Algorithmische Methoden in der Numerik, SS 2022
Übungen für 17.6.2022

Die Beispiele sind gruppenweise (eine Gruppe hat höchstens 6 Mitglieder) zu lösen und bis zum 17.6.2022, 24:00 Uhr über Moodle abzugeben (1 Exemplar pro Gruppe).

Für die Lösung der Beispiele wird die Software `MATLAB` benötigt. Falls Sie diese Software nicht besitzen, kontaktieren Sie bitte für einen Zugang zu unseren Institutsrechnern unseren Systemadministrator Oliver Koch (SP II, Zi 349 , `koch@numa.uni-linz.ac.at`).

Die Übungen müssen bis zur angegebenen Deadline in Moodle in Form **einer einzigen Datei** im **zip (.zip)** Format gesendet werden. Diese Datei **muss** beinhalten:

- Eine PDF Datei welche aus folgenden Teilen besteht:
  1.) Deckblatt mit Namen der LVA und der Autoren (maximal 6 Studenten, Name, Matrikelnummer und email Adresse
  2.) Für jedes theoretische Beispiel ein PDF-File (kann auch eingescannt sein)
  3.) Für jeden Source File (d.h., m-file), ein Listing im PDF Format
  4.) Eine Beschreibung von allen durchgeführten Tests und Resultaten (Programminputs und –outputs), Antworten auf die gestellten Fragen usw.
  5.) Zusätzliche Erklärungen die Sie geben möchten.
- Jede Source Datei, damit Tests nachvollzogen werden können.

Der Name dieser Datei **muss** sein

Ueb1_AA_BB_CC_DD_EE_FF.zip

wobei AA, BB, ..., FF die Familiennamen der Studenten in alphabetischer Reihenfolge sind.
Jede Gruppe bestimmt eine/einen "Administratorin/Administrator", die/der diese Datei stellvertretend fr alle Gruppenmitglieder in Moodle abgibt. Diese Person ist auch fr die Weitergabe der korrigierten bungen an die anderen Gruppenmitglieder verantwortlich

3) Lesen Sie aufmerksam die MATLAB Dokumentation zu den angeführten Befehlen!!!

   (a) Erstellen Sie eine `MATLAB` Funktion

```
function [A, D, p, k] = QRFact(A)
```

   die für eine $m \times n$ Matrix A die QR Faktorisierung mit Pivotsuche gemäß Algorithmus 2.3 des Skriptums berechnet. Der Vektor p soll dabei die Permutationsmatrix P darstellen, sodass $QR = AP$. Die Diagonale von R wird auf dem Vektor D zurückgegeben und k soll der ermittelte Rang der Matrix A sein. Achten Sie dabei auf effiziente Programmierung, insbesonders dass Sie Schleifen vermeiden und durch die colon-Notation ersetzen (es lassen sich nicht alle Schleifen vermeiden). So kann z.B. das Pivotelement durch den Befehl

```
max(sigma(p(k:bar_n))./ bar_sigma(p(k:bar_n)))
```

   ermittelt werden.

   Testen Sie Ihre Funktion mindestens mit zufällig erzeugten Matrizen der Dimension $(2,2)$, $(4,2)$, $(10,5)$, $(1000,100)$, wobei nicht alle vollen Spaltenrang besitzen sollen: Wie erzeuge ich so eine Matrix mit bekanntem Rang k:

```
A=zeros(m,n);
A(:,1:k)=randn(m,k);   % besitzt Rang k mit Wahrscheinlichkeit 1
A(:,k+1:n)= A(:,1:k)*randn(k,n-k);
```

d.h. die Spalten $k+1,\dots,n$ sind eine zufällig erzeugte Linearkombination der Spalten $1,\dots,k$

(b) Erstellen Sie eine `MATLAB` Funktion

```
function [x] = QRSolve(B,D,p,k,b)
```

die eine least-sqares Lösung des Gleichungssystems $Ax = b$ berechnet für den Fall, dass die Matrix A vollen Spaltenrang A besitzt, wobei `[B, D, p, k] = QRFact(A)`. Falls A nicht vollen Spaltenrang besitzt, soll der Nullvektor zurückgegeben werden. Achten Sie wiederum auf effiziente Programmierung und die Vermeidung von Schleifen. Testen Sie Ihr Programm mit mehreren zufällig erzeugten A und b und vergleichen Sie Ihre Lösung mit der des MATLAB-Befehls `linsolve` (Wie groß ist die (absolute, relative) Abweichung zwischen den beiden Lösungsvektoren?)

(c) Erstellen Sie zwei `MATLAB` Funktionen

```
function [Q] = CompQ(B,p,k)
function [R] = CompR(B,D,p,k)
```

die die Matrizen Q und R zurückgeben, wobei `[B,D,p,k]=QRFact(A)`.
Für mehrere zufällig erzeugte Matrizen A berechnen Sie den relativen Fehler `norm(Q*R-A(:,p))/norm(A)` und vergleichen Sie dies mit dem relativen Fehler des MATLAB-Befehls `[Q,R,e] = qr(A,'vector')`