Serie 4

P4 (Task-parallele geblockte Matrix-Inversion) [10P.]

Ausgehend von der LR-Zerlegung ist es möglich einen effizienten Algorithmus zur Inversion von Matrizen zu entwickeln.
Sei $n \in \mathbb{N}$ und $A \in \mathbb{R}^{n \times n}$ besitze eine LR-Zerlegung. Blockweise gestaltet sich die LR-Zerlegung in folgender Form:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ A_{21}A_{11}^{-1} & 1 \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} - A_{21}A_{11}^{-1}A_{12} \end{pmatrix}$$

Unter Verwendung des sogenannten *Schur-Komplements* $S := A_{22} - A_{21}A_{11}^{-1}A_{12}$ sowie der Hilfsmatrizen $B_{12} := A_{11}^{-1}A_{12}$ und $B_{21} := A_{21}A_{11}^{-1}$ ergibt sich also die folgende Inverse:

$$\begin{aligned} A^{-1} &= \begin{pmatrix} A_{11} & A_{12} \\ 0 & S \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 \\ A_{21}A_{11}^{-1} & 1 \end{pmatrix}^{-1} \\ &= \begin{pmatrix} A_{11}^{-1} & -B_{12}S^{-1} \\ 0 & S^{-1} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -B_{21} & 1 \end{pmatrix} \\ &= \begin{pmatrix} A_{11}^{-1} + B_{12}S^{-1}B_{21} & -B_{12}S^{-1} \\ -SB_{21} & S^{-1} \end{pmatrix} \end{aligned}$$

Der Algorithmus besteht somit aus den folgenden Schritten:

(i) Inversion A_{11}.

(ii) Berechnung von $B_{12} := A_{11}^{-1}A_{12}$.

(iii) Berechnung von $B_{21} := A_{21}A_{11}^{-1}$.

(iv) Update von A_{22} zu $S := A_{22} - A_{21}A_{11}^{-1}A_{12} = A_{22} - B_{21}A_{12}$.

(v) Inversion von S.

(vi) Update von A_{21} zu $-S^{-1}B_{21}$.

(vii) Update von A_{12} zu $-B_{12}S^{-1}$.

(viii) Update von A_{11}^{-1} zu $A_{11}^{-1} + B_{12}S^{-1}B_{21}$.

Dieser Algorithmus ist eintragsweise im vorliegenden Programmcode in der Methode `void invert(pmatrix a)` implementiert.
Implementieren Sie nun einen Algorithmus, der die Matrix rekursiv in Blöcke unterteilt, bis die Größe der Teilblöcke eine gegebene Auflösung unterschreitet. Unterhalb dieser Auflösung kann die Inversion von Teilblöcken dann eintragsweise ausgeführt werden. Sie können sich dabei am Vorgehen in der `invert`-Methode orientieren. Parallelisieren Sie Ihren Algorithmus durch die Verwendung der Direktive `#pragma omp task`.
Testen Sie Ihre Implementierung für die Matrixgrößen $n \in \{512, 1024, 2048\}$ sowie ver-

schiedene Anzahlen von Unterteilungen pro Schritt und verschiedene Auflösungen.
Stellen Sie Ihre Ergebnisse graphisch dar (z. B. mit `Gnuplot`).

Abgabe der Übungen bis **Freitag, den 03.05.2019, um 18:00 Uhr** per OLAT.