T2.) (LR-Zerlegung an einem Beispiel)

a. Bestimmen Sie eine LR-Zerlegung der Matrix A. Geben Sie Ihren Rechenweg an.

$$A = \begin{pmatrix} 2 & -1 & -3 & 3 \\ 4 & 0 & -3 & 1 \\ 6 & 1 & -1 & 6 \\ -2 & -5 & 4 & 1 \end{pmatrix}$$

b. Bestimmen Sie die Lösung x des linearen Gleichungssystems $Ax = b$ unter Verwendung der LR-Zerlegung aus a., wobei die rechte Seite b gegeben ist durch:

$$b = \begin{pmatrix} 1 \\ -8 \\ -16 \\ -12 \end{pmatrix}.$$

T3.) (Hauptuntermatrizen und LR-Zerlegung)

Seien $n \in \mathbb{N}_{\geq 1}, A \in \mathbb{R}^{n \times n}$. Für alle $m \in \{1, \ldots, n\}$ sei die m-te Hauptuntermatrix H_m von A regulär, d.h. die Matrix

$$H_m := \begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mm} \end{pmatrix}$$

sei regulär.

Zeigen Sie, dass die Matrix A eine LR-Zerlegung besitzt.

➢ Zerlegen Sie A in geeignete Blöcke und zeigen Sie die Aussage mit einer vollständigen Induktion.

Bemerkung:

In den weit verbreiteten Softwarebibliotheken BLAS (***B****asic* ***L****inear* ***A****lgebra* ***S****ubprograms*) und LAPACK (***L****inear* ***A****lgebra* ***PACK****age*) werden Matrizen von Funktionen überwiegend in einem speziellen Format benutzt. Im Kontext dieser Übung werden wir uns an die dort verwendete Konvention halten.
Eine Matrix mit Zeilenanzahl $rows$ und Spaltenanzahl $cols$ wird als „Vektor“der Länge $rows * cols$ gespeichert, wobei die Einträge spaltenweise hintereinander im Speicher liegen:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \rightsquigarrow \boxed{a_{11} \mid a_{21} \mid a_{31} \mid a_{12} \mid a_{22} \mid a_{32} \mid a_{13} \mid a_{23} \mid a_{33}}.$$

P3.) (Nützliche Funktionen)

Schreiben Sie die Funktionen

`double`	`get_entry(double*` a, `int` $ldim$, `int` row, `int` col),
`void`	`set_entry(double*` a, `int` $ldim$, `int` row, `int` col, `double` $value$),
`void`	`print_matrix(double*` a, `int` $rows$, `int` $cols$),
`void`	`mvm(double*` a, `int` $rows$, `int` $cols$, `double*` x, `double*` y),

wobei `get_entry` den Eintrag (row, col) der Matrix a mit Zeilenanzahl $ldim$ (*leading dimension*) zurückgibt, `set_entry` an die Stelle (row, col) den Wert $value$ schreibt, `print_matrix` die Matrix a mit $rows$ Zeilen und $cols$ Spalten auf dem Bildschirm ausgibt und `mvm` das Matrix-Vektor-Produkt $a * x = y$ berechnet. Verwenden Sie hierzu gegebenenfalls die Datei `matrix_incomplete.c` und ergänzen Sie den fehlenden Quellcode.

➢ Achten Sie darauf, dass alle Funktionen auch für nicht-quadratische Matrizen funktionieren und der erste Eintrag einer Matrix mit $(0, 0)$ anzusprechen ist.

P4.) (Vorwärts- und Rückwärtseinsetzen)

Programmieren Sie mit Hilfe von **P3** die Funktionen

`void`	`forward_subst(int` $ldim$, `double*` l, `double*` b, `double` $*x$),
`void`	`backward_subst(int` $ldim$, `double*` r, `double*` b, `double*` x),

die mit einer unteren Dreiecksmatrix l ein Gleichungssystem durch Vorwärts- beziehungsweise mit einer oberen Dreiecksmatrix r ein Gleichungssystem durch Rückwärtseinsetzen lösen und überprüfen Sie die Ergebnisse mit der Funktion `mvm` aus Programmieraufgabe 3. Verwenden Sie hierzu gegebenenfalls die Dateien `forward_incomplete.c` beziehungsweise `backward_incomplete.c` und ergänzen Sie den fehlenden Quellcode.

➢ Fügen Sie in `forward_incomplete.c` und `backward_incomplete.c` gegebenenfalls Ihren Quellcode aus **P3** ein.

Abgabe: Die theoretischen Aufgaben **(T)** am Mo., d. 12.5.2014 bis 17 Uhr in bzw. vor unserem Sekretariat R906 im 9. Stock des Uni-Hochhauses. Die praktischen Aufgaben **(P)** bis Mo., d. 12.5.2014 bis 17 Uhr per Mail an `nal@informatik.uni-kiel.de`.