit $y_{j+1} = z$.

Mit der so erzeugten Punktmenge kann man die Kurve (näherungsweise) plotten, bzw. sie für ein gewähltes Beispiel mit der exakten Kurve vergleichen.

b) Betrachten Sie sodann ein Beispiel, bei dem die Voraussetzung $\partial F/\partial y \neq 0$ entlang der Kurve irgendwann verletzt ist, jedoch $\partial F/\partial x \neq 0$ gilt. Erweitern und testen Sie den Algorithmus für diesen Fall.

c) Was wäre zu tun, wenn der Fall $\partial F/\partial x = \partial F/\partial y = 0$ (bzw. 'sehr klein') auftritt?

d) Diese Aufgabe ist beliebig ausbaubar, z.B.

— adaptive Wahl der Schrittweite h, oder

— Parametrisierung der Kurve nach der Bogenlänge s (d.h. in der Form $(x(s), y(s))$, bei der die Länge des Tangentialvektors entlang der Kurve genau gleich 1 ist).

Aufgabe 6:
Die meisten Internetbenutzer kennen die Suchmaschine Google™. Deren Grundkonzept wurde von Larry Page und Sergey Brin entwickelt und basiert auf dem PageRank™ Algorithmus. Dieser betrachtet die Menge aller Webseiten der Mächtigkeit N und ordnet jeder Seite einen Rang zu. Stellt man sich das Internet als einen gerichteten Graphen aus Ecken (Webseiten) und Pfeilen (Verlinkungen) vor (vgl. Abb. 1), so lässt sich diese Struktur in einer Adjazenzmatrix $L \in \mathbb{R}^{N\times N}$ mathematisch realisieren:

$$L_{ij} = \begin{cases} 1, & \text{falls ein Link } \textbf{von} \text{ Seite } j \textbf{ zur} \text{ Seite } i \text{ existiert} \\ 0, & \text{sonst.} \end{cases}$$

Insbesondere gilt: $L_{ii} = 0$ für $i = 1, .., N$. Definiert man

$$R_i := \sum_j L_{ij} \qquad \text{und} \qquad C_j := \sum_i L_{ij},$$

so ist R_i bzw. C_j die Menge aller Verlinkungen zur Seite i bzw. von Seite j. R_i bezeichnet man auch als *Indegree* und C_j als *Outdegree*.
Für den Benutzer der Suchmaschine ist es nun von großem Interesse, dass die Webseiten nach Relevanz sortiert werden. Der PageRank Algorithmus berechnet einen Vektor $\mathbf{g} \in \mathbb{R}^N$, der als Komponente g_j diejenige Wahrscheinlichkeit (bzw. dasjenige Gewicht) enthält, mit der ein „Zufalls-Surfer" die Webseite j besuchen würde. Der „Zufalls-Surfer" ist ein Internetbenutzer, der sich *zufällig* von einem Link zum nächsten klickt. Das Gewicht einer Seite ist umso größer, je mehr Seiten (mit möglichst hohem eigenen Gewicht) auf diese Seite verweisen. Das Gewicht g_i einer Seite i berechnet sich also aus den Gewichten g_j der auf i verlinkenden Seiten j. Verlinkt j auf insgesamt C_j verschiedene Seiten, so wird das Gewicht von g_j anteilig auf diese Seiten aufgeteilt. Mithilfe einer Matrix $T \in \mathbb{R}^{N\times N}$,

$$T_{ij} := \begin{cases} 1/C_j, & \text{falls } L_{ij} = 1, \\ 0 & \text{sonst,} \end{cases}$$

definiert folgende Rekursionsformel nun den PageRank Algorithmus:

$$g_i \qquad = \frac{1-p}{N} + p \sum_{\forall j:\, L_{ij}=1} \frac{g_j}{C_j} \qquad = \frac{1-p}{N} + p(T\mathbf{g})_i$$

Der Parameter p wird dabei als Dämpfungsfaktor eingeführt und stellt die Wahrscheinlichkeit dar, dass der Zufalls-Surfer überhaupt den Links folgt. Mit einer Wahrscheinlichkeit $(1-p)$ kann er sich, beispielsweise aus Langeweile, für eine beliebige Webseite entscheiden. Ein gebräuchlicher Wert für p ist $p = 0.85$, den Sie im Folgenden verwenden sollen.
Obige Rekursionsformel kann entweder über ein lineares Gleichungssystem

$$(I - pT)\mathbf{g} = \mathbf{f}$$

mit der rechten Seite $\mathbf{f}$ und $\mathrm{f}_i = \frac{1-p}{N}$ oder über ein Eigenwertproblem gelöst werden.
Für eine Matrix $E \in \mathbb{R}^{N\times N}$ mit $E_{ij} = 1$ löst der normierte Gewichtsvektor $\mathbf{g}$ mit $\|\mathbf{g}\|_1 := \sum_{i=1}^N |g_i| = 1$ das Gleichungssystem

$$G\mathbf{g} := \left(\frac{1-p}{N} E + pT\right) \mathbf{g} = \mathbf{g}. \tag{6}$$

Der in der $\| \ \|_1$-Norm normierte Gewichtsvektor $\mathbf{g}$ ist also Eigenvektor der *Google-Matrix* G zum Eigenwert 1.
Im Folgenden werden wir uns auf dieses Eigenwertproblem konzentrieren. Wir werden uns zunächst auf Netzwerke beschränken, in denen jede Seite einen Outdegree größer null hat (auch bezeichnet als Netzwerke ohne *hängende Knoten*), d.h. auf zumindest eine andere Seite weiterverlinkt und die Adjazenzmatrix keine Nullspalte besitzt.

(a) Zeigen Sie, dass die zu einem solchen Neztwerk gehörige Google-Matrix für $p < 1$ nur positive Einträge besitzt und spalten-stochastisch ist, d.h. alle Einträge einer Spalte sind nicht negativ und addieren sich zu eins.

(b) Zeigen Sie, jede spalten-stochastische Matrix G besitzt 1 als Eigenwert (Hinweis: Berechnen Sie $G^T\mathbf{e}$ mit $\mathbf{e} = (1, \ldots, 1)^T$).

(c) Zeigen Sie, dass jeder Eigenvektor der Matrix G zum Eigenwert 1 nur positive oder nur negative Einträge besitzt. (Hinweis: Sie nehmen an, die Komponenten v_i eines Eigenvektors $\mathbf{v}$ besitzen verschiedene Vorzeichen. Folgern Sie nun aus der Eigenwertgleichung und der Positivität von G dass die Ungleichung $|\mathrm{v}_i| < \sum_j G_{ij}|\mathrm{v}_j|$ gilt.)

(d) Beweisen Sie, dass zu zwei linear unabhängige Vektoren $\mathbf{v}$ und $\mathbf{w}$ Zahlen s und t existieren, so dass $\mathbf{x} := s\mathbf{v} + t\mathbf{w}$ sowohl negative als auch positive Einträge besitzt. Folgern Sie nun daraus, dass die Diemension des Eigenraumes der Matrix G zum Eigenwert eins gleich eins ist.

Aus diesen Teilaufgaben kann nun Exsistenz und Eindeutigkeit der Lösung des Eigenwertproblems (6) für Netzwerke ohne hängende Knoten gefolgert werden. Mit etwas mehr Aufwand läßt sich nun zeigen, dass die Vektoriteration für Startvektoren mit nur positiven Einträgen immer gegen diese Lösung konvergiert (falls der Eigenvektor richtig normiert wird).
Liegt nun ein Netzwerk mit hängenden Knoten vor (vgl. das Netzwerk in Abb. 1), so besitzt die Matrix G zwar nur Eigenwerte die betragsmäßig kleiner gleich eins sind, 1 ist aber nicht notwenigerweise ein Eigenwert. Einen Gewichtsvektor $\mathbf{g}$ kann man aber trotzdem auf ähnliche Weise bestimmen. Es läßt sich zeigen, dass die Matrix G einen positiven Eigenwert besitzt, dessen Eigenvektor ausschließlich nichtnegative Einträge hat (der sog. *Perron Eigenvektor*). Dieser Eigenvektor ist ebenfalls über eine Vektoriteration bestimmbar, und er wird verwendet um die Seiten eines Netzwerkes mit hängenden Knoten zu reihen.

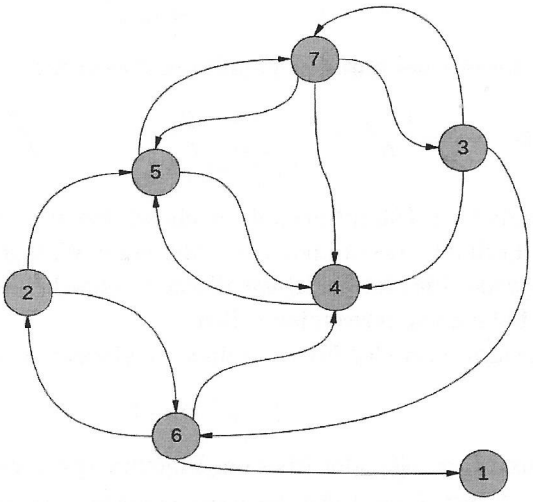

Abbildung 1: Beispiel-Netzwerk

(e) Implementieren Sie die Vektoriteration zur Berechnung des Eigenvektors $\mathbf{g}$ und des zugehörigen Eigenwerts. Die Iteration sollte dann abbrechen, wenn für den Eigenvektor im Iterationsschritt k gilt $\|\mathbf{g}^{(k)} - \mathbf{g}^{(k-1)}\|_1 \leq 10^{-12}$. Testen Sie Ihren Code für Googlematrizen verschiedener Dimension N. Verwenden Sie dabei Adjazenzmatrizen, die Sie mit Zufallszahlen (d.h. zufällig mit Einsen und Nullen) besetzen. Plotten Sie für verschiedene Dimensionen die Größe $\|\mathbf{g}^{(k)} - \mathbf{g}^{(k-1)}\|_1$ über k. Was können Sie über den zweitgrößten Eigenwert von G aussagen? Wie ändert sich die Iterationszahl mit der Dimension?

(f) Wieviele Operationen benötigt ein Iterationsschritt der Vektoriteration bei einer Dimension N, wenn man annimmt, dass jede Seite maximal $K \in \mathbb{N}$ Links auf andere Seiten enthält.

(g) Implementieren Sie eine Funktion `webrank`, die für eine vorgegebene Adjazenzmatrix L eine Reihung der Seiten über die Vektoriteration bestimmt, und das Ranking der Seiten in absteigender Reihenfolge ausgibt.

(h) *Suchmaschinenoptimierung:*
Abbildung 1 zeigt ein kleines Netzwerk aus Webseiten und ihren Verlinkungen. Erstellen Sie das

zugehörige Ranking.
Angenommen Sie sind Besitzer der Homepage Nummer 1. Können Sie Ihr Ranking durch geeignetes setzen von Links verbessern? Führen Sie sukzessive zusätzliche Seiten ein um Ihren Rang zu verbessern und geben Sie die zugehörigen Verlinkungen an. Wie viele zusätzliche Seiten müssen Sie einführen um Ihre Seite auf Rang 1 zu bringen? Beachten Sie dabei, dass Sie lediglich Links auf Seiten in Ihrem Besitz verändern können.