MATHEMATISCHES INSTITUT
PROF. DR. CHRISTIANE HELZEL
PAWEL BUCHMÜLLER

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

18. FEBRUAR 2016

Schriftliche Prüfung zur Computergestützten Mathematik zur Linearen Algebra

(PO 2014: Erste Klausur / PO 2008: Klausur)

Bitte folgende Angaben ergänzen und **DEUTLICH LESBAR** in Druckbuchstaben schreiben:

Name: ..... Vorname: .....

Matrikel-Nr.: ..... Studienfach: .....

Fachsemester: .....

Account zur Klausur: ..... Nummer des Computers: .....

Hiermit bestätige ich, dass ich zu dieser schriftlichen Prüfung zugelassen bin, da ich

☐ die Zulassung im WS 2015/2016 erworben habe,

☐ an einer schriftlichen Prüfung zur Computergestützten Mathematik zur Linearen Algebra bei ..... im WS/SS ..... teilgenommen, aber nicht bestanden habe,

☐ die Zulassung zur Prüfung im WS/SS ..... erworben habe.

.....
Unterschrift

Aufgabe	1	2	3	4	5	$\sum$
max. Punkte	10	10	10	15	15	60
err. Punkte						

Zum Bestehen der Klausur sind 30 Punkte hinreichend.

Wichtige Hinweise zur Klausur

- In Ihrem Home-Verzeichnis finden Sie die Datei `WerBinIch.m` sowie die Dateien `vorname.txt`, `nachname.txt`, `aufgabe1.m`, `aufgabe2.m`, `aufgabe3a.m`, `aufgabe3b.m`, `aufgabe3c.m`, `aufgabe4a.m`, `aufgabe4b.m`, `aufgabe4c.m`, `aufgabe5a.m`, `aufgabe5b.m`.
- **Ergänzen Sie zuerst die Datei `WerBinIch.m` mit Ihrem Namen, Vornamen, usw.**
- Als Hilfsmittel dürfen Sie ein von Ihnen selbst erstelltes DIN-A4-Blatt verwenden. Im Order `VL/` in Ihrem Home-Verzeichnis finden Sie darüber hinaus die Dateien der Vorlesungen von der Website.
- Es werden nur Lösungsvorschläge gewertet, die in Dateien mit dem jeweils in der Aufgabe angegebenen Namen in Ihrem Home-Verzeichnis gespeichert sind. Speichern Sie daher in kurzen Abständen Ihre Lösungen, um ggf. den Verlust von Daten zu vermeiden, falls MATLAB einmal abstürzt.
- Anders als bei den Übungsaufgaben werden bei den Klausuren in der Regel keine Testfälle vorgegeben. Diese müssen Sie sich ggf. selbst konstruieren, um Ihre Implementierungen zu überprüfen.
- Ein nicht lauffähiger Code kann in der Regel höchstens die Hälfte der Punkte erzielen. Kommentieren Sie ihren Code sinnvoll, so dass nachvollzogen werden kann, was Sie tun wollten.
- Die vorgegebene Form einer zu implementierenden Funktion muss ggf. eingehalten werden (das heißt keine zusätzlichen Eingaben oder Rückgaben).
- Fehlerabfragen vom Typ "überprüfen Sie, ob die Eingabematrix symmetrisch ist" oder "geben Sie einen Fehler aus, falls..." werden in der Aufgabenstellung explizit gefordert. Fehlermeldungen sollen das aufgetretene Ereignis sinnvoll charakterisieren.
- Schreiben Sie Ihre MATLAB-Skripte so, dass sie auch nach einem `clear all` noch lauffähig sind.
- Sie haben 120 Minuten Zeit für die Bearbeitung der Klausur.

Aufgabe 1: (10 Punkte)

Schreiben Sie eine Funktion

```
function M = aufgabe1(n)
```

in die Datei `aufgabe1.m`.

(a) Die Funktion soll mit Hilfe des Befehls `diag` eine $n \times n$ Matrix der Form

$$M = \begin{pmatrix} -2 & 1 & 0 & 0 & \cdots & 0 & 1 \\ 1 & -2 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 & -2 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 1 & -2 \end{pmatrix}.$$

konstruieren und zurückgeben.

(b)
- Falls $n > 0$ und $n \notin \mathbb{N}$ ist, dann soll auf die nächst kleinere natürliche Zahl gerundet werden, d.h. die Dimension der zu erzeugenden Matrix sei $m = \max\{s \in \mathbb{N}; s < n\}$.
- In diesem Fall soll der Benutzer darauf hingewiesen werden, dass die übergebene Zahl gerundet wurde.
- Ist $n \leq 0$, soll eine Fehlermeldung ausgegeben werden.

Aufgabe 2: (10 Punkte)

Das Sekantenverfahren ist gegeben durch die Iterationsvorschrift

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \cdot f(x_n)$$

und konvergiert (unter gewissen Bedingungen) gegen eine Nullstelle von f.

(a) Schreiben Sie eine Funktion

```
function [x, n, e] = aufgabe2(f,x0,x1,tol)
```

in die Datei `aufgabe2.m`, die mit Hilfe dieser Iterationsvorschrift die Nullstelle von f approximiert.

- `tol` gibt an, wie genau die Näherung sein soll. D.h. die Iteration bricht ab, falls $\|f(x_n)\| < tol$ oder $\|x_{n+1} - x_n\| < tol$.
- Da das Verfahren nicht immer konvergiert, sollen maximal 100 Iterationen erlaubt sein.
- Neben dem berechneten Wert für die Nullstelle x und der Anzahl der Iterationen n, soll auch der geschätzte Fehler $e = \|x_{n+1} - x_n\|$ zurückgegeben werden.

(b) Fügen Sie einen Hilfe-Text hinzu, d.h. der Befehl `help aufgabe2` soll eine kurze Beschreibung der Funktion liefern.

---

bitte wenden

Aufgabe 3: (10 Punkte)

(a) Schreiben Sie eine Funktion

```
function z = aufgabe3a(x,y)
```

in die Datei `aufgabe3a.m`. Die Funktion soll zu gegebenen Werten x und y einen Wert $z = Re(e^{x+iy})$ berechnen.

(b) Schreiben Sie eine Funktion

```
function z = aufgabe3b(x,y)
```

in die Datei `aufgabe3b.m`. Die Funktion soll zu gegebenen Werten x und y einen Wert $z = Im(e^{x+iy})$ berechnen.

(c) Schreiben Sie ein MATLAB-Script in die Datei `aufgabe3c.m`, welches die zuvor definierten Funktionen in einem gemeinsamen Fenster nebeneinander darstellt. Beachten Sie dabei die folgenden Punkte

- x soll im Intervall $[-3, 3]$ liegen. Verwenden Sie 20 äquidistante Punkte.
- y soll im Intervall $[-9, 6]$ liegen. Verwenden Sie 40 äquidistante Punkte.
- Beschriften Sie beide Plots.
- Schränken Sie die z Variable bei der Darstellung auf das Interval $[-20, 20]$ ein.

Aufgabe 4: (15 Punkte)

Gegeben sind zwei Polynome a_ℓ und b_ℓ in x, die für $\ell = 1, \ldots$ rekursiv durch

$$\begin{aligned} a_0(x) &= 1, \quad a_1(x) = 2 + x/2, \quad a_{\ell+1}(x) = (2+x)a_\ell(x) + (x/2)^2 a_{\ell-1}(x) \\ b_0(x) &= 0, \quad b_1(x) = 1, \qquad\quad b_{\ell+1}(x) = (2+x)b_\ell(x) + (x/2)^2 b_{\ell-1}(x). \end{aligned}$$

definiert sind.

(a) Schreiben Sie eine Funktion

```
function z = aufgabe4a(x,n)
```

in die Datei `aufgabe4a.m`, die für einen Vektor x und eine natürliche Zahl n **rekursiv** $a_n(x)$ berechnet.

(b) Schreiben Sie eine Funktion

```
function z = aufgabe4b(x,n)
```

in die Datei `aufgabe4b.m`, die für einen Vektor x und eine natürliche Zahl n, $b_n(x)$ mit einer Schleife berechnet, also **nicht** rekursiv.

(c) Schreiben Sie ein MATLAB-Script in die Datei `aufgabe4c.m`, welches die Polynome $a_n(x), b_n(x)$ für $n = 2, 4$ in einem Plot auf dem Intervall $[-3, 0]$ darstellt.

- Jedes Polynom soll in einer anderen Farbe dargestellt werden.
- Stellen Sie die $b_n(x)$ gestrichelt dar.
- Fügen Sie eine Legende in die linke obere Ecke ein.

bitte wenden

Aufgabe 5: (15 Punkte)

Für eine Übungsaufgabe im WS16/17 benötigen wir eine einfache Notenliste. Wir haben bereits eine Liste mit möglichen Vornamen `vornamen.txt` sowie mit möglichen Nachnamen `nachnamen.txt` vorbereitet.

(a) Schreiben Sie eine Funktion

```
function name = aufgabe5a(datei)
```

in die Datei `aufgabe5a.m`, die eine solche Liste (`vornamen.txt`/`nachnamen.txt`) einliest und ein Cell Array mit den Namen zurückgibt.

(b) Schreiben Sie ein MATLAB-Skript `aufgabe5b.m`, in dem Sie folgendes umsetzen:

- Lassen Sie die Dateien `vornamen.txt` und `nachnamen.txt` mit Hilfe von `aufgabe5a(datei)` einlesen.
- Kombinieren Sie jeden Vornamen mit jedem Nachnamen und speichern Sie diese in das "Feld" `name` der "Struktur" `student`.
  Hinweis: Sie erhalten 100 Kombinationen.
- Erzeugen Sie nun für jeden Studenten eine sechsstellige, zufällige Matrikelnummer und speichern Sie diese in das "Feld" `mat` der "Struktur" `student`.
- Erzeugen Sie nun für jeden Studenten eine zufällige Note und speichern Sie diese in das "Feld" `note` der "Struktur" `student`. Zulässig sind natürlich nur die üblichen Noten $1.0, 1.3, 1.7, 2.0, ..., 3.7, 4.0$ und 5.0.
- Erzeugen Sie nun die Datei `liste.txt` mit Name, Matrikelnummer und Note. Die Datei soll wie folgt formatiert sein.

```
Student/in  Mat.Nr.   Note
Schmidt Peter  216497  2.0
Schneider Peter  164548  1.7
...
```

Hinweis: Falls Sie mit dem 3. und/oder 4. Unterpunkt Probleme haben, dann verwenden Sie 999999 als Matrikelnummer und 2.0 als Note, um den 5. Unterpunkt bearbeiten zu können.