Prof. Dr. Karsten Urban
Dr. Timo Tonn
Dipl.-Math. oec. Klaus Stolle
Institut für Numerische Mathematik
Universität Ulm

WiSe 2023/2024
20.12.2023

Angewandte Numerik

Übungsblatt 09

Besprechung in den Tutorien von Montag, 08.01.2024 bis Freitag, 12.01.2024

Für dieses Übungsblatt gibt es 12 Theorie- und 27 Matlab-Punkte, sowie 7 Theorie- und 15 Matlab-Zusatzpunkte. Punkte, die mit einem * gekennzeichnet sind, sind Zusatzpunkte.
Die 70-Prozent-Grenzen liegen aktuell (inklusive Blatt 09) bei 117,6 Theorie- und 112,7 Matlab-Punkten.

Aufgabe 33 *(Programmieraufgabe: QR-Aufdatierung: Rang-1-Update)* b

e
(6M+2T+3M*+4M*+2T* Punkte)

a) Schreiben Sie eine MATLAB-Funktion `[Q, R] = myQrUpdateRank1(Q, R, u, v)`, mit der Sie eine gegebene QR-Zerlegung $A = Q \cdot R$ einer Matrix $A \in \mathbb{R}^{m \times n}$ mit $m, n \in \mathbb{N}$, $m \geq n$ effizient aktualisieren können. Ihre MATLAB-Funktion `myQrUpdateRank1` soll die QR-Zerlegung der Matrix $\tilde{A} = A + uv^T$ zurückliefern, wobei $u \in \mathbb{R}^m$ und $v \in \mathbb{R}^n$ zwei Vektoren sind.

   Achten Sie darauf, dass Ihre MATLAB-Funktion `myQrUpdateRank1` die neue QR-Zerlegung mit quadratischem Aufwand berechnet.

b) Begründen Sie, warum Ihre MATLAB-Funktion `myQrUpdateRank1` die neue QR-Zerlegung mit quadratischem Aufwand berechnet.

c) Testen Sie Ihre MATLAB-Funktion `myQrUpdateRank1` an verschiedenen Matrizen A unterschiedlicher Dimensionen. Überprüfen Sie dabei jeweils auch, ob die Matrix Q orthogonal und ob die Matrix R eine rechte obere Dreiecksmatrix ist.

d) Vergleichen Sie die Laufzeiten Ihrer MATLAB-Funktion `myQrUpdateRank1` mit den für eine volle Neuberechnung der QR-Zerlegung benötigten Laufzeiten.

   Schreiben Sie dazu ein MATLAB-Skript `laufzeitenA33`, in dem Sie für $n \in \{2^i \mid i = 3, \ldots, 10\}$ und $m \geq n$ (beispielsweise $m = \frac{3}{2}n$) jeweils eine Matrix $A \in \mathbb{R}^{m \times n}$ und Vektoren $u \in \mathbb{R}^m$ und $v \in \mathbb{R}^n$ aufstellen. Berechnen Sie mit Ihrer MATLAB-Funktion `qrGivens` aus Aufgabe 23 von Übungsblatt 07 (Sie dürfen alternativ die entsprechende Funktion aus der Vorlesung verwenden) und der MATLAB-Funktion `qr_Householder` aus dem Material zu Übungsblatt 07 jeweils eine QR-Zerlegung der Matrix A sowie mit Ihrer MATLAB-Funktion `myQrUpdateRank1` eine QR-Zerlegung der Matrix $\tilde{A} = A + uv^T$.

   Plotten Sie die zur Berechnung der QR-Zerlegungen benötigten Zeiten über die Dimension `n` der Matrix in einem Schaubild mit logarithmischen Achsen. Zeichnen Sie in Ihr Schaubild geeignete Steigungsgeraden ein.

   Beachten Sie, dass Ihr MATLAB-Skript je nach Leistungsfähigkeit Ihres Computers zur Berechnung längere Zeit brauchen wird. Falls möglich, können Sie auch Matrizen größerer Dimensionen betrachten.

e) Interpretieren Sie Ihr Schaubild.

Aufgabe 34 *(Programmieraufgabe: Newton-Verfahren für skalare Funktionen)*

(3T*+3M+2M+2T+2T+2M+1T+1T Punkte)

a d e f g

a) Lösen Sie das nichtlineare Gleichungssystem $x^2 = 5$ näherungsweise, indem Sie von Hand drei Iterationen des Newton-Verfahrens durchführen. Verwenden Sie als Startwert $x_0 = 1$.

b) Schreiben Sie eine MATLAB-Funktion `xk = newtonSkalar(f, df, x0, toly, maxIt)`, die eine Nullstelle einer Funktion f mit Hilfe des Newton-Verfahrens näherungsweise berechnet. Dabei soll der Parameter `f` die Funktion f als *function handle*, der Parameter `df` die Ableitung der Funktion f als *function handle* und der Parameter `x0` der Startwert sein. Die weiteren Parameter `toly` und `maxIt` sowie der Rückgabewert `xk` sollen die gleiche Bedeutung wie bei den MATLAB-Funktionen `regulaFalsi` und `sekanten` aus Aufgabe 32 vom letzten Übungsblatt 08 haben.

c) Erweitern Sie Ihr MATLAB-Skript `testKonvergenz` und Ihre Grafik aus Aufgabe 32 b) auch um das Newtonverfahren. Wählen Sie als Startwert $x_0 = 3$.

d) Interpretieren Sie das Ergebnis. Welche Konvergenz im Sinne der Definition 5.0.1 (Konvergenzgeschwindigkeit) auf Folie 8 (Definition 5.0.5 auf Seite 65 im Skript) liegt vor?

   Begründen Sie Ihre Aussage.

e) Variieren Sie den Startwert für das Newton-Verfahren. Testen Sie beispielsweise auch den Startwert $x_0 = 0$. Wie erklären Sie sich das Ergebnis?

f) Erweitern Sie nun auch Ihr MATLAB-Skript und Ihre Grafik aus Aufgabe 32 e) um das Newtonverfahren. Wählen Sie als Startwert wieder $x_0 = 3$.

g) Interpretieren Sie auch dieses Ergebnis. Liegt Konvergenz vor? Falls ja, mit welcher Konvergenzordnung? Begründen Sie Ihre Aussage.

h) Wie passt das beobachtete Ergebnis zu den Aussagen aus der Vorlesung, insbesondere zu Satz 5.1.5 (Konvergenz Newton-Verfahren, Folie 18 bzw. Seite 74 im Skript)?

Aufgabe 35 *(Programmieraufgabe: Konvergenzbereiche des Newton- und des Sekanten-Verfahrens)*

(4M+2T+3M+2M+4M+1M+2T+5M*+3M*+2T* Punkte)

b g j

In dieser Aufgabe wollen wir für das Newton- und das Sekanten-Verfahren die Abhängigkeit der Konvergenz von den Startwerten untersuchen.

a) Schreiben Sie ein MATLAB-Skript `konvergenzBereiche.m`, welches für das Newton-Verfahren und für die Funktion f mit $f(x) = \arctan x$ die Abhängigkeit der Anzahl der durchgeführten Iterationen vom Startwert veranschaulicht. Wählen Sie dazu 200 Startwerte $x_0 \in [-10, 10]$ und bestimmen Sie für jeden Startwert die Anzahl der durchgeführten Iterationen. Visualisieren Sie die Ergebnisse mittels einer Grafik (x-Achse: Startwert, y-Achse: Anzahl der Iterationen). Verwenden Sie `toly = 1e-10`.

   Sie können Ihre MATLAB-Funktion `xk = newtonSkalar(f, df, x0, toly, maxIt)` aus Aufgabe 34 verwenden.

b) Was fällt auf? Wie interpretieren Sie die Anzahl der Iterationen für betragsmäßig große Startwerte. Erklären Sie beispielhaft an einigen Startwerten den jeweiligen Verlauf der Iteration.

c) Überlegen Sie sich, wie Sie für jeden Startwert überprüfen können, ob das Newtonverfahren mit der vorgegebenen Toleranz sowie innerhalb der maximalen Iterationen konvergiert und einen Näherungswert für die Nullstelle liefert. Visualisieren Sie den Konvergenzbereich in Ihrem Schaubild.

d) Betrachten Sie das Newton-Verfahren für den Startwert $x0 = 1.3917452002707349244$. Wieviele Iterationen werden durchgeführt? Woran liegt das?

e) Untersuchen Sie analog zu Aufgabenteil a) auch das Sekanten-Verfahren. Wählen Sie als zweiten Startwert $x_1 = -5$, falls $x_0 \geq 0$, und $x_1 = +5$, falls $x_0 < 0$. Visualisieren Sie den Konvergenzbereich des Sekantenverfahrens analog zu Aufgabenteil c).

f) Vertauschen Sie die Startwerte des Sekantenverfahrens. Wählen Sie also $x_0 = \pm 5$ und $x_1 \in [-10, 10]$.

g) Interpretieren Sie Ihre Ergebnisse. Für welche Startwerte konvergiert das Sekantenverfahren? Unterscheiden sich die Ergebnisse des Aufgabenteils e) und des Aufgabenteils f)? Falls ja, warum?

h) Zur Lösung eines Nullstellenproblems ist es durchaus üblich, verschiedene Verfahren für die näherungsweise Berechnung von Nullstellen zu kombinieren. Einerseits würde das Bisektionsverfahren für die in dieser Aufgabe betrachtete Funktion f mit $f(x) = \arctan x$ mit den Startwerten aus Aufgabenteil e) oder Aufgabenteil f) immer zu einer Näherungslösung finden. Andererseits würde beispielsweise das Sekantenverfahren, falls es konvergiert, die Näherungslösung schneller ermitteln.

Das *Dekker*–Verfahren ist ein Verfahren zur numerischen Berechnung von Nullstellen, welches die Vorteile des Bisektionsverfahrens und des Sekantenverfahrens vereint. Der Algorithmus des Dekkerverfahrens lautet:

Algorithmus 1 Dekker-Verfahren

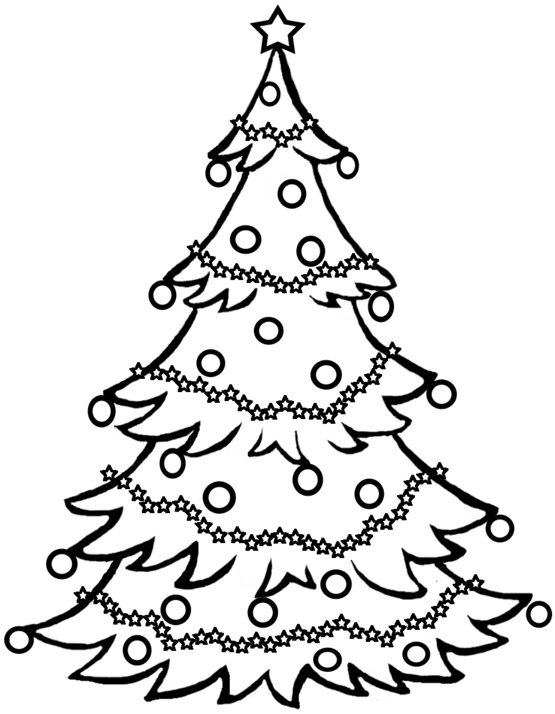
Require: $I = [a, b]$ and $f \in C(I)$ with $f(a)f(b) < 0$, $\texttt{tol} > 0$.

$a_0 = a$ and $b_0 = b$ and $b_{-1} = a_0$
for $k = 0, 1, 2, \ldots$ **do**
    Compute $s = b_k - \frac{b_k - b_{k-1}}{f(b_k) - f(b_{k-1})} f(b_k)$ and $m = \frac{a_k + b_k}{2}$.
    if s between b_k and m **then**
        $b_{k+1} = s$
    else
        $b_{k+1} = m$
    end if
    if $|f(b_{k+1})| \leq \texttt{tol}$ **then**
        `STOPP`
    end if
    if $f(b_{k+1})f(a_k) < 0$ **then**
        $a_{k+1} = a_k$
    else
        $a_{k+1} = b_k$
    end if
end for

Schreiben Sie eine MATLAB-Funktion `xk = dekker(f, a, b, tol, maxIt)`, die das Dekkerverfahren implementiert. Die Parameter und der Rückgabewert sollen den Parametern und dem Rückgabewert Ihrer MATLAB-Funktion `sekanten` entsprechen.

i) Untersuchen Sie analog zu Aufgabenteil a) und Aufgabenteil e) auch das Dekker-Verfahren. Wählen Sie die Startwerte wie beim Sekantenverfahren. Vertauschen Sie auch beim Dekker-Verfahren die Startwerte. Visualisieren Sie den Konvergenzbereich des Dekkerverfahrens analog zu Aufgabenteil c).

j) Interpretieren Sie Ihre Ergebnisse. Für welche Startwerte konvergiert das Dekkerverfahren? Wie viele Iterationen benötigt das Dekkerverfahren im Vergleich zum Sekantenverfahren?



Wir wünschen Frohe Weihnachten und einen guten Rutsch!

Hinweise:

Teamarbeit ist bei der Bearbeitung der Übungsblätter erlaubt und erwünscht. Falls Sie die Aufgaben im Team lösen, muss jedes Teammitglied eine eigene Lösung votieren und abgeben. Geben Sie dann bitte auf Ihrer Lösung alle an der Aufgabe beteiligten Teammitglieder an.

Die Programmieraufgaben sind in MATLAB zu lösen. Der Source Code muss strukturiert und dokumentiert sein.

Speichern Sie Ihre Lösungen der Theorieaufgaben, Ihre MATLAB-Programme und Ihre Ergebnisse in einem Directory mit dem Namen **Blatt09_*Vorname*_*Nachname*** und verpacken Sie dieses in eine `.zip`-Datei. **Spätestens bis Sonntag, 07.01.2024, 23:59 Uhr** müssen Sie diese `.zip`-Datei in Moodle hochladen und für die Aufgabenteile, die Sie vorstellen können, votieren.

Aufgabe 33
b)
Die Funktion berechnet das Update in $O(n^2)$, da (n-2) Givensrotationen mit einer Komplexität von O(n) und nochmal (n-1) weitere Givensrotationen ausgeführt werden. Daraus folgt:
$(n-2) * O(n) + (n-1) * O(n) = O(n^2) + O(n^2) = O(n^2)$

c) Wenn die Ableitung niemals oder für die verwendeten Werte null wird, kann eine Lösung immer berechnet werden, hierfür ist aber dennoch wichtig, dass der Startwert in naher Umgebung zur Lösung liegt, da sonst ein falscher Wert errechnet wird oder das Verfahren divergiert.

e) es ist sehr leicht zu erkennen, dass das Rang-1 QR-Update deutlich schneller eine QR Zerlegung berechnen kann. Es fällt auch auf das die QR-Zerlegung über Householder-Matrizen deutlich schneller ist, da wir eine vollbesetzte Matrix besitzen.

Aufgabe 34

a) $x^2 = 5 \Rightarrow f(x) = x^2 - 5 = 0 \qquad x_0 = 1$

$f'(x) = 2x$

1. : $x_0 = 1$

$$x_1 = 1 - \frac{f(1)}{f(1)} = 1 - \frac{-4}{2} = 1 + 2 = 3$$

2:

$$x_0 = 3$$

$$x_1 = 3 - \frac{f(3)}{f'(3)} = 3 - \frac{4}{6} = \frac{8}{6} = \frac{4}{3}$$

3:

$$x_0 = \frac{4}{3}$$

$$x_1 = \frac{4}{3} - \frac{f(4/3)}{f'(4/3)} = \frac{4}{3} - \frac{\left(\frac{16}{9} - 5\right)}{\frac{8}{3}} = \frac{4}{3} - \frac{-\frac{29}{9}}{\frac{8}{3}} = \frac{4}{3} + \frac{29}{\cancel{9}_{3}} \cdot \frac{\cancel{3}^{1}}{8}$$

$$= \frac{4}{3} + \frac{29}{24} = \frac{32}{24} + \frac{29}{24}$$

$$= \frac{61}{24}$$

d)
Es handelt sich um eine Konvergenz quadratischer Ordnung, da das Newtonverfahren merklich schneller konvergiert als das superlineare Sekanten verfahren.

e) An der Stelle ist df gleich null, wodurch matlab in der Berechnung für den Wert x1 durch Null

g)
Hier handelt es sich um eine superlineare Konvergenz, da das Newtonverfahren schneller fällt als das linear konvergierende Sekantenverfahren.

h)
Die Beispiele decken das beobachtet Verhalten, da alle Funktionen entsprechend konvergieren, bis auf das Newtonverfahren welches nicht für das zweite Beispiel quadratisch konvergiert.

Aufgabe 35

b)
Größere Startwerte (weiter von der Nullstelle weg) divergieren und in den einzelnen Iterationsschritten sieht man sehr gut, dass mit jeder weiteren Iteration die Werte weiter im steigen / fallen und sich immer weiter von der Nullstelle entfernen.

c)
Anfangswert sollte relativ nah an der tatsächlichen Nullstelle liegen.

g)
Das Sekantenverfahren konvergiert einmal nicht für Werte zwischen [-2.16.. -0.75..] und [2.16.. 0.75..] für die normalen Grenzen und $|x| > 6.88...$ für die vertauschten Grenzen.
Für die vertauschten Grenzen könnte der Grund in den sehr flachen Kurven liegen, aber warum ein unterschiedliches Verhalten auftritt kann ich nicht erklären.

j)
Das Dekkerverfahren konvergiert immer, aber ist sehr unzuverlässig wie viele Iterationsschritte gebraucht werden und brnötigt gegenüber dem Newton- und Sekantenverfahren deutlich mehr Iterationsschritte bis eine Lösung bestimmt wurde.