Übung 1 *Singulärwertzerlegung quadratischer Matrizen*

Satz: *Zu einer regulären Matrix $A \in \mathbb{R}^{n\times n}$ existieren orthonormale Matrizen $U, V \in \mathbb{R}^{n\times n}$ und eine Diagonalmatrix $\Sigma := diag(\sigma_1, ..., \sigma_n) \in \mathbb{R}^{n\times n}$ mit Zahlen $\sigma_1 \geq ... \geq \sigma_n > 0$, so dass gilt:*

$$A = V\Sigma U^T.$$

(Beweis: Buch von Wendland/Schabak oder Rannacher-Skript.)
Diese Faktorisierung heisst **Singulärwertzerlegung** von A und die Zahlen $\sigma_1, ..., \sigma_n$ werden **Singulärwerte** von A genannt.

(a) Zeige: Sei $x \in \mathbb{R}^n$ eine Linearkombination der Basis $u_1, ..., u_n \in \mathbb{R}^n$ aus Spaltenvektoren von U, also $x = \sum_{k=1}^n x_k u_k$. Dann gilt:

$$Ax = \sum_{k=1}^{n} x_k \sigma_k v_k,$$

wobei $v_1, ..., v_n \in \mathbb{R}^n$ die Spaltenvektoren von V sind.

(b) Berechne die Kondition von $\|A\|_2$, $\|A^{-1}\|_2$ und $\mathrm{cond}_2(A)$ mithilfe der Singulärwerte von A.

(4 Punkte)

Übung 2 *Schur-Komplement*
Es sei $\mathbb{K} = \mathbb{R}$ oder $\mathbb{K} = \mathbb{C}$. Gegeben sei eine Blockpartitionierung einer Matrix $A \in \mathbb{K}^{n\times n}$ von folgender Form:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}.$$

wobei die Teilmatrix $A_{11} \in \mathbb{K}^{p\times p}$ $(1 \leq p \leq n)$ invertierbar sei. Demnach ist $A_{12} \in \mathbb{K}^{p\times(n-p)}$, $A_{21} \in \mathbb{K}^{(n-p)\times p}$ und $A_{22} \in \mathbb{K}^{(n-p)\times(n-p)}$. Die Block-LR-Zerlegung von A lässt sich dann darstellen als

$$\begin{pmatrix} Id & 0 \\ A_{21}A_{11}^{-1} & Id \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ 0 & S \end{pmatrix}$$

mit entsprechend dimensionierten Einheitsmatrizen Id. Die Teilmatrix $S \in \mathbb{K}^{(n-p)\times(n-p)}$ wird *Schur-Komplement* von A_{11} in A genannt.

(a) Zeige:

$$S = A_{22} - A_{21}A_{11}^{-1}A_{12}$$

(b) Zeige: Wenn A zudem hermitesch und positiv definit sind, dann ist die Teilmatrix A_{11} regulär und sowohl A_{11} als auch S sind hermitesch und positiv definit.

(5 Punkte)

Übung 3 *LR-Zerlegung tridiagonaler Matrizen*

Gegeben sei eine Tridiagonalmatrix

$$A = \begin{pmatrix} a_1 & b_1 & & & 0 \\ c_2 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & b_{n-1} \\ 0 & & & c_n & a_n \end{pmatrix}$$

mit

$$\begin{array}{l} |a_1| > |b_1| > 0 \\ |a_j| \geq |b_j| + |c_j| > 0, \quad b_j, c_j \neq 0, \quad j \in \{2, ..., n-1\} \\ |a_n| \geq |c_n| > 0\ . \end{array}$$

Zeige: Der Algorithmus

$$\begin{array}{ll} r_1 := a_1 & \\ l_j := c_j / r_{j-1} & j \in \{2, ..., n\} \\ r_j := a_j - l_j b_{j-1} & j \in \{2, ..., n\} \end{array}$$

ist durchführbar, liefert die LR-Zerlegung

$$A = \begin{pmatrix} 1 & & & 0 \\ l_2 & 1 & & \\ & \ddots & \ddots & \\ 0 & & l_n & 1 \end{pmatrix} \begin{pmatrix} r_1 & b_1 & & 0 \\ & r_2 & \ddots & \\ & & \ddots & b_{n-1} \\ 0 & & & r_n \end{pmatrix}$$

und A ist invertierbar.

(4 Punkte)

Übung 4 *LR-Zerlegung mit Pivotsuche (Praktische Übung)*

a) Füge der Headerdatei `LR.hh` aus der letzten praktischen Übung drei neue Template-Funktionen hinzu:

```
template<typename T>
void row_equilibrate (hdnum::DenseMatrix<T>& A, hdnum::Vector<T>& s)
{...}
```

soll die Zeilenäquilibrierung der Matrix A durchführen. (Siehe Übungsblatt, Aufgabe 3a.) In den Vektor `s` sollen die einzelnen Zeilenbetragssummen abgespeichert werden. Diese werden benötigt, auch die rechte Seite b richtig zu skalieren. Dafür ist die Funktion

```
template<typename T>
void apply_equilibrate (const hdnum::Vector<T>& s, hdnum::Vector<T>& b)
{...}
```

gedacht.

Die Funktion

```
template<typename T>
void lr_partialpivot (hdnum::DenseMatrix<T>& A,
                      hdnum::Vector<std::size_t>& perm)
{...}
```

soll daraufhin die LR-Zerlegung von A *mit Spaltenpivotsuche* berechnen und das Ergebnis L und R wiederum in die Matrix `A` speichern. Der Indexvektor `perm` soll sich die Permutation der Zeilen merken. Diese Funktion soll die Funktion `void lr(...)` aus der letzten Übung ersetzen. Der Rest der Schritte zur Lösung eines linearen Gleichungssystems analog wie in der letzten praktischen Übung. (Siehe `testLR.cc` aus Übungsblatt 6.)

b) Schreibe ein neues C++-Programm, welches unter Benutzung der LR-Zerlegung

- ohne Spaltenpivotsuche
- mit Spaltenpivotsuche
- mit Spaltenpivotsuche und zusätzlich noch vorheriger Zeilenäquilibrierung

das Gleichungssystem

$$\begin{pmatrix} 10^{-16} & 0 & 10^{-16} & 3\cdot 10^{-16} \\ 1 & 10 & 0 & 23 \\ 4 & 12 & 12 & 0 \\ 5\cdot 10^{12} & 0 & 10^{12} & 10^{11} \end{pmatrix} \cdot x = \begin{pmatrix} 1.6\cdot 10^{-15} \\ 113 \\ 64 \\ 8.4\cdot 10^{+12} \end{pmatrix}$$

löst und vergleiche das numerische Ergebnis mit der exakten Lösung $x = (1,2,3,4)^T$.

c) *Bonusaufgabe:* Schreibe eine neue Funktion

```
template<typename T>
void lr_fullpivot( hdnum::DenseMatrix<T>& A,
                   hdnum::Vector<std::size_t>& p,
                   hdnum::Vector<std::size_t>& q )
{...}
```

zur LR-Zerlegung von *A mit totaler Pivotierung*. Dabei merkt man sich nicht nur die Zeilenvertauschungen in einem Indexvektor `p`, sondern auch die Spaltenvertauschungen in einem Indexvektor `q`. Die Zeilenvertauschungen in `p` müssen mittels der Funktion `void permute_forward()` nach `b` transportiert werden. Entsprechend müssen die Spaltenvertauschungen in `q` mittels einer neuen Funktion

```
template<typename T>
void permute_backward (const hdnum::Vector<std::size_t>& q,
                       hdnum::Vector<T>& x )
{...}
```

nach `x` transportiert werden!

(5+2 Punkte)